Fakultät für Maschinenwesen

Lehrstuhl für Flugsystemdynamik

Run-Time Monitoring of Model Reference Adaptive Controllers

Dipl.-Ing. (Univ.) Maximilian Mühlegg

Vollständiger Abdruck der von der Fakultät für Maschinenwesen
der Technischen Universität München
zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.) genehmigten Dissertation.

Vorsitzender: Prof. Dr.-Ing. habil. Boris Lohmann

Prüfer der Dissertation: 1. Prof. Dr.-Ing. Florian Holzapfel

2. Asst. Prof. Girish Chowdhary, Ph.D.

Die Dissertation wurde am 10.04.2019 bei der Technischen Universität München eingereicht und durch die Fakultät für Maschinenwesen am 04.11.2019 angenommen.

Acknowledgement

At first, I want to thank my advisor Professor Florian Holzapfel for providing me the opportunity to write this thesis as well as his continuous trust and support over the past years. I want to highlight the freedom, high level of responsibility and ever difficult challenges in the work environment set by him, which allowed me to significantly grow as a person. I want to thank my second examiner and mentor Dr. Girish Chowdhary for his advice, dedication and encouragement. He offered new perspectives, provided major opportunities and supported many endeavors. Furthermore, I would like to thank the chairman of my committee, Professor Boris Lohmann.

I would like to thank my colleagues at the Institut of Flight System Dynamics for the pleasant work environment and the teamwork in common challenges. In this context, I want to highlight Guillermo Falconi with whom I worked in projects from the beginning. Without the many many invested hours into core engineering tasks for the multirotor systems, important parts of this thesis would not have been possible. Special thanks to Thomas Raffler and Sravan Akkinapalli for their support in performing the experiments as testers and trustworthy pilots. Furthermore, I want to thank Benedikt Grüter, Patrick Lauffs, Volker Schneider, Markus Geiser, Christian Heise, Matthias Rieck and Simon Schatz among many others for their continuous support and contributions to the outstanding atmosphere at this institute.

I would like to thank my parents Ricarda and Johannes, my brother Carlo and my grandparents Karolina and Johann for their unwavering, continuous support over the past years. Without your backup, this thesis would not have been possible.

A special thanks to my friends for accepting my frequent absence, in particular in the last two years.

Finally, I want to thank my girlfriend Kathrin. I would not have been able to complete this thesis without her continuous support, patience and understanding. Thank you for standing by my side through all the ups and downs in the past years.

Kurzfassung

Hochautomatisierte unbemannte Fluggeräte sollen zukünftig verstärkt in kommerziellen Lufträumen zum Einsatz kommen. Um für kommerzielle Anwendungen attractive zu sein, müssen unbemannte Fluggeräte schnelle Produktentwicklungszyklen durchlaufen, kosteneffizient sein und zuverlässig in sich ändernden Umgebungsbedingungen operieren. Adaptive Regelungsansätze versprechen diese Anforderungen zu erfüllen. Allerdings existiert bis heute keine Möglichkeit der Zertifizierung dieser Algorithmen. In der Vergangenheit wurden adaptive Regler vornehmlich mit dem Ziel entworfen, Stabilität beim Einsatz auf einem Modellsystem zu garantieren. Die Robustheit der Ansätze wurde unter Beaufschlagung ausgewählter Unsicherheiten und Störungen untersucht. Die daraus resultierenden Beurteilungen garantieren allerdings keine verlässliche Performanz des realen System, welche wiederum Grundvoraussetzung für eine Zertifizierung ist. Zudem basiert die heutige Zertifizierung von Flugregelungsalgorithmen darauf die Algorithmen vor dem Einsatz zu evaluieren. Adaptive Regler passen sich den Umgebungsbedingungen während der Laufzeit an. Eine vorherige Evaluierung aller möglichen Systemzustände ist demnach nicht durchführbar. In diesem Kontext sehen viele Wissenschaftler eine online Überwachung des Reglers als integralen Teil einer zukünftigen Zertifizierung an. Die vorliegende Arbeit stellt ein online Monitoring Konzept für Model Reference Adaptive Control vor, mit dessen Hilfe Verletzungen der Anforderungen in der nahen Zukunft frühzeitig detektiert werden sollen. Um dies zu erreichen, prediziert der Ansatz unter Ausnutzung eines Modells des Systems und des adaptiven Reglers den Verlauf der Systemzustände in die Zukunft. Oftmals sind Teile der Systemdynamik mit Modellunsicherheiten behaftet oder komplett unbekannt. Gauß'sche Prozesse stellen eine Möglichkeit dar Modellunsicherheiten abzubilden, oder ein Modell aus online generierten Messdaten abzuleiten. Da Systeme durch stochastische Unsicherheiten beeinflusst sind, müssen auch probabilistische Propagationsmethoden benutzt warden. Das vorgeschlagene Konzept approximiert die Verteilungen der System- und Reglerzustände mit Hilfe von Gaussian Mixture Modellen, welche eine große Bandbreite an Verteilungen abbilden können. Die Prediktion erfolgt indem jede Kompenente des Mixture Modells durch ein korrespondierendes lineares Ersatzmodell der Systemgleichungen sowie des Gauß'schen Prozess propagiert wird. Die Wiederholung dieses Prozesses resultiert im wahrscheinlichen Verlauf der System- und Reglerzustände innerhalb des Prädiktionshorizonts. Die Analyse der kummulativen Verteilungsfunktion an mehreren Punkten im Prädiktionshorizont ermöglicht Aussagen über die Wahrscheinlichkeit einer Anforderungsverletzung. Die Fähigkeiten des vorgeschlagenen Monitoring-Konzepts werden mit Hilfe numerischer Simulationen und der Applikation auf Multirotorsysteme demonstriert.

Abstract

Autonomous unmanned aerial vehicles are envisioned to become increasingly utilized in commercial airspace. In order to be attractive for commercial applications, unmanned aerial vehicle are required to undergo a quick development cycle, ensure cost effectiveness and work reliably in changing environments. Adaptive control techniques have been proposed to meet these demands. However, up to date no consistent certification framework exists for adaptive controllers. In the past, most adaptive control algorithms were solely designed to ensure stability of a model system and meet robustness requirements against selective uncertainties and disturbances. These assessments do not guarantee reliable performance of the real system required by the certification process. Also due to their evolving nature, classical certification processes, which rely on prior verification, are not applicable. According to an emerging consensus, online monitoring will play an integral role in closing this gap. In this context, this thesis proposes an online monitoring concept for Model Reference Adaptive Controllers, which aims at the detection of imminent state requirement violation. This is achieved by predicting the evolution of the state and parameter trajectories employing a model of the plant and the controller. Often parts of the system dynamics are uncertain or completely unknown. Gaussian Processes present a way to represent such uncertainties or infer them from online gathered data using Bayesian inference. Since systems are subject to stochastic uncertainties, probabilistic propagation techniques are leveraged. The proposed monitoring concept approximates the state and parameter distribution using Gaussian Mixture Models, which can represent a wide class of distributions. The prediction is performed by propagating each mixture component through a respective approximated linear system model as well as the inferred Gaussian Process model. Repeating this process over the prediction horizon yields the evolution of the state and parameter distribution. Analyzing the cumulative distribution function of the GMM at various points in the prediction horizon allows for the assessment of the probability of an imminent state requirement violation. The proposed monitoring concept is demonstrated in numerical simulation as well as applied to multirotor systems in experiment.

Table of Contents

List of Figures x

List of Tables xv

List of Examples xvi

List of Acronyms xvii

List of Symbols xviii

1 Introduction 1

1.1 General Introduction 1

1.2 Contributions 4

1.3 Outline of the thesis 7

2 Basics of Gaussian Mixture Models and Gaussian Processes 8

2.1 Gaussian Mixture Models 8

2.1.1 Fitting a Gaussian Mixture Model 9

2.1.2 Relevant Properties of Gaussian Mixture Models 11

2.2 Gaussian Processes for uncertainty quantification 15

2.2.1 Gaussian Process Regression 20

2.3 Gaussian Process Evaluation at an uncertain input 27

2.3.1 Evaluation of a GP prior model with linear kernel functions at an uncertain input 28

2.3.2 Evaluation of a GP with squared exponential kernels at an uncertain input 31

3 System Description and Control Design 36

3.1 System Definition 36

3.2 Control Architecture 38

3.3 Model Reference Adaptive Control 40

3.3.1 Direct MRAC with gradient based update laws 41

3.3.2 Predictor based MRAC 46

3.3.3 Modifications, Extensions and Alternatives 51

3.4 The Certification Challenge of Model Reference Adaptive Controllers 56

4 Run-Time Monitoring of Adaptive Control Algorithms 59

4.1 Existing Monitoring and Prediction Strategies 61

4.1.1 Existing Monitoring Strategies for Adaptive Controllers 61

4.1.2 Approaches for Prediction and Uncertainty Propagation 62

4.2 Proposed Monitoring Strategy 67

4.3 Nonlinear Uncertainty Propagation using Gaussian Mixture Models 75

4.3.1 Linearization of the adaptively controlled closed-loop system 75

4.3.2 Propagating a GMM through the linearized system dynamics 79
4.4 Online Uncertainty Propagation using Gaussian Processes 84
4.4.1 Linear Gaussian Process models 85
4.4.2 Gaussian Process Models with nonlinear regressor functions 90
4.4.3 Online GP regression for uncertainty quantification 94
4.5 Indicator Function, Confidence Measure and Monitoring Output 100
4.5.1 Indicator Function for Requirement Violation 100
4.5.2 Confidence in the Monitor Prediction Quality 102
4.5.3 Monitoring Output 105
4.6 Implementation Aspects 106
4.7 Conclusion 108
5 Application in Numerical Simulation 110
5.1 Direct Model Reference Adaptive Control for wing-rock dynamics 110
5.1.1 Numerical Simulation of Direct MRAC for wing-rock dynamics 111
5.1.2 Numerical Simulation with known nonlinearity 116
5.1.3 Simulation with uncertainty 134
5.2 Short Period with predictor based MRAC 143
5.3 Conclusion 159
6 Application to Multirotor Systems 161
6.1 Rotational Dynamics of the Multirotor Systems 161
6.2 Quadcopter Application 164
6.2.1 Quadcopter: Control Architecture 164
6.2.2 Quadcopter: Application of the Monitoring Algorithm 166
6.2.3 Quadcopter: Experimental Setup and Results 169
6.3 Hexacopter Application 178
6.3.1 Hexacopter: Control Architecture 178
6.3.2 Hexacopter: Application of the Monitoring Algorithm 180
6.3.3 Hexacopter: Experimental Setup and Results 182
6.4 Conclusion 194
7 Conclusion and Outlook 195
7.1 Conclusion 195
7.2 Summary of Contributions 196
7.3 Future Work 198
7.3.1 Application specific future work 198
7.3.2 Possible future advancement of the monitoring approach 199
References I

Appendix XVI
A Mathematical Preliminaries XVI
B Basics of Probability Theory XX
C The Gaussian probability distribution XXXI
D Fundamentals of Lyapunov Stability Theory XXXVIII
E Filter Equations XL
F Naming Convention XLIV
G Coordinate Frames XLVI
H Auxiliary Information LII

List of Figures

Figure 2-1: Example of a univariate and multivariate Gaussian Mixture Model with 2 mixture components... 9

Figure 2-2: Approximation of a normal pdf by a GMM with 14 and 20 mixture components..10

Figure 2-3: General idea of Gaussian Processes with comparison between weight (top) and function (bottom) space view. Based on plots from [91]..17

Figure 2-4: Example of a GP used to model uncertain parameters in the $\boldsymbol{CL} - \boldsymbol{\alpha}$ curve........19

Figure 2-5: Example for GP regression with a linear model (based on [91]). 23

Figure 2-6: Three examples for kernel functions - 4th order polynomial kernel (left), squared exponential kernel (middle) and periodic kernel (right)..26

Figure 2-7: Evaluation of a GP at a single point and at an input distribution 27

Figure 2-8: $\boldsymbol{CL}$ over $\boldsymbol{\alpha}$ modeled by a GP and evaluated at an input distribution.....................30

Figure 2-9: Evaluation of a GP with SE kernels at the uncertain input 35

Figure 3-1: General control architecture with baseline and adaptive controller 38

Figure 3-2: Control Architecture with Direct Model Reference Adaptive Controller 41

Figure 3-3: Geometricl Interpretation of the stability proof in Theorem 3.1............................44

Figure 3-4: Control Architecture augmented by Predictor-Based Model Reference Adaptive Control 46

Figure 4-1: Proposed Monitoring Strategy 67

Figure 4-2: Areas where 67%, 90% and 99% of all state trajectories reside in 70

Figure 4-3: State trajectory distribution after 0.4s, 2.0s, 8.0s and 12.0s...............................70

Figure 4-4: Difference between linear and sample based propagation 72

Figure 4-5: Proposed strategy for state and parameter monitoring of Model Reference Adaptive Controllers 74

Figure 4-6: Comparison between linear, sampling based and GMM propagation 81

Figure 4-7: Comparison of linear (left) and GMM (right) propagation based on the Kullback-Leibler Divergence for varying parameters of the prior and different nonlinear functions 83

Figure 4-8: $\boldsymbol{CL}$ over $\boldsymbol{\alpha}$ modeled by a GP and evaluated at an a GMM distributed input.........89

Figure 4-9: Uncertainty Quantification for the application to MRAC 94

Figure 4-10: State evolution and uncertainty quantification at different points in time 99

Figure 4-11: Overlapping Coefficient between two normal distributions (left) and two arbitrary distributions (right)..104

Figure 5-1: Tracking capability of the adaptive controller in the presence of wing-rock motion 112

Figure 5-2: Evolution of the adaptive parameters in the presence of wing-rock motion.......113

Figure 5-3: Tracking performance of the adaptive controller under wing-rock motion for normally distributed initial conditions 114

Figure 5-4: Distribution of the roll angle $\boldsymbol{\phi}$ in the wing-rock example after 1s, 2.8s, 5.0s and 10.0s 115

Figure 5-5: Distribution of the roll rate $\boldsymbol{pB}$in the wing-rock example after 1s, 2.8s, 5.0s and 10.0s 115

Figure 5-6: Evolution of the predicted system states $\boldsymbol{\phi}$ and $\boldsymbol{pB}$ over $\boldsymbol{40s}$ in the wing-rock example 118

Figure 5-7: Predicted evolution of the roll angle $\boldsymbol{\phi}$ over 15 seconds leveraging a GMM 119

Figure 5-8: Predicted evolution of the roll rate $\boldsymbol{pB}$over 15 seconds leveraging a GMM 119

Figure 5-9: Predicted evolution of the adaptive weights over 15 seconds of the wing-rock simulation leveraging a GMM 120

Figure 5-10: Predicted evolution of the roll angle $\boldsymbol{\phi}$ over 15 seconds leveraging a normal distribution........122

Figure 5-11: Predicted evolution of the roll rate $\boldsymbol{pB}$over 15 seconds leveraging a normal distribution........122

Figure 5-12: Predicted pdf of the roll angle $\boldsymbol{\phi}$ in the wing-rock example after 0s, 1s, 2.8s and 5.0s 124

Figure 5-13: Predicted pdf of the roll rate $\boldsymbol{pB}$in the wing-rock example after 0s, 1s, 2.8s and 5.0s 124

Figure 5-14: Interpretation of over- and underestimating distributions 125

Figure 5-15: Predicted cdf of the roll angle $\boldsymbol{\phi}$ in the wing-rock example after 0s, 1s, 2.8s and 5.0s 127

Figure 5-16: Predicted pdf of the roll rate $\boldsymbol{pB}$ in the wing-rock example after 0s, 1s, 2.8s and 5.0s 127

Figure 5-17: GoC of the plant states with varying relaxation term for the wing-rock example 129

Figure 5-18: Probability of a roll angle requirement violation in the wing-rock example.......131

Figure 5-19: Probability of a roll rate requirement violation in the wing-rock example 131

Figure 5-20: Predicted Confidence based on OVL in the wing rock example........132

Figure 5-21: Modified predicted probability of a requirement violation in the wing rock example 133

Figure 5-22: Wing rock uncertainty (left) and its approximation after 40 seconds into the simulation (right)........135

Figure 5-23: Contour plot of the uncertainty approximation error in the wing rock simulation example after $\mathbf{2.5}$, $\mathbf{10}$, $\mathbf{20}$ and $\mathbf{40}$ seconds 135

Figure 5-24: Contour plot of the GP standard deviation in the wing rock simulation example after $\mathbf{2.5}$, $\mathbf{10}$, $\mathbf{20}$ and $\mathbf{40}$ seconds........136

Figure 5-25: Predicted roll angle $\boldsymbol{\phi}$ over 15 seconds with GP regression based uncertainty estimation........137

Figure 5-26: Predicted roll rate $\boldsymbol{pB}$ over 15 seconds with GP regression based uncertainty estimation........137

Figure 5-27: GoC of the plant states with relaxation term for the wing-rock example with GP regression ... 139

Figure 5-28: Probability of a roll angle requirement violation in the wing-rock example with GP regression ... 141

Figure 5-29: Probability of a roll rate requirement violation in the wing-rock example with GP regression ... 141

Figure 5-30: Predicted Confidence based on OVL in the wing rock example with GP regression based uncertainty approximation ... 142

Figure 5-31: Modified predicted probability of a requirement violation in the wing rock example with GP regression based uncertainty approximation ... 142

Figure 5-32: Tracking performance of PMRAC for the short period example with normally distributed initial conditions ... 147

Figure 5-33: Distribution of the integral error $\boldsymbol{eI}$in the short period example after 0.2s, 1s, 5.0s and 7.5s ... 148

Figure 5-34: Distribution of the angle of attach $\boldsymbol{\alpha K}$in the short period example after 0.2s, 1s, 5.0s and 7.5s ... 148

Figure 5-35: Distribution of the pitch rate $\boldsymbol{qK}$in the short period example after 0.2s, 1s, 5.0s and 7.5s ... 149

Figure 5-36: Uncertainty (left) and its approximation (right) in the short period simulation example ... 150

Figure 5-37: Relative UQ error (left) and the approximated posterior standard deviation (right) for the GPR in the short period simulation example ... 150

Figure 5-38: Integral error $\boldsymbol{eI}$, angle of attach $\boldsymbol{\alpha K}$ and pitch rate $\boldsymbol{qK}$ predicted over 30 seconds in the short period simulation example ... 151

Figure 5-39: Predicted cdf of the integral error $\boldsymbol{eI}$in the short period example after 0.2s, 1s, 5s and 7.5s ... 153

Figure 5-40: Predicted cdf of the angle of attack $\boldsymbol{\alpha K}$in the short period example after 0.2s, 1s, 5s and 7.5s ... 153

Figure 5-41: Predicted cdf of the pitch rate $\boldsymbol{qK}$ in the short period example after 0.2s, 1s, 5s and 7.5s ... 154

Figure 5-42: GoC of the plant states with relaxation term for the short period example ... 155

Figure 5-43: Probability of an angle of attack requirement violation in the short period example ... 157

Figure 5-44: Probability of a pitch rate requirement violation in the short period example ... 157

Figure 5-45: Predicted Confidence based on OVL in the short period example ... 158

Figure 5-46: Modified predicted probability of a requirement violation in the short period example ... 158

Figure 6-1: Quadcopter Hummingbird (left) and Hexacopter Firefly (right) as used at FSD 161

Figure 6-2: Forces and rotational rate of a multirotor propulsion system exemplified on a quadcopter ... 162

Figure 6-3: Tracking performance of the adaptive controller in the quadcopter experiment 172

Figure 6-4: Evolution of the adaptive weights in the quadcopter experiment.......................173

Figure 6-5: Uncertainty estimation result in the quadcopter experiment for a vanishing yaw rate..173

Figure 6-6: Selected predictions for the roll rate in the quadcopter experiment...................174

Figure 6-7: Predicted roll, pitch and yaw rate in the quadcopter experiment.......................175

Figure 6-8: Probability of a angular rate requirement violation in the quadcopter experiment 176

Figure 6-9: OVL modified predicted probability of an angular rate requirement violation in the quadcopter example...177

Figure 6-10: Tracking performance of the adaptive controller in the hexacopter experiment 185

Figure 6-11: Evolution of the adaptive weights in the hexacopter experiment.....................186

Figure 6-12: Uncertainty estimation result in the hexacopter experiment for a vanishing yaw rate..186

Figure 6-13: Selected predictions for the roll rate in the hexacopter experiment with unknown reference command evolution 187

Figure 6-14: Predicted roll, pitch and yaw rate in the hexacopter experiment with unknown reference command evolution 189

Figure 6-15: Selected predictions for the roll rate in the hexacopter experiment with known reference command evolution 190

Figure 6-16: Predicted roll, pitch and yaw rate in the hexacopter experiment with known reference command evolution 191

Figure 6-17: Probability of a angular rate requirement violation in the hexacopter experiment 192

Figure 6-18: OVL modified predicted probability of an angular rate requirement violation in the hexacopter example 193

Figure 7-1: Conflicting goals regarding intuitive understanding and information contant.....198

Figure 7-2: Value-at-risk based reference command shaping..199

Figure C-1: Pdf of the Univariate and Multivariate normal distribution.............................. XXXI

Figure E-1: Timeline of an Optimal Fixed Point Smoother XLII

Figure G-1: ECI Frame (from [93])..XLVI

Figure G-2: ECEF Frame (from [93]) XLVII

Figure G-3: NED-Frame (from [93])..XLVIII

Figure G-4: Body-fixed frame (from [93]) XLIX

Figure G-5: Kinematic Frame (from [93]) L

Figure G-6: Aerodynamic Frame (from [93])...LI

Figure H-1: Adjusted scale for the Kullback-Leibler Divergence based comparison plots in Example 4-4 LIII

Figure H-2: Overlapping Coefficient of different nonlinear functions LV

Figure H-3: Evolution of the adaptive parameters under wing-rock motion for normally distributed initial conditions LVI

Figure H-4: Predicted evolution of the adaptive weights over 40 seconds of the wing-rock simulation leveraging a GMM LVII

Figure H-5: Predicted evolution of the adaptive weights over 40 seconds of the wing-rock simulation leveraging a normal distribution only........ LVIII

Figure H-6: Predicted evolution of the adaptive weights over 15 seconds of the wing-rock simulation leveraging a normal distribution only........ LIX

Figure H-7: Predicted roll angle $\boldsymbol{\phi}$ over 40 seconds with GP regression based uncertainty estimation LX

Figure H-8: Predicted roll rate $\boldsymbol{pB}$ over 40 seconds with GP regression based uncertainty estimation LX

Figure H-9: Predicted evolution of the adaptive weights over 40 seconds of the wing-rock simulation leveraging a GMM and GP regression based uncertainty estimation LXI

Figure H-10: Predicted evolution of the adaptive weights over 30 seconds in the short period simulation example leveraging a GMM LXII

Figure H-11: Predicted evolution of the predictor states over 30 seconds in the short period simulation example leveraging a GMM LXIII

List of Tables

Table 3-1: Examples for uncertainties $\Delta x P, u$ satisfying Definition 3.1 37

Table 3-2: Update Law Modifications; based on [223, p. 68] and [17] 54

Table 4-1: Strategies for the parameter selection of the indicator function 101

Table 4-2: The proposed monitoring algorithm exemplarily depicted for direct MRAC 106

Table 5-1: List of parameters used in the MRAC simulation example 111

Table 5-2: List of parameters used for the GMM forecast in the wing-rock example 117

Table 5-3: Average mean error, $\mathcal{L}2$ −like norm and maximum mean error for the prediction of the mean trajectory in the wing-rock simulation example comparing a GMM and a normal distribution forecast 121

Table 5-4: GoC in the wing-rock simulation example 130

Table 5-5: List of parameters used for the indicator function and monitoring output in the wing rock simulation example 132

Table 5-6: List of parameters used GP regression in the wing rock simulation example 134

Table 5-7: Average mean error, $\mathcal{L}2$ −like norm and maximum mean error for the prediction of the mean trajectory in the wing-rock simulation example with GP regression based uncertainty estimation..138

Table 5-8: GoC in the wing-rock simulation example with GP regression based uncertainty approximation..139

Table 5-9: List of parameters used in the PMRAC simulation example 145

Table 5-10: List of monitoring parameters used in the PMRAC simulation example 146

Table 5-11: Average mean error, $\mathcal{L}2$ −like norm and maximum mean error for the prediction of the mean trajectory in the pitch rate simulation example 152

Table 5-12: Numerical values for the GoC with relaxation term for the short period simulation example 156

Table 6-1: List of controller parameters used in the quadcopter experiment 169

Table 6-2: List of monitoring parameters used in the quadcopter experiment 170

Table 6-3: List of controller parameters used in the hexacopter experiment 182

Table 6-4: List of monitoring parameters used in the hexacopter experiment 183

Table H-1: Simulation Parameters for Example 4-1 LII

Table H-2: Simulation Parameters for Example 4-6 LIV

List of Examples

Example 2-1: Possible View of a Gaussian Process 17

Example 2-2: GP used to model uncertainty in aerodynamic data.......................................18

Example 2-3: Evaluation of a GP prior model with linear kernels at an uncertain input.........30

Example 2-4: Evaluation of a GP formed by SE kernels at an uncertain input 35

Example 4-1: Trajectory distribution for a nonlinear simulation 69

Example 4-2: Propagation of a normally distributed random variable through a sine function by sampling and linear approximation 72

Example 4-3: Propagation of a normally distributed random variable through a sine function by sampling and linear as well as GMM approximation 81

Example 4-4: Propagation of a normally distributed random variable through a variety of nonlinear functions with varying parameters of the prior distribution.....................................82

Example 4-5: Evaluation of a GP prior model with linear kernels at a GMM distributed input89

Example 4-6: GP Regression example applied to MRAC 97

List of Acronyms

Acronym	Description
AME	Average Mean Error
EKF	Extended Kalman Filter
GoC	Grade of Circumvallation
GMM	Gaussian mixture model
GP	Gaussian Process
MPC	Model Predictive Control
MRAC	Model Reference Adaptive Control
OFPS	Optimal Fixed Point Smoother
OVL	Overlapping Coefficient
PCE	Polynomial Chaos Expansion
PLF	Piecewise linear function
RBF	Radial basis function
SE	Squared exponential
UAS	Unmanned aerial system
V&V	Verification and Validation
cdf	Cumulative distribution function
i.i.d.	Independent and identically distributed
mgf	Moment Generating Function
pdf	Probability Density Function
rv	Random Variable

List of Symbols

Latin Letters	
Symbol	**Description**
$\boldsymbol{A}_{(\cdot)}$	System matrix
$\boldsymbol{B}_{(\cdot)}$	Input matrix
$\boldsymbol{B}_{X_H Y_H}$	Auxiliary matrix
C_L	Aerodynamic lift coefficient
$C_{Mu}, C_{M\omega}$	Parameter matrix for multirotor dynamics
$\mathbb{C}(x, x')$	Covariance of x and x'
D_{KL}	Kullback-Leibler Divergence
$E_{(\cdot)}$	Error value
$\mathbb{E}(x)$	Expectation of x
$F_{(\cdot)}$	Cummulative distribution function
$F_{i,z}$	Force
$\boldsymbol{G}(x)$	Nonlinear state dependent system input function
$I_{(\cdot)}$	Element of the moment of inertia matrix
$\boldsymbol{I}, \boldsymbol{I}_{3\times3}$	Identity matrix, Identity matrix of dimension 3×3
$\boldsymbol{I}_{BB}$	Moment of inertia matrix
$\boldsymbol{K}_{(\cdot)}$	Controller gain matrix
$\boldsymbol{L}$	Auxiliary matrix
$M_{i,z}$	Moment
$M_x(s)$	Moment generating function
$\mathrm{M}(\cdot)$	Builds a matrix of appropriate dimension of the input $(\cdot)$
M_α, M_q, M_η	Partial derivatives of the pitching moment
$\vec{\boldsymbol{M}}$	Vector of moments
$N(\mu, \sigma^2)$	Normal distribution with mean μ and variance σ^2
$O^2(\cdot)$	Higher order terms
P_ζ, P_{mod}	Probability of a requirement violation, Modified probability
$\boldsymbol{P}$	Positive definite solution to the Lyapunov equation
$\boldsymbol{Q}$	Positive definite matrix
V	Trimmed airspeed
$V(\cdot)$	Continuously differentiable positive definite function
$\mathbb{V}(x)$	Variance of x
$\boldsymbol{X}_{(\cdot)}, \boldsymbol{Y}_{(\cdot)}, \boldsymbol{Z}_{(\cdot)}$	Vector or matrix of data points or random variables
Z_α, Z_q, Z_η	Partial derivatives of the aerodynamic vertical force
c_i	Auxiliary coefficients
$\boldsymbol{e}$	Tracking error
$f(x)$	Function of x

g_{ind}	Test function
h_{ind}	Test function
i	Index
$k(x,x')$	Kernel function
k_M, k_T	Torque and thrust parameters of a rotor
l	Length of a multirotor arm
$l_i, \boldsymbol{l}$	Auxiliary value / vector
m, n	Auxiliary scalar
$m(x)$	Mean function of x
$p_x(x)$	Probability density function of x
p, p_i	Number of data points, i –th data point in a history stack
p_B	Body-fixed Roll rate
p_p	Percentile
q_B, q_K	Body-fixed Pitch rate, Kinematic Pitch rate
$\boldsymbol{r}$	Exogenous reference command
r_B	Body-fixed yaw rate
$\vec{\boldsymbol{r}}$	Position vector
t	Time
$\boldsymbol{u}$	Control input vector
$vec(\boldsymbol{M})$	Vectorizes the matrix $\boldsymbol{M}$
$\boldsymbol{w}$	Parameters of a Gaussian Process
$\bar{\boldsymbol{w}}$	Mean of $\boldsymbol{w}$
$x, \boldsymbol{x}$	State, State vector
$\hat{\boldsymbol{x}}$	State vector of the state predictor
$y(x), \boldsymbol{y}(x)$	Output, Output vector

Greek Letters	
Symbol	**Description**
$\boldsymbol{\Gamma}$	Learning rates
$\Delta(\cdot)$	Modelling uncertainty
Δx	Change in x
$\boldsymbol{\Theta}, \boldsymbol{\Theta}^*, \widetilde{\boldsymbol{\Theta}}$	Adaptive parameters, optimal parameters, parameter error
$\boldsymbol{\Lambda}$	Matrix containing the bandwith of kernel functions
$\boldsymbol{\Lambda}_u$	Parameter matrix of the multirotor inputs
Λ_i	Element of $\boldsymbol{\Lambda}$
Σ	Variance
$\Sigma(x, x')$	Covariance function
$\boldsymbol{\Phi}_{(\cdot)}$	Vector of nonlinear basis functions
Ω	Compact set
α, α_K	Angle of attack, Kinematic Angle of attack
$\alpha_i, \boldsymbol{\alpha}$	i –th weight of a Gaussian Mixture Model, Vector of mixture weights
α_{ind}	Parameter in the monitor for the prediction horizon
β_{ind}	Parameter in the monitor for the prediction horizon
γ	Indicator
δ_{ind}	Indicator threshold
δ_ζ	Probability limit in the monitoring algorithm
$\boldsymbol{\varepsilon}$	Measurement noise
$\epsilon_{(\cdot)}$	Positive numerical value
ζ	Damping of a linear second order system
ζ_C	Aileron command
$\boldsymbol{\eta}(\cdot)$	Representation error
$\eta_{(\cdot)}$	Elevator command
λ_i	Level of motor degradation
$\lambda_{min/max}(\cdot)$	Minimum/Maximum eigenvalue of a matrix
μ	Mean
ν	Pseudo control input
σ	σ –Modification
$\sigma_{(\cdot)}$	Standard Deviation
ϕ	Roll angle
$\phi(x)$	Activation function / regressor function / basis functions
$\varphi(x)$	Nonlinear basis functions of the adaptive controller
ω_i	Rotational rate of the i –th propeller
$\vec{\boldsymbol{\omega}}$	Vector of rotational rates

Indices	
Symbol	**Description**
0	Initial value
AM	Average mean
B	Body-fixed
C	Control
GMM	Gaussian Mixture Model
H	History Stack
I	Integrated
K	Kinematic
L	Lift
M	Moment
T	Thrust
$\mathcal{L}_2$	$\mathcal{L}_2$ −norm
$\mathcal{L}_\infty$	$\mathcal{L}_\infty$ −norm
P	Plant
ad	Adaptive
app	Application
bl	Baseline
cmd	Command
d	Disturbance
e	Error
end	Final value
i	Running index
$\cdot\|i$	With respect to the i −th mixture element in a Gaussian mixture model
ind	Indicator
k	Discreite time step
l	Reference point of a Taylor series expansion
lin	Result from a linearization
mon	Monitoring output
n	Noise
$post$	Posterior
$pred$	Prediction
prd	Predictor
req	Requirement
$req, high$	Upper State Requirement
$req, high$	Lower State Requirement
r	Reference input
rm	Reference model
s	System states

w	Relative to the parameter vector $\boldsymbol{w}$
x	Relative to the state $\boldsymbol{x}$
Δ	Relative to the modelling uncertainty Δ
Θ	Relative to the adaptive parameters $\boldsymbol{\Theta}$
η	Elevator
φ	Relative to the basis function vector $\boldsymbol{\varphi}$
$*$	Test input

Superscripts	
$+$	Moore-Penrose Pseudo-inverse
$*$	Optimal adaptive parameters

Additional notation	
$\bar{\cdot}$	Discretized (for all except $\bar{\boldsymbol{w}}$)
$\hat{\cdot}$	Estimated value
$\tilde{\cdot}$	Parameter error
$\vec{\cdot}$	Vector

Furthermore, the naming convention of the Institute of Flight System Dynamics at TU Munich applies. The parts, which are important for this thesis are summarized in Appendix F.

1 Introduction

1.1 General Introduction

Unmanned aerial systems (UAS) are increasingly utilized in commercial airspace. Their application spans a broad range from surveillance and inspection to delivery and hobby photography. Engineers envision future applications to reduce the number of operators per UAS and to tackle tasks in cooperative swarms ([66]). This requires a drastic increase in the level of UAS autonomy. Meanwhile, in order to be attractive for commercial applications, UAS shall undergo quick development cycles, ensure cost effectiveness and work reliably in changing environments. In this context, developing flight controllers for each single vehicle type or for each single configuration presents a current undesired necessity. The sustained trend to higher levels of autonomy, the demanded versatility and the cost effectiveness pose a challenge for future UAS control systems.

Commercially available UAS are equipped with flight control systems based on linear control theory. The latter leverages a model of the UAS in order to derive fixed controller gains, which yield desired robustness and performance characteristics. The model is constructed by first fracturing the operating domain and approximating the highly nonlinear system dynamics by linear models. In a second step the model parameters are determined for each such fraction. Two main factors render linear control theory unattractive for UAS development and increasing levels of autonomy. First, for large UAS the parameters of the derived models are mostly determined by extensive experiments, such as wind tunnel tests. As a consequence, the development requires a lot of time, money and effort. Human life is not put under direct risk during UAS flight tests. Hence, instead of wind tunnel tests, flight tests present an often cheaper and quicker alternative to generate the required model parameter sets for certain flight conditions ([191], [36]). Still, this approach requires a lot of effort in order to build a database for a huge variety of flight conditions. Furthermore, a reliable knowledgebase needs to be available a-priori, in order to even perform flight tests. The second main drawback of linear flight controllers is that UAS are often subject to configurable and structural modifications. This results in changes of the model parameters such as drag and lift coefficients. As the modelling uncertainty leads to the deterioration or even nullification of derived performance and robustness guarantees, an already developed database is rendered unreliable.

Nonlinear control techniques (see i.a. [118], [155]) eliminate many of the undesirable features of linear controllers. They operate through large sections of the operating domain, do not necessarily require an exact model of the system and are able to compensate structural changes of the UAS. These features allow them to be equipped to a huge variety of systems with only minor adjustment. Thus, they promise the desired versatility within a reduction of the development time and effort. A special kind of nonlinear control techniques that exhibits the aforementioned capabilities is the class of adaptive controllers.

Model Reference Adaptive Control (MRAC) is a well-studied technique from the family of adaptive controllers (see i.a. [155], [8], [204]). MRAC aims at making a system behave like an a-priori chosen dynamical model. This is achieved by approximating the modeling uncertainty using a weighted combination of known basis functions. The underlying assumption is that there exists an optimal set of gains that best captures the modeling uncertainty for a particular structure of the adaptive element. If the gains do converge to their ideal values and the uncertainty is entirely canceled, the closed loop system matches the a-priori chosen model and recovers its properties. Most adaptive control algorithms try to achieve this goal by adjusting the parameters based on the minimization of a quadratic cost. A Lyapunov based stability proof ensures boundedness of all system signals. Thus, adaptive control techniques promise versatile application to a huge variety of systems only requiring minor adjustments. Still, the desired dynamical behavior needs to lie within the physical capabilities of the controlled UAS.

MRAC has repeatedly shown good performance in various applications, in particular when the system dynamics are uncertain or during environmental and structural changes ([114], [41], [195]). However, several shortcomings prevent a successful application in commercial aviation. This is mainly due to missing certification guidelines and procedures for nonlinear and, in particular, adaptive flight control techniques.

Airborne systems are extensively regulated by governments and need to be certified before deployment. From an abstract point of view, the certification process is intended to generate trust in a successful application of the flight control algorithm and the UAS. Ways to create this trust are summarized amongst others in guiding documents (see e.g. [176]). The certification process was designed with linear control systems in mind. For instance, analytically derived performance and robustness metrics, such as gain and phase margins, represent a crucial tool in the certification process. Unfortunately, these metrics are not applicable to nonlinear systems and universal metrics do not yet exist for adaptive controllers. Apart from metrics, researchers identified various other gaps, which currently prevent the successful certification of adaptive flight control algorithms (see i.a. [105], [169], [220]). This includes amongst others the need for a concise software verification plan, a consistent requirement formulation and appropriate verification tools.

A central challenge within the introduction of a certification plan for adaptive flight controllers involves the *Verification and Validation* (V&V) processes. In the context of V&V, the analysis of fixed-gain linear control systems is performed prior to deployment. Hence, guarantees and the expected behavior of the closed-loop system are known before the system is even used. In contrast, adaptive systems adjust their parameters online. As a result they are able to compensate modeling uncertainties and unforeseen configurational or environmental changes. If well established V&V procedures for linear controllers were applied to adaptive controllers, requirements would have to be tested against an infinite amount of parameter combinations ([220, p. 14], [49, p. 13]).

Recent updates to the certification specifications ([47b]) support novel design solutions in order to show the safe operation of UAS. The evolving nature of adaptive controllers sparked an emerging consensus between various authors to move the verification from an a-priori analysis to a verification during the runtime of the system (see i.a. [105], [169], [170], [177]). An integral part of run time verification is the establishment of an online monitoring tool. These algorithms are envisioned to provide information about the adaptation performance and impending undesired behavior of the adaptive controller and the closed loop system.

Several approaches to online monitoring of adaptive control algorithms exist (see i.a. ([129], [222], [189], [186]). A frequent feature is that current monitoring techniques only focus on the assessment of the learning capabilities of the adaptive element. The underlying hypothesis is that the successful operation of the controlled system is ensured by solely monitoring the learning capability of the adaptive element. Thereby, the mentioned approaches intrinsically decouple the controller from the plant and thus neglect three key aspects. First, requirements are formulated in the context of plant states. Hence, even if the adaptive controller operates as desired, plant states might exceed structural and operational limits. Secondly, the coupling between the nonlinear control algorithm and the system dynamics is neglected. As a result, effects resulting from learning transients can lead to harsh and undesired state responses. Finally, due to residual errors, signal injection may cause unacceptable transient states even after the adaptive parameters converged. In this context, a deteriorated learning performance might even be acceptable if it improves the response characteristics of the state dynamics.

This thesis introduces an online monitoring concept, which aims at predicting the evolution of the state and parameter trajectories. The underlying goal is to detect imminent state requirement violation and undesired behavior of the closed loop system. The monitor achieves this by recursive propagation of the states and adaptive parameters through a model of the plant and parameter dynamics. In the absence of uncertainty, this approach results in the exact trajectory the UAS will take. However, systems are naturally subject to stochastic uncertainties originating from i.a. noisy state measurements, distributed system parameters or external disturbances. Hence, the prediction has to be carried out in the context of stochastic uncertainty propagation. Thereby, three key challenges arise. First, parts of the closed loop dynamics might not be known. Their negligence leads to heavily deteriorated prediction performance. Secondly, the closed loop dynamics are highly nonlinear. Even if the plant was linear, the adaptive component still adds a nonlinear feedback to the control signal. As a result, propagated distributions are seldom of the Gaussian type. Furthermore, approaches that rely on (approximately) linear systems, such as the prediction step in the Kalman filter ([106]), again lead to heavily deteriorated prediction performance. Lastly, the algorithm has to run on UAS hardware. UAS exhibit fast dynamics, while the online available computational resources are usually low. As a result, techniques that use sampling, such as the particle filter ([163]), are not suited to carry out the prediction. Instead, a monitoring algorithm based on analytic predictions is aspired.

To overcome the first challenge, this thesis employs *Gaussian Processes* (GP, see i.a. [173], [21], [152]) in order to represent the modeling uncertainty. GPs are probabilistic models that can be completely constructed using online gathered data. A means to achieve this is GP regression, which leverages Bayesian inference in order to fit a GP model to a set of data. This way unknown parts of the system dynamics can be quantified during the runtime of the system. To solve the second and third challenge, the proposed monitoring algorithm employs *Gaussian Mixture Models* (GMM, [21]) in order to model the state and parameter distribution. While each component of a GMM represents a Gaussian, their combination is able to represent a huge variety of distributions, including multimodal and heavily skewed distributions. During the prediction, the nonlinear dynamics are approximated by linear models at the center of each mixture component. The underlying assumption is that the covariance of each mixture component is small enough to warrant the linearization. The prediction is performed by propagating each mixture component through its respective approximated linear system model as well as the inferred GP. Repeating this process over the prediction horizon yields the evolution of the state and parameter distribution. Analyzing the cumulative distribution function of the GMM at various points in the prediction horizon allows for the assessment of the probability of imminent state requirement violations.

A detailed review of the aspects discussed above can be found in the respective chapters. The following section details the contributions over the state of the art.

1.2 Contributions

This thesis aims at the design and the implementation of an online monitoring algorithm for adaptive control algorithms. The following five points highlight the contributions of this thesis over the state of the art.

Analytic long term prediction for monitoring of adaptively controlled systems

A central contribution of this thesis is the proposal of a stochastic monitoring system for adaptively controlled systems. The monitor relies on analytic long term forecasts of the system states in order to assess potential undesired behavior in the near future. Stochastic long term predictions are employed in a variety of applications such as weather forecasts, safety assessments or stock development. However, the employed techniques predominantly rely on linear systems subject to uncertainties governed by a Gaussian probability distribution, or sampling techniques such as Monte Carlo. Recently analytic long term predictions of highly nonlinear and non-normally distributed dynamical systems came into focus (see e.g. [85], [171], [71]).

The contribution of this thesis is the application of analytic long term prediction to the monitoring of adaptively controlled and highly nonlinear systems. To the best of the authors' knowledge, this approach has neither been employed for flight control applications nor adaptive controllers. The approach leverages as much prior information as possible. The known part of the system dynamics is used in order to propagate a GMM, which describes the system state distribution. The uncertain parts of the system dynamics are represented using GP models. Two cases are considered. First, in the case of parameter uncertainties with known parameter distribution, a GP prior model is utilized. If the uncertainty is completely unknown, GP regression is used in order to generate a model from online gathered data. In this, the proposed approach differs from existing applications, which either only consider the known system dynamics, or use GP regression to form a model of the complete dynamical system.

Combined state and control prediction for system monitoring

A central contribution of the proposed approach is the combined evaluation of controller and plant states. Existing monitoring approaches predominantly aim at assessing the learning capabilities of the adaptive element only (see amongst others [129], [133], [134], [132], [188], [222], [68], [186], [186], [82], [188]). The underlying idea is that if the adaptive element successfully learns the modeling uncertainty, its impact on the plant dynamics is mitigated and the plant dynamics reduce to a predefined system. Since the latter is known, the closed loop system assumes well defined properties. Hence, the authors argue that if the learning capability of the adaptive element is monitored, the successful operation of the controlled system can be assessed.

The mentioned approaches intrinsically decouple the controller from the plant. In particular, the closed loop system is seen to be functioning as intended as long as the learning capability of the adaptive element is deemed ‘good enough’. However, this reasoning neglects several aspects. First, requirements are mostly formulated with system states in mind. In the end, mainly the states of the plant define whether or not it exceeds structural or operational limits. Even if the adaptive element functions as intended, the system might exceed those limits due to small residual errors. Secondly, the effect of the adaptive element on the plant dynamics during transient phases is ignored. In particular, adaptive elements with gradient based update laws often suffer from harsh learning transients. As a consequence, also state trajectories exhibit huge overshoots, which can cause the plant to exceed predefined limits. Finally, signal injection may cause unacceptable transient states even after learning convergence. This occurs if the uncertainty approximation is not uniform and residual approximation errors remain. Even though the learning is assessed to be as desired, the plant states might again violate predefined requirements. Furthermore, a deteriorated learning performance might even be acceptable if it improves the properties of the state dynamics.

The proposed monitoring technique significantly differs from the mentioned techniques in its strategy. Thereby, the approach models the dependencies between the plant and the controller. The combined prediction of plant and controller state trajectories enables the assessment of the adaptive control impact on the system dynamics in the future. As a consequence, the monitor is able to consider transients in the controller and plant state. Furthermore, the effects of signal injection can be analyzed.

Contribution to the closure of gaps in the certification process of adaptive flight control algorithms

The proposed monitoring algorithm directly contributes to the closure of gaps in the certification process of adaptive control algorithms. The certification process of flight control algorithms is specifically tailored for linear controllers. Exemplarily, requirements are often formulated in the frequency domain and robustness metrics, such as the gain and phase margin, are only calculable for linear system (see e.g. [181], [180]). Thereby, the V&V process plays a central role in the certification of adaptive flight control algorithms. A variety of authors (see amongst others [105], [169], [170], [205], [220], [187], [177], [178]) identified technical and formal drawbacks that prevent the direct adaptation of the V&V process for linear systems to adaptive controllers. Overcoming these V&V challenges is seen as one of the major challenges for future commercial adaptive control applications.

Recent updates to the certification specifications ([47b]) support novel design solutions in order to show safe operation of UAS. An emerging consensus between the mentioned authors classifies runtime verification as a key component in the V&V process for adaptive control algorithms. The underlying reasoning is that the desired and inherent property of the adaptive element to react to unforeseen situations cannot be verified a-priori. Performing verification for a nonlinear system requires, amongst other things, the evaluation of all possible situations and parameter combinations. Since such systems assume an infinite number of testing combinations ([220, p. 14], [49, p. 13]), a-priori verification is not feasible or the verification domain has to be significantly restricted. In contrast, the convexity of linear systems allows for the assessment of whole subsets of the verification domain by e.g. robustness metrics such as the gain and phase margin.

Following the arguments of, amongst others, Jacklin ([105]), the proposed monitoring algorithm presents a contribution to the closure of gaps in the certification process of adaptive flight control algorithms. Until now, most monitoring algorithms focus on the learning quality of the adaptive element and neglect the effect of the controller on the plant e.g. in transient phases. As a consequence, requirements on the plant states are not directly relatable to the results of the monitoring algorithm. In contrast, the proposed monitor employs a stochastic model of the dynamic system in order to predict the evolution of the plant and controller state trajectories. This allows the monitor to assess undesired behavior or immanent requirement violations and perform suitable countermeasures. Therefore, the proposed approach is able to directly relate the performance of the adaptive control algorithm to a-priory formulated state requirements and thus test for their adherence online.

Introduction of a Confidence Measure to the prediction horizon

A further contribution of this thesis involves the introduction of a confidence measure in order to express the certainty of the algorithm in its own prediction. Approaches that use predictions rely on assumptions and approximations, inevitably introducing errors. The long term prediction of the system model also leads to the propagation of these errors. As a consequence, the predicted state trajectories are rendered unreliable with increasing prediction horizon. However, how trustworthy the prediction is cannot be assessed a-priori. It rather depends on the current approximation quality of the monitoring algorithm and the covariance of the GMM components.

Here, a confidence measure is introduced based on the approximation errors made during the prediction. It consists of two parts. The first part considers propagation errors due to the approximation of nonlinear system dynamics. In [206], Terejanu discusses errors originating from the propagation of a GMM through a nonlinear system. The second part considers errors due to the propagation of a GMM through a GP prior model. Therefore, the proposed confidence measure extends the technique in [206] by also considering propagation errors through a GP prior model.

Novel use of Gaussian Process regression in adaptive flight control

A further contribution of the proposed technique is the use of GP regression to construct a model for long term prediction in adaptive flight control applications. GP regression has been applied to a huge variety of problems in order to model an uncertain system based on gathered data. Only recently, GP regression caught attention in the field of adaptive flight control. In [38], [37] Chowdhary and Kingravi use GP regression to construct a GP model of the uncertainty using online gathered data only. Instead of employing a gradient descent based update law, as is common in MRAC frameworks, the authors use the mean of the predictive posterior distribution as the adaptive control signal.

The proposed monitoring approach uses a similar strategy to construct a GP model if the modelling uncertainty is entirely unknown. Instead of employing the mean of the predictive posterior distribution as the control signal, the entire GP model is used for prediction instead. A similar approach has been utilized in various applications such as predictions in the field of automotive dynamics (see e.g. [85]). To the best of the authors' knowledge, an online generated GP model for multistep ahead prediction has neither been used in flight control nor adaptive control approaches yet.

1.3 Outline of the thesis

The outline of this thesis is as follows. Chapter 2 presents the basics of GMMs and GPs, which in turn represent the core tools used in the proposed online monitoring concept. The former are used in order to approximate the state and parameter distribution and propagate it through the nonlinear dynamics of the system. The latter are used in order to construct a probabilistic model of the modelling uncertainty. Chapter 3 formalizes the system for which the runtime monitor will be designed. Afterwards, this chapter derives MRAC and presents the direct and predictor based approach. This chapter closes with an overview of the certification challenge faced by adaptive flight control algorithms. Chapter 4 represents the core of this thesis as it introduces the concept of the runtime monitor. Following a literature review on existing monitoring approaches for adaptive controllers, it discusses the steps of the proposed runtime monitoring concept. Afterwards it details how GMMs and GPs are used in order to propagate a distribution through nonlinear system dynamics. This chapter also contains a discussion about possible propagation errors and implementation aspects. Chapter 5 applies the runtime monitor to simple models in numerical simulation in order to foster understanding and highlight key aspects of the proposed concept. Chapter 6 applies the runtime monitor to the attitude loop of multirotor systems. Thereby, this chapter summarises the system dynamics as well as the employed adaptive control algorithm. It closes with the presentation of the experimental results. Finally, chapter 7 concludes this thesis. Additional and auxiliary information can be found in the Appendices.

2 Basics of Gaussian Mixture Models and Gaussian Processes

This section deals with the basics behind Gaussian Mixture Models and Gaussian Processes. Both concepts will play an integral role in the monitoring approach of section 4. The fundamentals of probability theory are assumed to be known. However, the most important aspects are summarized in Appendices B and C.

2.1 Gaussian Mixture Models

There exist phenomena, which are not well modelled by commonly used distributions such as the normal or Cauchy distribution. This includes i.e. phenomena subject to a multi-modal probability distribution. In this case, mixture models often present a way of modelling them nonetheless. Mixture models are weighted linear combinations of specific probability distributions. If the mixture model in turn satisfies the requirements of a probability distribution, it can be used as a probabilistic model. Examples where authors used mixture models include the 'Old Faithful data set' ([9]) or the representation of the speech spectrum in speech recognition ([201]). A famous representative of such models is the *Gaussian Mixture Model* (GMM). The components of the latter are all normal distributions. Definition 2.1 formally introduces GMMs.

Definition 2.1 – Gaussian Mixture Model ([21, p. 111])

A *Gaussian Mixture Model* or *Mixture of Gaussians* is a superposition of m weighted Gaussian densities or *Mixture Components* of the form

$$p(x) = \sum_{i=1}^{m} \alpha_i N(\mu_i, \Sigma_i), \tag{2-1}$$

where the parameters α_i satisfy

$$\alpha_i \geq 0 \; \forall i$$
$$\sum_{i}^{m} \alpha_i = 1. \tag{2-2}$$

The conditions in (2-2) imply that a GMM never assumes negative values and integrates to one. Hence, it represents a valid probability distribution. Figure 2-1 shows an example of an univariate and a multivariate GMM, where each consists of 2 mixture components. By adjusting the weight, the variance and the mean of each mixture, a sufficient number of components approximate almost any continuous distribution to arbitrary accuracy ([21, p. 111]). Section 2.1.1 deals with determining the GMM parameters for fitting it to a probability distribution. Section 2.1.2 summarizes important mathematical properties of GMMs, which will be needed throughout this thesis.

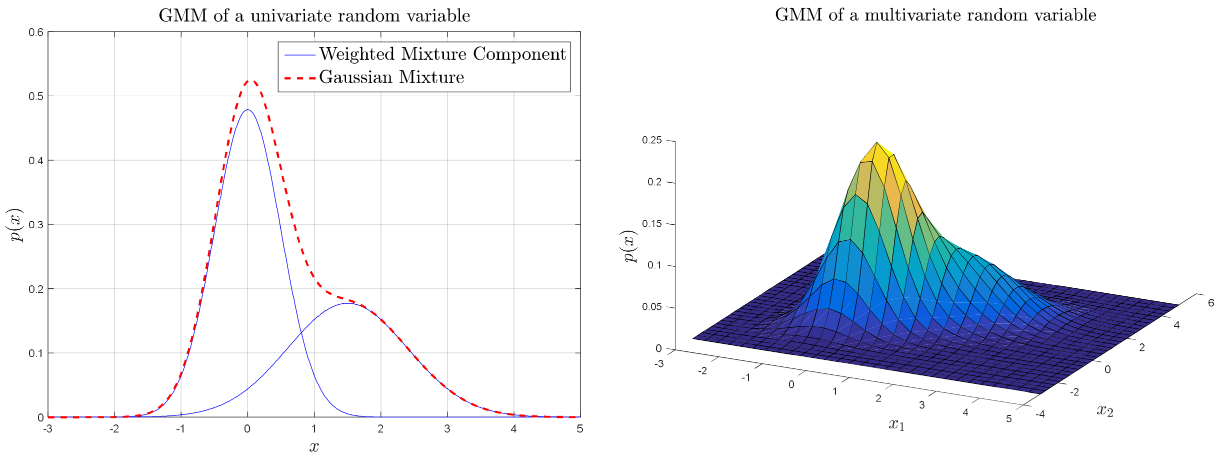


Figure 2-1: Example of a univariate and multivariate Gaussian Mixture Model with 2 mixture components

2.1.1 Fitting a Gaussian Mixture Model

In essence, there are $3m$ parameters, which need to be selected in order to construct a GMM. This includes the weight α_i, the mean μ_i and the variance σ_i^2 of each mixture component. In fact, the number of components in a GMM represent yet another free parameter. However, techniques to determine the optimal number of mixture components a-priori are not known to the author. The target probability distribution is usually not known. Rather, only a set of data is available. Fitting a GMM therefore becomes a parameter estimation problem with the goal to represent the data as close as possible.

One common approach to solve the approximation problem is to use the Maximum Likelihood estimation (see e.g. [139]). The latter aims at maximizing the likelihood of the GMM representing the data. However, the problem becomes nonlinear in the parameters and a direct analytic solution is not possible. Possibly the most famous technique is the Expectation Maximization approach ([55]). Initially, the parameters of the GMM are chosen at random. The Expectation Maximization algorithm is an iterative approach, essentially consisting of two steps. The first step associates the data points with the most probable mixture component given its current parameters. The second step updates the parameters to better fit the associated data points. The procedure is repeated until a convergence criterion is met. The algorithm is shown to increase the likelihood of the model representing the data with each step.

Contrary to most GMM applications, the target probability distribution is known in this thesis. This allows for the investigation of the approximation accuracy of the GMM after the parameter estimation. Furthermore, it is not mandatory to solve the parameter estimation problem for all GMM parameters. Instead, the mean μ_i and the variance σ_i^2 can be chosen a-priori and kept constant. By doing so, the parameter estimation problem becomes significantly easier.

Only adjusting the weights α_i reduces fitting the GMM to an approximation problem with a neural network structure. Consider a continuous *random variable* (rv) $\boldsymbol{x} \in \mathbb{R}^m$. In fact, choosing the mixture components as activation functions $\boldsymbol{\phi}^T(x) = [\phi_1(\boldsymbol{x}) \quad \dots \quad \phi_m(\boldsymbol{x})]$ with $\boldsymbol{\phi} \in \mathbb{R}^m$ results in a radial-basis function neural network with the output

$$\mathrm{y}(x) = \boldsymbol{\alpha}^T \boldsymbol{\phi}(x). \tag{2-3}$$

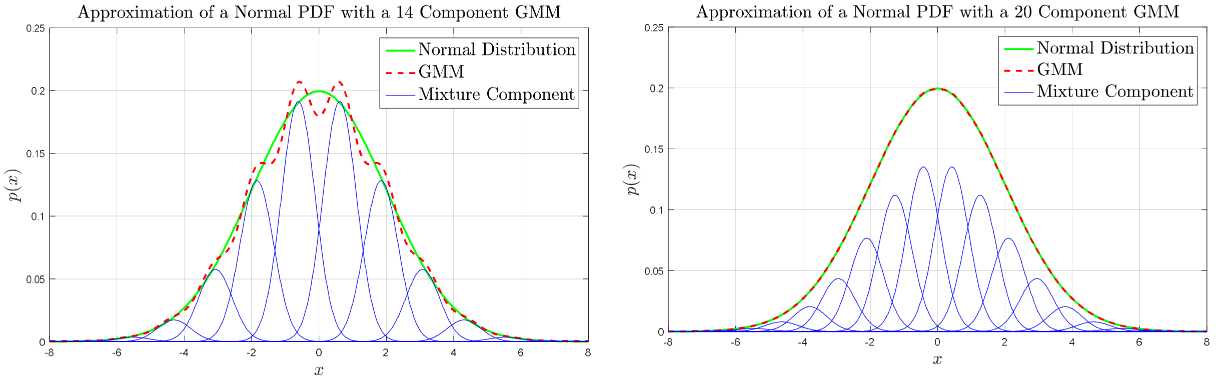


Figure 2-2: Approximation of a normal pdf by a GMM with 14 and 20 mixture components

Here, $\boldsymbol{\alpha}^T = [\alpha_1 \quad \cdots \quad \alpha_m], \boldsymbol{\alpha} \in \mathbb{R}^m$ represents the vector of mixture weights. To represent the GMM with a radial-basis function neural network, the i-th activation function $\phi_i(x)$ is henceforth given by the equation of a normal pdf with mean μ_i and standard deviation σ_i:

$$\phi_i(x) = \frac{1}{\sqrt{2\sigma_i^2\pi}} e^{-\frac{(x-\mu_i)^2}{2\sigma_i^2}}. \tag{2-4}$$

The neural network approximation property ensures that, given a sufficient number of neurons ϕ_i, the radial basis function neural network in (2-3) is able to approximate any continuous nonlinear function within a compact domain (see i.a. [130], [69], [165]). By increasing the number of neurons, the approximation error can be made arbitrarily small.

Let p_n be the probability distribution, which is to be approximated. In order to fit the GMM to p_n, the latter is evaluated at $m_e \geq m$ different points. Let $\boldsymbol{\Phi}_H = [\boldsymbol{\phi}(x_1) \quad \ldots \quad \boldsymbol{\phi}(x_{m_e})] \in \mathbb{R}^{m \times m_e}$ denote the matrix containing the regressor functions evaluated at m_e data points. The latter need to be chosen such that $\operatorname{rank}(\boldsymbol{\Phi}_H) = m$. Furthermore, let $\boldsymbol{Y}_H = [p_n(x_1) \quad \ldots \quad p_n(x_{m_e})] \in \mathbb{R}^{1 \times m_e}$ denote the evaluation of p_n at the respective data points. Fitting the GMM to p_n is achieved by solving the following optimization problem:

$$\begin{gathered} \min_{\alpha} \|\boldsymbol{Y}_H - \boldsymbol{\alpha}^T \boldsymbol{\Phi}_H\|^2 \\ \text{subject to } \alpha_i \geq 0 \; \forall i \in [1, m] \\ \sum_i^m \alpha_i = 1. \end{gathered} \tag{2-5}$$

Equation (2-5) represents a quadratic program with linear equality and inequality constraints. Approaches to solve this can be found i.a. in [16], [137]. This thesis simply uses the Matlab function *lsqlin* in order to solve (2-5).

Figure 2-2 exemplarily shows the difference between modelling a normal distribution with either 14 or 20 mixture components. In the first case, a significant error between the normal distribution and the GMM can be observed. In the second case, the approximation is nearly indistinguishable from p_n. Hence, increasing the number of mixture components leads to a decreasing approximation error. However, note that a GMM will in almost any case never exactly resemble a normal pdf.

2.1.2 Relevant Properties of Gaussian Mixture Models

This section summarizes important mathematical properties of a GMM, which will be used throughout the thesis.

Normal distributions are completely defined by their mean and covariance. Since the mixture components of a GMM are normal pdfs, the assumption arises, that a GMM is similarly defined by its mean and covariance. In fact, Flam ([63], [64], [65]) and Carreira-Perpiñán ([31]) show how to calculate the mean and covariance of a GMM. However, both values are not sufficient to fully describe a GMM (an exception arises if only one mixture component is used). This can be illustrated easily by imagining a multi-modal probability distribution, which can be approximated well by a GMM. Reducing the GMM to mean and covariance only is similar to approximating the multi-modal distribution with a normal one, inevitably introducing errors. However, higher moments could be used to describe the GMM. Unfortunately, it is not straight forward to derive the number of moments necessary to represent the GMM well. Instead, knowing the weight, mean and covariance of every mixture component in fact completely defines the GMM.

The following part deals with GMMs under affine transformations and the sum of a GMM with a normal rv. In order to derive consecutive properties, the *moment generating function* (mgf) of a GMM needs to be introduced.

Theorem 2.1 – Moment Generating Function of a GMM

Let $\boldsymbol{X} \in \mathbb{R}^n$ be a continuous random vector with Gaussian Mixture distribution $\boldsymbol{X} \sim \sum_{i=1}^{m} \alpha_i N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ as defined in Definition 2.1. The moment generating function M_x of $\boldsymbol{X}$ is

$$M_x(\boldsymbol{s}) = \sum_{i=1}^{m} \alpha_i M_{x,i}(\boldsymbol{s}), \tag{2-6}$$

where $M_{x,i}(\boldsymbol{s})$ denotes the moment generating function of the i-th mixture component.

Proof:

Per definition (see Definition B.20 in Appendix B.8), the mgf of the random variable $\boldsymbol{X}$ is

$$M_x(\boldsymbol{s}) = \mathbb{E}_x\left[e^{\boldsymbol{s}^T \boldsymbol{X}}\right]. \tag{2-7}$$

Inserting the definition of the expected value (see Definition B.13 in Appendix B.2) yields

$$M_x(\boldsymbol{s}) = \int_{-\infty}^{\infty} e^{\boldsymbol{s}^T \boldsymbol{x}} p_x(\boldsymbol{x}) d\boldsymbol{x} = \int_{-\infty}^{\infty} e^{\boldsymbol{s}^T \boldsymbol{x}} \sum_{i=1}^{m} \alpha_i N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\, d\boldsymbol{x}. \tag{2-8}$$

Swapping the integral and the sum yields

$$M_x(\boldsymbol{s}) = \sum_{i=1}^{m} \alpha_i \int_{-\infty}^{\infty} e^{\boldsymbol{s}^T \boldsymbol{x}} N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) d\boldsymbol{x}. \tag{2-9}$$

The mgf of the i-th mixture component $M_{x,i}(\boldsymbol{s})$ is equal to the mgf of a multivariate normal distribution (see (C-16) in Appendix C.2). Hence, the mgf of the GMM is

$$M_x(\boldsymbol{s}) = \sum_{i=1}^{m} \alpha_i \underbrace{\int_{-\infty}^{\infty} e^{s^T x} N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) d\boldsymbol{x}}_{M_{x,i}(\boldsymbol{s})} = \sum_{i=1}^{m} \alpha_i M_{x,i}(\boldsymbol{s}). \tag{2-10}$$

□

Using the mgf of the GMM, the result of a GMM under an affine transformation can be derived.

Theorem 2.2 – GMM under an affine transformation

Consider a continuous random vector $\boldsymbol{X} \in \mathbb{R}^n$ with Gaussian Mixture distribution $\boldsymbol{X} \sim \sum_{i=1}^{m} \alpha_i N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ as defined in Definition 2.1. Furthermore, consider the continuous random variable $\boldsymbol{Y} \in \mathbb{R}^n$. If $\boldsymbol{Y}$ is an affine transform of $\boldsymbol{X}$ such that $\boldsymbol{Y} = \boldsymbol{AX} + \boldsymbol{B}$ with $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{B} \in \mathbb{R}^n$ then $\boldsymbol{Y}$ is distributed as $\boldsymbol{Y} \sim \sum_{i=1}^{m} \alpha_i N(\boldsymbol{A\mu}_i + \boldsymbol{B}, \boldsymbol{A\Sigma}_i \boldsymbol{A}^T)$.

Proof:

Considering the mgf of $\boldsymbol{Y}$ and substituting the linear transformation $\boldsymbol{Y} = \boldsymbol{AX} + \boldsymbol{B}$ yields

$$M_y(\boldsymbol{s}) = \mathbb{E}_y\left[e^{\boldsymbol{s}^T \boldsymbol{Y}}\right] = \mathbb{E}_x\left[e^{\boldsymbol{s}^T (\boldsymbol{AX}+\boldsymbol{B})}\right] = e^{\boldsymbol{s}^T \boldsymbol{B}} \mathbb{E}_x\left[e^{\boldsymbol{s}^T \boldsymbol{AX}}\right] = e^{\boldsymbol{s}^T \boldsymbol{B}} M_x(\boldsymbol{A}^T \boldsymbol{s}). \tag{2-11}$$

With the mgf of (2-10) equation (2-11) becomes

$$M_y(\boldsymbol{s}) = e^{\boldsymbol{s}^T \boldsymbol{B}} M_x(\boldsymbol{A}^T \boldsymbol{s}) = \sum_{i=1}^{m} \alpha_i e^{\boldsymbol{s}^T \boldsymbol{B}} M_{x,i}(\boldsymbol{A}^T \boldsymbol{s}). \tag{2-12}$$

Inserting the mgf of a rv with a multivariate Gaussian probability distribution (see (C-16) in Appendix C.2) yields

$$M_y(\boldsymbol{s}) = \sum_{i=1}^{m} \alpha_i e^{\boldsymbol{s}^T \boldsymbol{B}} M_{x,i}(\boldsymbol{A}^T \boldsymbol{s}) = \sum_{i=1}^{m} \alpha_i e^{\boldsymbol{s}^T \boldsymbol{B}} e^{\boldsymbol{s}^T \boldsymbol{A}\left(\boldsymbol{\mu}_i + \frac{1}{2}\boldsymbol{\Sigma}_i \boldsymbol{A}^T \boldsymbol{s}\right)} = \sum_{i=1}^{m} \alpha_i e^{\boldsymbol{s}^T\left(\boldsymbol{A\mu}_i + \boldsymbol{B} + \frac{1}{2}\boldsymbol{A\Sigma}_i \boldsymbol{A}^T \boldsymbol{s}\right)}. \tag{2-13}$$

By defining $\boldsymbol{\mu}_{y,i} = \boldsymbol{A\mu}_i + \boldsymbol{B}$ and $\boldsymbol{\Sigma}_{y,i} = \boldsymbol{A\Sigma}_i \boldsymbol{A}^T$ equation (2-13) simplifies to

$$M_y(\boldsymbol{s}) = \sum_{i=1}^{m} \alpha_i \underbrace{e^{\boldsymbol{s}^T\left(\boldsymbol{\mu}_{y,i} + \frac{1}{2}\boldsymbol{\Sigma}_{y,i} \boldsymbol{s}\right)}}_{M_{y,i}} = \sum_{i=1}^{m} \alpha_i M_{y,i}(\boldsymbol{s}) \tag{2-14}$$

Note, that the exponential function retains the structure of a mgf $M_{y,i}$ of a normal probability distribution with mean $\boldsymbol{\mu}_{y,i}$ and covariance $\boldsymbol{\Sigma}_{y,i}$.

Using equation (2-10) results in

$$\begin{aligned} M_y(\boldsymbol{s}) &= \sum_{i=1}^{m} \alpha_i M_{y,i}(\boldsymbol{s}) = \sum_{i=1}^{m} \alpha_i \int_{-\infty}^{\infty} e^{\boldsymbol{s}^T\boldsymbol{y}} N\left(\boldsymbol{\mu}_{y,i}, \boldsymbol{\Sigma}_{y,i}\right) d\boldsymbol{y} \\ &= \int_{-\infty}^{\infty} e^{\boldsymbol{s}^T\boldsymbol{y}} \sum_{i=1}^{m} \alpha_i N\left(\boldsymbol{\mu}_{y,i}, \boldsymbol{\Sigma}_{y,i}\right) d\boldsymbol{y} = \int_{-\infty}^{\infty} e^{\boldsymbol{s}^T\boldsymbol{y}} p_y(\boldsymbol{y}) d\boldsymbol{y} = \mathbb{E}_y\left[e^{\boldsymbol{s}^T\boldsymbol{Y}}\right]. \end{aligned} \tag{2-15}$$

As a result, the distribution of $\boldsymbol{Y}$ is

$$p_y(\boldsymbol{y}) = \sum_{i=1}^{m} \alpha_i N\left(\boldsymbol{\mu}_{y,i}, \boldsymbol{\Sigma}_{y,i}\right) = \sum_{i=1}^{m} \alpha_i N(\boldsymbol{A}\boldsymbol{\mu}_i + \boldsymbol{B}, \boldsymbol{A}\boldsymbol{\Sigma}_i\boldsymbol{A}^T). \tag{2-16}$$

□

Theorem 2.2 implies that under an affine transformation each mixture component of the GMM can be treated separately. The following theorem deals with the sum of a GMM distributed rv and a normally distributed rv.

Theorem 2.3 – Sum of a GMM distributed rv with an independent normally distributed rv

Let $\boldsymbol{X} \in \mathbb{R}^n$ and $\boldsymbol{Y} \in \mathbb{R}^n$ be two independent random vectors with probability distributions $\boldsymbol{X} \sim \sum_{i=1}^{m} \alpha_i N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ and $\boldsymbol{Y} \sim N\left(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y\right)$. Then the linear combination $\boldsymbol{Z} = \boldsymbol{X} + \boldsymbol{Y}$ has the probability distribution $\boldsymbol{Z} \sim \sum_{i=1}^{m} \alpha_i N\left(\boldsymbol{\mu}_i + \boldsymbol{\mu}_y, \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_y\right)$.

Proof:

Adding two independent rvs amount to multiplying their respective mgf:

$$M_z(\boldsymbol{s}) = M_x(\boldsymbol{s}) M_y(\boldsymbol{s}). \tag{2-17}$$

Considering the mgf of a GMM in (2-10) as well as the mgf of a multivariate Gaussian (see (C-16) in Appendix C.2) yields

$$\begin{aligned} M_z(\boldsymbol{s}) &= \sum_{i=1}^{m} \alpha_i M_{x,i}(\boldsymbol{s})\, M_y(\boldsymbol{s}) = \sum_{i=1}^{m} \alpha_i e^{\boldsymbol{s}^T\left(\boldsymbol{\mu}_i + \frac{1}{2}\boldsymbol{\Sigma}_i \boldsymbol{s}\right)} e^{\boldsymbol{s}^T\left(\boldsymbol{\mu}_y + \frac{1}{2}\boldsymbol{\Sigma}_y \boldsymbol{s}\right)} \\ &= \sum_{i=1}^{m} \alpha_i e^{\boldsymbol{s}^T\left(\boldsymbol{\mu}_i + \boldsymbol{\mu}_y + \frac{1}{2}\boldsymbol{\Sigma}_i \boldsymbol{s} + \frac{1}{2}\boldsymbol{\Sigma}_y \boldsymbol{s}\right)}. \end{aligned} \tag{2-18}$$

By defining $\boldsymbol{\mu}_{z,i} = \boldsymbol{\mu}_i + \boldsymbol{\mu}_y$ and $\boldsymbol{\Sigma}_{z,i} = \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_y$, $M_z(\boldsymbol{s})$ becomes

$$M_z(\boldsymbol{s}) = \sum_{i=1}^{m} \alpha_i e^{\boldsymbol{s}^T\left(\boldsymbol{\mu}_{z,i} + \frac{1}{2}\boldsymbol{\Sigma}_{z,i} \boldsymbol{s}\right)}. \tag{2-19}$$

Note that (2-19) resembles the mgf of a GMM with distribution $\boldsymbol{Z} \sim \sum_{i=1}^{m} \alpha_i N(\boldsymbol{\mu}_{z,i}, \boldsymbol{\Sigma}_{z,i})$. Resubstituting $\boldsymbol{\mu}_{z,i}$ and $\boldsymbol{\Sigma}_{z,i}$ yields $\boldsymbol{Z} \sim \sum_{i=1}^{m} \alpha_i N(\boldsymbol{\mu}_i + \boldsymbol{\mu}_y, \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_y)$.

□

Again, Theorem 2.2 implies that each mixture component of the GMM can be treated separately if the summands are independent.

2.2 Gaussian Processes for uncertainty quantification

Gaussian Processes (GPs) play an ever increasing role in various scientific disciplines, including machine learning, robotics, and signal processing. A GP is a probabilistic model, which allows for a probabilistic treatment of uncertainties, including regression and classification tasks. Three key features made GP regression popular. First, the GP allows for an elegant modelling of uncertainties and statistical dependencies between points in its domain. Second, the GP model can completely be constructed using available data; only little a-priori knowledge is required. However, existing information can be incorporated in a so called prior model. A means to achieve a fit between the model and the data is Bayesian inference. The third key feature of GPs is that inference is fast and tractable for data with Gaussian likelihood. As a result, regression based on data with Gaussian likelihood has been studied extensively (see i.a. [173], [21], [152]). This chapter formally introduces the basics of GPs as well as GP regression. The following discussion largely follows the derivation in [173]. Definition 2.2 is commonly used in the literature in order to define a GP.

Definition 2.2 – Gaussian Process ([173, p. 13], Definition 2.1)

A *Gaussian Process* is a collection of random variables, any finite number of which have a joint Gaussian distribution.

A GP provides a probabilistic view of modelling a function $f(\boldsymbol{x})$. Therefore, let $\boldsymbol{x} \in \mathbb{R}^n$ denote the input vector and consider the following auxiliary model $f(\boldsymbol{x})\colon \mathbb{R}^n \to \mathbb{R}$, which is formed by a linear combination of parameters $\boldsymbol{w} \in \mathbb{R}^h$ and nonlinear basis functions $\boldsymbol{\phi}(\boldsymbol{x})\colon \mathbb{R}^n \to \mathbb{R}^h$:

$$f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}). \tag{2-20}$$

The function $\boldsymbol{\phi}(\boldsymbol{x})$ represents a so-called feature function, which projects the input $\boldsymbol{x}$ into a higher dimensional feature space. This projection overcomes the limited expressiveness of a linear model ([173]). For the cause of better readability, subscripts for the feature function will resemble its argument, i.e. $\boldsymbol{\phi}_{\boldsymbol{x}} = \boldsymbol{\phi}(\boldsymbol{x})$. It is worth noting that the machine learning literature prevalently employs the transposed version of (2-20). The form (2-20) is chosen to resemble the structure of the adaptive signal in (3-15).

The model in (2-20) becomes a GP once a probability distribution $p(\boldsymbol{w})$ is placed on the parameter vector $\boldsymbol{w}$. Since $p(\boldsymbol{w})$ can be updated, the initial distribution of $\boldsymbol{w}$ is called prior distribution. If not stated otherwise, this thesis assumes multivariate normal priors. As a consequence, also the function $f(\boldsymbol{x})$ has a probability distribution. Consider a set $\Omega \subset \mathbb{R}^n$ and choose $i \in \mathbb{Z}^+$ input vectors $\boldsymbol{x} \in \Omega$ from this set. Using these input vectors in the model (2-20) yields

$$\begin{pmatrix} f(\boldsymbol{x}_1) & \dots & f(\boldsymbol{x}_i) \end{pmatrix} = \boldsymbol{w}^T \begin{pmatrix} \boldsymbol{\phi}(\boldsymbol{x}_1) & \dots & \boldsymbol{\phi}(\boldsymbol{x}_i) \end{pmatrix}. \tag{2-21}$$

Due to the introduction of the prior distribution, the parameter vector $\boldsymbol{w}$ in (2-21) is multivariate Gaussian. Consequently, also the vector of random variables $\begin{pmatrix} f(\boldsymbol{x}_1) & \dots & f(\boldsymbol{x}_i) \end{pmatrix}^T$ in (2-21) is multivariate Gaussian. This result is independent of the number of inputs i. Hence per Definition 2.2, $\begin{pmatrix} f(\boldsymbol{x}_1) & \dots & f(\boldsymbol{x}_i) \end{pmatrix}^T$ is a GP. In this context, $f(\boldsymbol{x})$ is sayed to follow a GP defined on Ω.

Note that the joint distribution $p(f(\boldsymbol{x}_1),\dots,f(\boldsymbol{x}_i))$ is multivariate Gaussian. Hence, the GP is fully described by its mean and covariance function. Therefore, assume a prior distribution $p(\boldsymbol{w})\sim N(\boldsymbol{w}|\bar{\boldsymbol{w}},\boldsymbol{\Sigma}_w)$ with mean $\bar{\boldsymbol{w}}$ and variance $\boldsymbol{\Sigma}_w$ on the weights $\boldsymbol{w}$. Then the mean $m(\boldsymbol{x})$ of $f(\boldsymbol{x})$ is a function dependent on the input $\boldsymbol{x}$:

$$m(\boldsymbol{x}) = \mathbb{E}_f[f(\boldsymbol{x})] = \mathbb{E}_f[\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x})] = \underbrace{\mathbb{E}_w[\boldsymbol{w}^T]}_{=\bar{\boldsymbol{w}}}\boldsymbol{\phi}(\boldsymbol{x}) = \bar{\boldsymbol{w}}^T\boldsymbol{\phi}(\boldsymbol{x}). \tag{2-22}$$

Similarly, the covariance function $\Sigma(\boldsymbol{x},\boldsymbol{x}')$ of $f(\boldsymbol{x})$ is

$$\begin{aligned}\Sigma(\boldsymbol{x},\boldsymbol{x}') &= \mathbb{E}_f\left[\left(f(\boldsymbol{x})-m(\boldsymbol{x})\right)^T\left(f(\boldsymbol{x}')-m(\boldsymbol{x}')\right)\right]\\ &= \mathbb{E}_f\left[\left(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x})-\bar{\boldsymbol{w}}^T\boldsymbol{\phi}(\boldsymbol{x})\right)^T\left(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}')-\bar{\boldsymbol{w}}^T\boldsymbol{\phi}(\boldsymbol{x}')\right)\right]\\ &= \boldsymbol{\phi}^T(\boldsymbol{x})\underbrace{\mathbb{E}_w[(\boldsymbol{w}^T-\bar{\boldsymbol{w}}^T)^T(\boldsymbol{w}^T-\bar{\boldsymbol{w}}^T)]}_{=\boldsymbol{\Sigma}_w}\boldsymbol{\phi}(\boldsymbol{x}') = \boldsymbol{\phi}^T(\boldsymbol{x})\boldsymbol{\Sigma}_w\boldsymbol{\phi}(\boldsymbol{x}').\end{aligned} \tag{2-23}$$

Note, that the covariance function $\Sigma(\boldsymbol{x},\boldsymbol{x}')$ takes two inputs, namely $\boldsymbol{x}$ and $\boldsymbol{x}'$. In essence, this notation is attributed to the fact that GPs are defined with respect to a countable number of data points. Hence, the covariance function also computes the covariance between two data points. The notation $(\boldsymbol{x},\boldsymbol{x}')$ is used to indicate that these data points need to be part of the set Ω, which the GP is defined on. Choosing $\boldsymbol{x}=\boldsymbol{x}'$ yields the variance of $f(\boldsymbol{x})$ at $\boldsymbol{x}$. As an example, the application of (2-22) to (2-21) yields:

$$(m(\boldsymbol{x}_1) \quad \dots \quad m(\boldsymbol{x}_i)) = \bar{\boldsymbol{w}}^T(\boldsymbol{\phi}(\boldsymbol{x}_1) \quad \dots \quad \boldsymbol{\phi}(\boldsymbol{x}_i)) \tag{2-24}$$

Similarly the application of (2-23) to (2-21) yields

$$\Sigma\left(\begin{pmatrix}\boldsymbol{x}_1\\ \vdots\\ \boldsymbol{x}_i\end{pmatrix},\begin{pmatrix}\boldsymbol{x}_1\\ \vdots\\ \boldsymbol{x}_i\end{pmatrix}\right) = \begin{bmatrix}\Sigma_{11} & \dots & \Sigma_{1i}\\ \vdots & \ddots & \vdots\\ \Sigma_{i1} & \dots & \Sigma_{ii}\end{bmatrix} = \begin{bmatrix}\boldsymbol{\phi}^T(\boldsymbol{x}_1)\boldsymbol{\Sigma}_w\boldsymbol{\phi}(\boldsymbol{x}_1) & \dots & \boldsymbol{\phi}^T(\boldsymbol{x}_1)\boldsymbol{\Sigma}_w\boldsymbol{\phi}(\boldsymbol{x}_i)\\ \vdots & \ddots & \vdots\\ \boldsymbol{\phi}^T(\boldsymbol{x}_i)\boldsymbol{\Sigma}_w\boldsymbol{\phi}(\boldsymbol{x}_1) & \dots & \boldsymbol{\phi}^T(\boldsymbol{x}_i)\boldsymbol{\Sigma}_w\boldsymbol{\phi}(\boldsymbol{x}_i)\end{bmatrix}. \tag{2-25}$$

Note, that each subset of $(\boldsymbol{x}_1 \quad \cdots \quad \boldsymbol{x}_i)$ is again multivariate Gaussian. Furthermore, its mean and covariance matrix can be easily extracted from (2-24) and (2-25). With (2-22) and (2-23), the common notation to indicate that $f(\boldsymbol{x})$ follows a GP is

$$f(\boldsymbol{x})\sim GP\big(m(\boldsymbol{x}),\Sigma(\boldsymbol{x},\boldsymbol{x}')\big). \tag{2-26}$$

The literature often distinguishes two equal ways to deal with GPs, namely the weight and function space view. The former focusses on the distribution of the parameters $\boldsymbol{w}$. In this, a draw from the parameter distribution leads to a specific realization of the function $f(\boldsymbol{x})$. The connection between the parameters and the function $f(\boldsymbol{x})$ gives rise to an equivalent interpretation of GPs. Instead of focusing on the weights, GPs are often dealt with in function space, where they are considered to be distributions over functions. That is, GPs can be seen as a family of functions, where each member is associated with a certain probability. Consequently, a draw from a GP directly corresponds to the draw of a function. The following example is intended to visualize the GP model and give a first impression of its use.

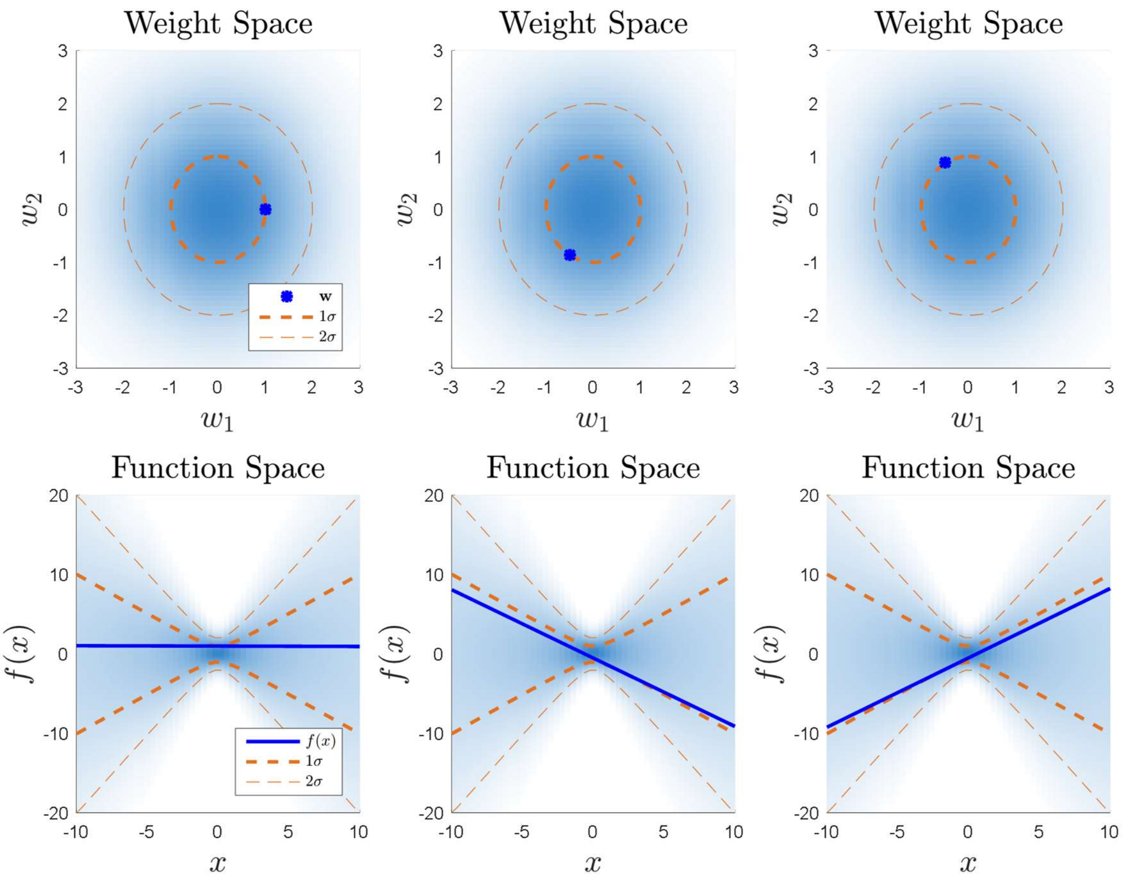


Figure 2-3: General idea of Gaussian Processes with comparison between weight (top) and function (bottom) space view. Based on plots from [91].

Example 2-1: **Possible View of a Gaussian Process**

Consider the simplified linear version of the model in (2-20) with $\boldsymbol{w} = [w_1 \quad w_2]^T$ and $\boldsymbol{\phi} = [1 \quad x]^T$. Assume a zero-mean multivariate normal distribution $p(\boldsymbol{w})\sim N(\boldsymbol{0}, \boldsymbol{I})$ as the prior for the parameters $\boldsymbol{w}$. The top row of Figure 2-3 depicts the so called weight space. In any case, the dashed circles represent the standard deviation and twice the standard deviation of $p(\boldsymbol{w})$, respectively. The probability density function of the prior $p(\boldsymbol{w})$ is projected onto the weight space and depicted as a color map with its maximum at the origin $\boldsymbol{w} = [0 \quad 0]^T$. A pair of parameters resembles a realization of $\boldsymbol{w}$ and therefore a point in the weight space.

The bottom row of Figure 2-3 depicts the function space, which evaluates the underlying function $f(x)$ of the model in (2-20) for a domain $x \in [-10,10]$. Similarly to the weight space, the dashed lines represent the standard deviation and twice the standard deviation of $p\big(f(x)\big)$, respectively. The probability density function of the prior $p(\boldsymbol{w})$ is projected into the function space and depicted as a color map. Note, that GPs are defined for a set of data points. Hence, depicting lines instead of points in the function space represents a slight abuse of notation.

Assume that the pair of weights $\boldsymbol{w} = [1 \quad 0]^T$ is drawn from the multivariate Gaussian prior distribution $p(\boldsymbol{w})$. This pair resembles the point in the weight space depicted in Figure 2-3 (top left). Note that this point lies on the one standard deviation circle of the prior. The function $f(\boldsymbol{x})$ corresponding to this pair of parameters is shown in Figure 2-3 (bottom left) as a horizontal line. Now assume a rotation of the weight space around its origin. Figure 2-3 (middle) depicts the result. The new pair of weights is a rotated version of the initial draw, but still lies on the circle representing a standard deviation of one. Hence, the probability of picking the initial or new set of parameters remains the same and is independent from the rotation. Similarly, even though the function depicted in Figure 2-3 (bottom middle) looks utterly different to the initial horizontal line, the probability of picking it stays the same. Rotating the weight space further yields a third pair of weights, which corresponds to the results in Figure 2-3 (right). Again, the probability of picking this set of parameters or the resulting function is equal to the two previous cases.

The connection between the parameters and the generated function gives rise to an equivalent interpretation of GPs. Instead of focusing on the weight space view, GPs are often dealt with in function space, where they are considered to be distributions over functions. That is, GPs can be seen as a family of functions, where each member is associated with a certain probability. Consecutively, a draw from a GP directly corresponds to the draw of a function.

GPs are wildly used in order to model uncertainties. Most applied algorithms assume little to no prior knowledge about the uncertainty. Instead, they infer a GP model from a set of data using regression. However, in practical applications prior knowledge about the uncertainty is often available. If the parameters of a physical model are estimated based on experiments, the results are often formulated with a stochastic uncertainty. This is e.g. due to measurement errors. If the parametric uncertainty in turn is normal, this formulation already represents a GP. To this length, the following example uses a GP to model the parameter uncertainties in the lift coefficient.

Example 2-2: **GP used to model uncertainty in aerodynamic data**

This example intends to show how GPs can be applied to practical problems with prior knowledge of parametric uncertainties. Consider the linear approximation of the lift coefficient C_L over the angle-of-attack α curve:

$$C_L = C_{L0} + C_{L\alpha}\alpha. \tag{2-27}$$

Here, additional influences, such as the change of the lift coefficient due to a pitch rate C_{Lq}, are neglected. Assume, that numerical values for the coefficients C_{L0} and $C_{L\alpha}$ result from an experiment. For this sake, let the expected values of C_{L0} and $C_{L\alpha}$ be $\bar{C}_{L0} = 0.1$ and $\bar{C}_{L\alpha} = 0.1$. Due to the experiment, the parameters are subject to uncertainty, which in turn is assumed to be normally distributed. Therefore, assume a 1σ error of 4% in C_{L0} as well as $C_{L\alpha}$, respectively. As a result, the standard deviations are $\sigma_{C_{L0}} = 0.04$ and $\sigma_{C_{L\alpha}} = 0.04$. Note, that C_{L0} and $C_{L\alpha}$ are assumed to be stochastically independent. Furthermore, note that the mentioned numerical values due not necessarily reflect an existing aircraft, but are picked to highlight the argument of this example. Following (2-22) and (2-23), the mean and covariance function of the GP are

$$\begin{aligned} m(\alpha) &= \begin{bmatrix} 0.1 & 0.1 \end{bmatrix} \begin{bmatrix} 1 \\ \alpha \end{bmatrix}, \\ \Sigma(\alpha, \alpha') &= \begin{bmatrix} 1 & \alpha \end{bmatrix} \begin{bmatrix} 0.04^2 & 0 \\ 0 & 0.04^2 \end{bmatrix} \begin{bmatrix} 1 \\ \alpha' \end{bmatrix}. \end{aligned} \tag{2-28}$$

Figure 2-4 (left) shows the realization of (2-28) over $\alpha \in [-5,12]$. Here the mean function, as well as 1 and 2 standard deviations are depicted. Due to the linear nature of the parametric model, the standard deviation increases proportionally with the angle of attack.

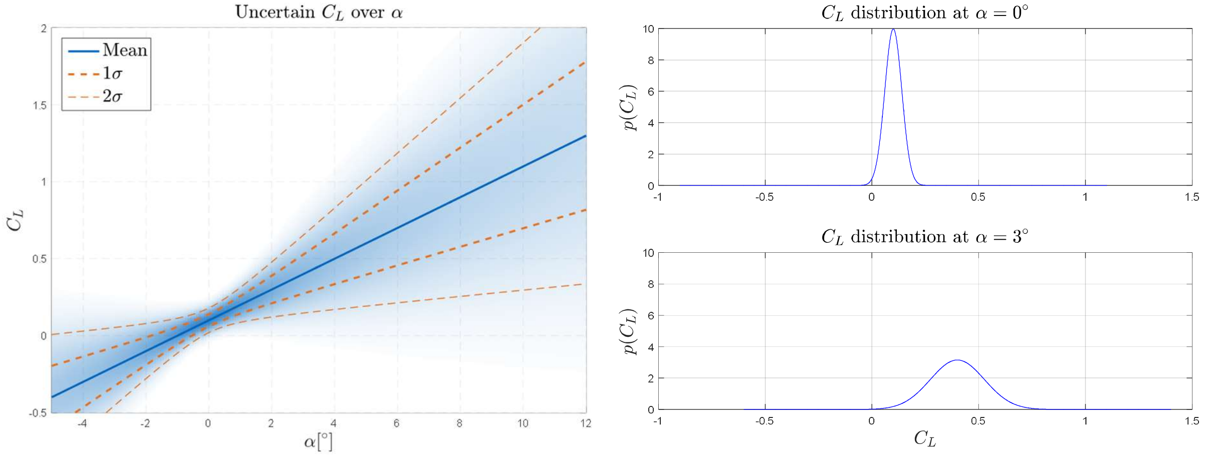


Figure 2-4: Example of a GP used to model uncertain parameters in the $C_L - \alpha$ curve

The evaluation of the GP model at a single point yields a normal distribution of the lift coefficient C_L at that point. Figure 2-4 (right) shows the evaluation of the GP at $\alpha_1 = 0°$ and $\alpha_2 = 3°$. Using equation (2-28) yields

$$\begin{aligned} m(0°) &= 0.1, \qquad \sigma^2(0°, 0°) = 0.0016, \\ m(3°) &= 0.4, \qquad \sigma^2(3°, 3°) = 0.016. \end{aligned} \tag{2-29}$$

Note, that the parameters of this distribution are heavily dependent on α. Since the GP model allows to evaluate the distribution of C_L at different α, it achieves a model of the uncertainty in C_L at different points in the state space. Even though this result seems trivial, it plays a vital role for the monitoring algorithm developed in this thesis.

The last example showed how GPs can model parametric uncertainties if prior knowledge is available. However, situations might occur, in which little to no information is available or the uncertainty is completely unknown. The following section shows how to use regression to infer a GP model from a set of data points.

2.2.1 Gaussian Process Regression

In cases with less to none prior information about the uncertainty, it can still be inferred from gathered data using GP regression. At first, a prior distribution is introduced for the model parameters. Without initial knowledge about the parameters, $p(\boldsymbol{w})$ is often selected to be a zero-mean multivariate Gaussian with variance one. While a-priori information can be incorporated, Rasmussen argues in [173] that choosing a zero mean prior is not a drastic limitation, since the posterior distribution is not confined to zero. Remember that a GP can be seen as a family of functions, where each member is associated with a certain probability. The underlying goal of GP regression is to narrow down the initially assumed array of functions using gathered data. This is achieved by increasing the probability of functions, which better match the observed data. Vice versa, the probability of functions, which do not fit the data, is decreased.

To perform GP regression, consider an extended version of the model in (2-20):

$$\begin{aligned} f(\boldsymbol{x}) &= \boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}), \\ y &= f(\boldsymbol{x}) + \boldsymbol{\varepsilon}. \end{aligned} \tag{2-30}$$

Here, $f(\boldsymbol{x})\colon \mathbb{R}^n \rightarrow \mathbb{R}$ represents the underlying function, which is to be fit to the data. In turn, the output y represents the underlying function $f(\boldsymbol{x})$ corrupted by noise. Throughout this thesis, the measurement noise $\boldsymbol{\varepsilon}$ is assumed to be i.i.d. Gaussian with zero mean and variance σ_n^2, i.e. $\boldsymbol{\varepsilon} \sim N(\boldsymbol{0}, \sigma_n^2\boldsymbol{I})$.

The following sections show how to perform GP regression on a set of input-output data in weight and function space. While the function space view is prevalently used in the reminder of the thesis and equivalent to the derivation in the weight space, the latter is intuitively easier to understand. Hence, in section 2.2.1.1 we first focus on GP regression derived from the weight space view and extend the result to the function space in section 2.2.1.2.

2.2.1.1 Gaussian Process Regression – Weight Space View

Let $\boldsymbol{X}_H \in \mathbb{R}^{n\times p_H}$ *with* $\boldsymbol{X}_H = [\boldsymbol{x}_1 \;\; \cdots \;\; \boldsymbol{x}_{p_H}]$ denote a set of p_H input data points and let $\boldsymbol{Y}_H \in \mathbb{R}^{1\times p_H}$ with $\boldsymbol{Y}_H = [y_1 \;\; \cdots \;\; y_{p_H}]$ denote a set of associated observations. The main goal of GP regression is to infer a belief of the weights $\boldsymbol{w}$ in (2-30) based on the observed data $(\boldsymbol{X}_H, \boldsymbol{Y}_H)$. This inference is performed by employing Bayes' theorem ([11]):

$$p(\boldsymbol{w}|\boldsymbol{Y}_H, \boldsymbol{X}_H) = \frac{p(\boldsymbol{Y}_H|\boldsymbol{X}_H, \boldsymbol{w})p(\boldsymbol{w})}{p(\boldsymbol{Y}_H|\boldsymbol{X}_H)}. \tag{2-31}$$

In general, Bayes' theorem connects the belief in a proposition before and after accounting for evidence. Here, $p(\boldsymbol{w})$ denotes the prior or initial belief in the parameters before seeing the data $(\boldsymbol{Y}_H, \boldsymbol{X}_H)$. The distribution $p(\boldsymbol{Y}_H|\boldsymbol{X}_H, \boldsymbol{w})$ denotes the likelihood of seeing the outcome $\boldsymbol{Y}_H$ given $\boldsymbol{X}_H$ and the (prior) belief on $\boldsymbol{w}$. The posterior distribution $p(\boldsymbol{w}|\boldsymbol{Y}_H, \boldsymbol{X}_H)$ represents the updated belief in the weights after performing inference on the data $(\boldsymbol{Y}_H, \boldsymbol{X}_H)$. Finally, $p(\boldsymbol{Y}_H|\boldsymbol{X}_H)$ is the normalization constant or marginal likelihood. Note that the prior $p(\boldsymbol{w})$ is not conditioned on $\boldsymbol{X}_H$ as it is preselected and doesn't depend on the data $\boldsymbol{X}_H$. If not stated otherwise, a Gaussian prior with mean $\overline{\boldsymbol{w}}$ and covariance $\boldsymbol{\Sigma}_p$ is used in this thesis:

$$p(\boldsymbol{w}) = N(\overline{\boldsymbol{w}}, \boldsymbol{\Sigma}_w). \tag{2-32}$$

The output equation in (2-30) defines the likelihood $p(\boldsymbol{Y}_H|\boldsymbol{X}_H,\boldsymbol{w})$ of observing $\boldsymbol{Y}_H$ given the inputs $\boldsymbol{X}_H$ and the current belief in the parameters $\boldsymbol{w}$:

$$\begin{aligned} p(\boldsymbol{Y}_H|\boldsymbol{X}_H,\boldsymbol{w}) &= \prod_{i=1}^{p} p(y_i|\boldsymbol{x}_i,\boldsymbol{w}) = \prod_{i=1}^{p} \frac{1}{\sqrt{2\pi\sigma_n^2}} e^{-\frac{\left(y_i-\boldsymbol{w}^T\boldsymbol{\phi}_{x_i}\right)^2}{2\sigma_n^2}} \\ &= \frac{1}{(2\pi\sigma_n^2)^{\frac{p}{2}}} e^{-\left((\boldsymbol{Y}_H-\boldsymbol{w}^T\boldsymbol{\phi}_{X_H})\frac{1}{2\sigma_n^2}(\boldsymbol{Y}_H-\boldsymbol{w}^T\boldsymbol{\phi}_{X_H})^T\right)} \end{aligned} \tag{2-33}$$

Lastly, the marginal likelihood can be expanded as

$$p(\boldsymbol{Y}_H|\boldsymbol{X}_H) = \int p(\boldsymbol{Y}_H|\boldsymbol{X}_H,\boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w}. \tag{2-34}$$

The marginal likelihood ensures that the posterior distribution $p(\boldsymbol{w}|\boldsymbol{Y}_H,\boldsymbol{X}_H)$ integrates to one. The marginal doesn't change the mean and the covariance of the posterior if all distributions in (2-31) are of the Gaussian type. Hence, its computation is often omitted. As a result, inference is often performed based on the proportionality of the posterior with regard to the product of the likelihood and the prior:

$$p(\boldsymbol{w}|\boldsymbol{Y}_H,\boldsymbol{X}_H) \propto p(\boldsymbol{Y}_H|\boldsymbol{X}_H,\boldsymbol{w})p(\boldsymbol{w}). \tag{2-35}$$

With all distributions being Gaussian, inserting (2-32) and (2-33) into (2-35) gives

$$\begin{aligned} p(\boldsymbol{w}|\boldsymbol{Y}_H,\boldsymbol{X}_H) \propto \exp\Bigg[-\frac{1}{2}\Bigg(&\left(\boldsymbol{Y}_H-\boldsymbol{w}^T\boldsymbol{\phi}_{X_H}\right)\frac{1}{\sigma_n^2}\left(\boldsymbol{Y}_H-\boldsymbol{w}^T\boldsymbol{\phi}_{X_H}\right)^T \\ &+ (\boldsymbol{w}-\bar{\boldsymbol{w}})^T\boldsymbol{\Sigma}_w^{-1}(\boldsymbol{w}-\bar{\boldsymbol{w}})\Bigg)\Bigg]. \end{aligned} \tag{2-36}$$

Using the calculation rules in Appendix C.3, the joint probability $p(\boldsymbol{w},\boldsymbol{Y}_H|\boldsymbol{X}_H)$ is

$$\begin{aligned} &p(\boldsymbol{w},\boldsymbol{Y}_H|\boldsymbol{X}_H) \\ \propto \exp&\left(-\frac{1}{2}\begin{bmatrix}(\boldsymbol{w}-\bar{\boldsymbol{w}}) \\ \left(\boldsymbol{Y}_H-\bar{\boldsymbol{w}}^T\boldsymbol{\phi}_{X_H}\right)^T\end{bmatrix}^T \begin{bmatrix}\boldsymbol{\Sigma}_w^{-1}+\boldsymbol{\phi}_{X_H}\frac{1}{\sigma_n^2}\boldsymbol{\phi}_{X_H}^T & -\frac{1}{\sigma_n^2}\boldsymbol{\phi}_{X_H}\boldsymbol{I} \\ -\frac{1}{\sigma_n^2}\boldsymbol{I}\boldsymbol{\phi}_{X_H}^T & \frac{1}{\sigma_n^2}\boldsymbol{I}\end{bmatrix} \begin{bmatrix}(\boldsymbol{w}-\bar{\boldsymbol{w}}) \\ \left(\boldsymbol{Y}_H-\bar{\boldsymbol{w}}^T\boldsymbol{\phi}_{X_H}\right)^T\end{bmatrix}\right). \end{aligned} \tag{2-37}$$

Note that the mean and variance of the joint $p(\boldsymbol{w},\boldsymbol{Y}_H|\boldsymbol{X}_H)$ are independent of the marginal likelihood. Employing the rules for special matrix inverses in Appendix A.4 results in the representation of (2-37) as a multivariate normal distribution:

$$p(\boldsymbol{w},\boldsymbol{Y}_H|\boldsymbol{X}_H) = N\left(\begin{pmatrix}\bar{\boldsymbol{w}} \\ \left(\bar{\boldsymbol{w}}^T\boldsymbol{\phi}_{X_H}\right)^T\end{pmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_w & \boldsymbol{\Sigma}_w\boldsymbol{\phi}_{X_H} \\ \boldsymbol{\phi}_{X_H}^T\boldsymbol{\Sigma}_w & \boldsymbol{\phi}_{X_H}^T\boldsymbol{\Sigma}_w\boldsymbol{\phi}_{X_H}+\sigma_n^2\boldsymbol{I}\end{bmatrix}\right). \tag{2-38}$$

Conditioning (2-38) on the output data $\boldsymbol{Y}_H$ (see Appendix C.3 for calculation rules) yields the posterior distribution $p(\boldsymbol{w}|\boldsymbol{Y}_H, \boldsymbol{X}_H)$:

$$p(\boldsymbol{w}|\boldsymbol{Y}_H, \boldsymbol{X}_H) = N\begin{pmatrix} \bar{\boldsymbol{w}} + \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H}\left[\boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\left(\boldsymbol{Y}_H - \bar{\boldsymbol{w}}^T \boldsymbol{\phi}_{X_H}\right)^T, \\ \boldsymbol{\Sigma}_w - \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H}\left[\boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1} \boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \end{pmatrix}. \qquad \textbf{(2-39)}$$

Equation (2-39) forms a model for the weights based on the prior believe in the parameters and the observed data. In particular, the posterior mean $\bar{\boldsymbol{w}}_{post}$ is

$$\bar{\boldsymbol{w}}_{post} = \bar{\boldsymbol{w}} + \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H}\left[\boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\left(\boldsymbol{Y}_H - \bar{\boldsymbol{w}}^T \boldsymbol{\phi}_{X_H}\right)^T. \qquad \textbf{(2-40)}$$

And the posterior covariance $\boldsymbol{\Sigma}_{post}$ is

$$\boldsymbol{\Sigma}_{post} = \boldsymbol{\Sigma}_w - \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H}\left[\boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1} \boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w. \qquad \textbf{(2-41)}$$

The goal in GP regression is not only to form a model which best matches the observed data, but also to predict outcomes of the underlying function $f(\boldsymbol{x})$ for data points, which are not necessarily part of the training set $(\boldsymbol{X}_H, \boldsymbol{Y}_H)$. Therefore, let $\boldsymbol{x}_*$ denote a test input and let $f_{\boldsymbol{x}_*}$ denote the respective function value. The posterior predictive distribution allows for the assessment of $f_{\boldsymbol{x}_*}$. It is obtained by marginalizing the joint distribution $p\left(f_{\boldsymbol{x}_*}, \boldsymbol{w}|\boldsymbol{x}_*, \boldsymbol{Y}_H, \boldsymbol{X}_H\right)$ over all possible parameter values $\boldsymbol{w}$:

$$p\left(f_{\boldsymbol{x}_*}|\boldsymbol{x}_*, \boldsymbol{Y}_H, \boldsymbol{X}_H\right) = \int p\left(f_{\boldsymbol{x}_*}, \boldsymbol{w}|\boldsymbol{x}_*, \boldsymbol{Y}_H, \boldsymbol{X}_H\right) d\boldsymbol{w} = \int p\left(f_{\boldsymbol{x}_*}|\boldsymbol{x}_*, \boldsymbol{w}\right) p(\boldsymbol{w}|\boldsymbol{Y}_H, \boldsymbol{X}_H) d\boldsymbol{w}. \qquad \textbf{(2-42)}$$

Similar to the steps in (2-35)-(2-39), first the joint distribution $p\left(f_{\boldsymbol{x}_*}, \boldsymbol{w}|\boldsymbol{x}_*, \boldsymbol{Y}_H, \boldsymbol{X}_H\right)$ is calculated in order to solve (2-42). Employing the calculation rules in Appendix C.3, the marginal distribution is ([173]):

$$\begin{aligned} &p\left(f_{\boldsymbol{x}_*}|\boldsymbol{x}_*, \boldsymbol{Y}_H, \boldsymbol{X}_H\right) \\ &= N\begin{pmatrix} \bar{\boldsymbol{w}}^T \boldsymbol{\phi}_{\boldsymbol{x}_*} + \boldsymbol{\phi}_{\boldsymbol{x}_*}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H}\left[\boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\left(\boldsymbol{Y}_H - \bar{\boldsymbol{w}}^T \boldsymbol{\phi}_{X_H}\right)^T, \\ \boldsymbol{\phi}_{\boldsymbol{x}_*}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{\boldsymbol{x}_*} - \boldsymbol{\phi}_{\boldsymbol{x}_*}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H}\left[\boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1} \boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{\boldsymbol{x}_*} \end{pmatrix}. \end{aligned} \qquad \textbf{(2-43)}$$

In particular, the posterior preditive mean is

$$m(\boldsymbol{x}_*) = \bar{\boldsymbol{w}}^T \boldsymbol{\phi}_{\boldsymbol{x}_*} + \boldsymbol{\phi}_{\boldsymbol{x}_*}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H}\left[\boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\left(\boldsymbol{Y}_H - \bar{\boldsymbol{w}}^T \boldsymbol{\phi}_{X_H}\right)^T \qquad \textbf{(2-44)}$$

and the posterior predictive covariance is

$$\Sigma(\boldsymbol{x}_*, \boldsymbol{x}_*') = \boldsymbol{\phi}_{\boldsymbol{x}_*}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{\boldsymbol{x}_*'} - \boldsymbol{\phi}_{\boldsymbol{x}_*}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H}\left[\boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1} \boldsymbol{\phi}_{X_H}^T \boldsymbol{\Sigma}_w \boldsymbol{\phi}_{\boldsymbol{x}_*'}. \qquad \textbf{(2-45)}$$

The following example is intended to provide some basic insights into the regression process.

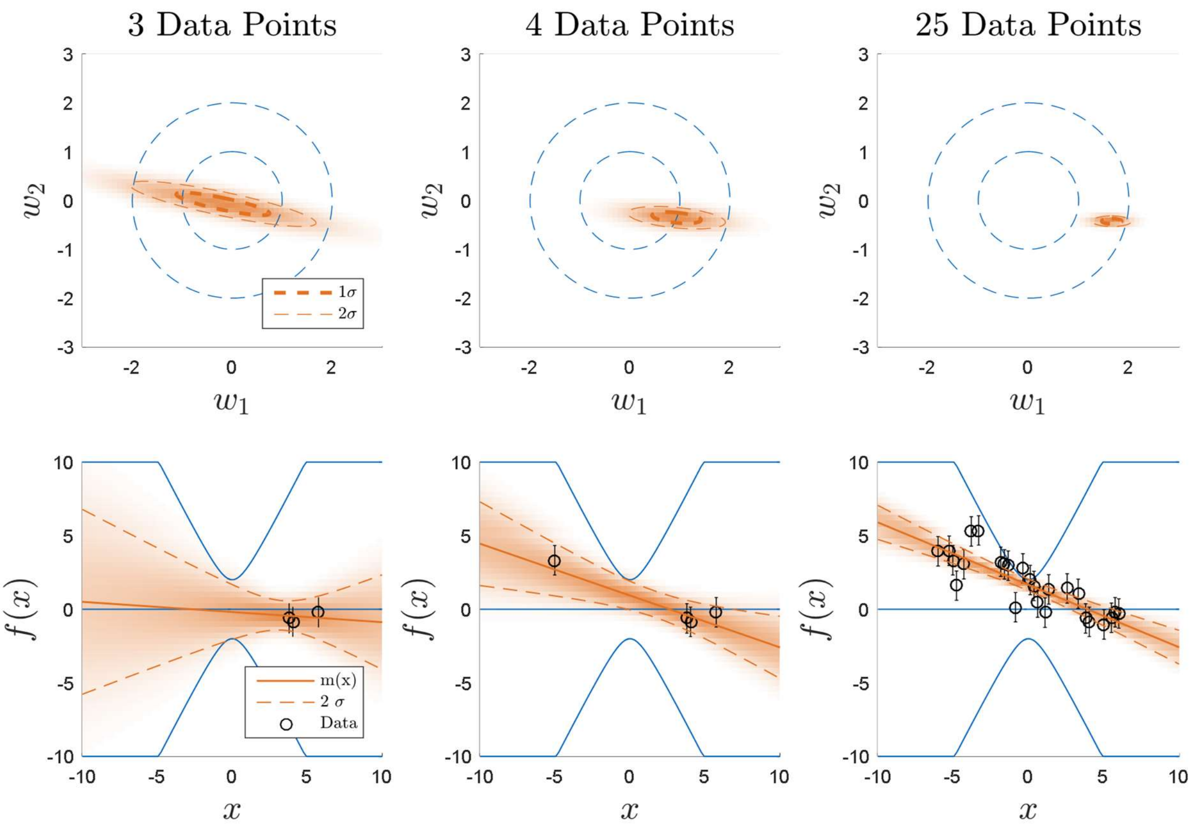


Figure 2-5: Example for GP regression with a linear model (based on [91]).

Example: Gaussian Process Regression

Consider data points $(\boldsymbol{X}_H, \boldsymbol{Y}_H)$ which are consecutively drawn from

$$f(x) = 1 - 0.4x,$$
$$y(x) = f(x) + \varepsilon. \tag{2-46}$$

The output y is corrupted by i.i.d. Gaussian noise with variance $\sigma_n^2 = 0.56$. Consider a zero-mean multivariate Gaussian prior $p(\boldsymbol{w}) = N(\boldsymbol{w}|\boldsymbol{0}, \boldsymbol{I})$ and the linear regression model with $\boldsymbol{w} = [w_1 \quad w_2]^T$ and $\boldsymbol{\phi} = [1 \quad x]^T$.

Figure 2-5 shows an exemplary result of GP regression. The top and bottom row depict the regression outcome in weight and function space, respectively. The blue lines denote the initial mean and standard deviation constituted by the prior distribution on the weights. The orange lines denote the mean and standard deviation imposed by the posterior distribution.

Figure 2-5 (left) shows the results after the observation of 3 data points, which lie in close proximity of each other. The initial guess of the mean changed only slightly. However, the variance decreased significantly in the vicinity of the data points. This constitutes one key property of GPs, namely that the regression quality is dependent on the richness of the observed data. The variance is drastically reduced in data rich areas, whereas the variance remains high in data scarce areas.

In the next step, a 4th data point is added, which significantly differs from the previous ones. Figure 2-5 (middle) shows the result after the observation of the 4th data point. It can be seen that the inference on 4 points decreases the uncertainty in w_1 drastically as the probable area for the parameters shrinks.

Finally, Figure 2-5 (right) shows the result for 25 data points. From the observation of a sufficient number of data points, a belief on the parameters is inferred, exhibiting a small variance. Adding further data points has only minor influence on the regression result.

2.2.1.2 Gaussian Process Regression – Function Space View and Kernel formulation

As already shown implicitly in Figure 2-5, an alternative and equal way to discuss GPs is to perform inference directly in function space. Compared to the previous approach, the new goal is to infer a belief directly on the function $f(\boldsymbol{x})$ based on the observed data $(\boldsymbol{X}_H, \boldsymbol{Y}_H)$. Let $\boldsymbol{f} \in \mathbb{R}^{1\times p}$ with $\boldsymbol{f} = \left[f(\boldsymbol{x}_1) \quad \dots \quad f(\boldsymbol{x}_p)\right]$ be a vector of function values. Applying Bayes' Theorem yields

$$p(\boldsymbol{f}|\boldsymbol{Y}_H, \boldsymbol{X}_H) = \frac{p(\boldsymbol{Y}_H|\boldsymbol{f})p\left(f(\boldsymbol{x})\right)}{p(\boldsymbol{Y}_H|\boldsymbol{X}_H)}. \tag{2-47}$$

The prior $p\left(f(\boldsymbol{x})\right)$ is found by multiplying $p(\boldsymbol{w})$ with the vector of feature functions $\boldsymbol{\phi}$ and is given by $p\left(f(\boldsymbol{x})\right) = N\left(\boldsymbol{\phi}^T(\boldsymbol{x})\bar{\boldsymbol{w}}, \boldsymbol{\phi}^T(\boldsymbol{x})\boldsymbol{\Sigma}_w\boldsymbol{\phi}(\boldsymbol{x})\right)$. The likelihood function $p(\boldsymbol{Y}_H|\boldsymbol{f})$ is equal to (2-33). Following steps (2-35)-(2-39) yields the posterior mean

$$m_{post}(\boldsymbol{X}_H) = \bar{\boldsymbol{w}}^T\boldsymbol{\phi}_{\boldsymbol{X}_H} + \boldsymbol{\phi}_{\boldsymbol{X}_H}^T\boldsymbol{\Sigma}_w\boldsymbol{\phi}_{\boldsymbol{X}_H}\left[\boldsymbol{\phi}_{\boldsymbol{X}_H}^T\boldsymbol{\Sigma}_w\boldsymbol{\phi}_{\boldsymbol{X}_H} + \sigma_n^2\boldsymbol{I}\right]^{-1}\left(\boldsymbol{Y}_H - \bar{\boldsymbol{w}}^T\boldsymbol{\phi}_{\boldsymbol{X}_H}\right)^T \tag{2-48}$$

and the posterior variance

$$\Sigma_{post}(\boldsymbol{X}_H, \boldsymbol{X}_H) = \boldsymbol{\phi}_{\boldsymbol{X}_H}^T\boldsymbol{\Sigma}_w\boldsymbol{\phi}_{\boldsymbol{X}_H} - \boldsymbol{\phi}_{\boldsymbol{X}_H}^T\boldsymbol{\Sigma}_w\boldsymbol{\phi}_{\boldsymbol{X}_H}\left[\boldsymbol{\phi}_{\boldsymbol{X}_H}^T\boldsymbol{\Sigma}_w\boldsymbol{\phi}_{\boldsymbol{X}_H} + \sigma_n^2\boldsymbol{I}\right]^{-1}\boldsymbol{\phi}_{\boldsymbol{X}_H}^T\boldsymbol{\Sigma}_w\boldsymbol{\phi}_{\boldsymbol{X}_H}. \tag{2-49}$$

The posterior predictive distribution takes on the same form as in (2-43). Despite being equal to the weight space view, performing inference in function space gives rise to the so-called kernel description. The kernel formulation is particularly useful in situations where it is easier to compute the kernel instead of the feature function (see e.g. [22]). Definition 2.3 is commonly used to define a kernel function.

Definition 2.3 – (Mercer) Kernel (based on [59, p. 8])

A *(Mercer) kernel* (also called a covariance function, kernel function, or covariance kernel) $k(x, x')$ is a continuous, symmetric, positive definite function of the two inputs x, x'.

Define the kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$ to be

$$k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}_{\boldsymbol{x}}^T\boldsymbol{\Sigma}_w\boldsymbol{\phi}_{\boldsymbol{x}'}. \tag{2-50}$$

Note that (2-50) resembles the covariance in (2-23). Hence, $k(\boldsymbol{x}, \boldsymbol{x}')$ is often called *covariance kernel*. The feature functions $\boldsymbol{\phi}(\cdot)$ form an inner product with respect to $\boldsymbol{\Sigma}_w$. Since $\boldsymbol{\Sigma}_w$ is positive definite, $k(\boldsymbol{x}, \boldsymbol{x}')$ satisfies Definition 2.3. For the cause of better readability, subscripts for the kernel function will resemble the argument, i.e. $k_{\boldsymbol{x}\boldsymbol{x}'} = k(\boldsymbol{x}, \boldsymbol{x}')$. Using the kernel formulation, for the posterior distribution of (2-48)-(2-49) yields

$$m_{post}(\boldsymbol{X}_H) = \bar{\boldsymbol{w}}^T\boldsymbol{\phi}_{\boldsymbol{X}_H} + \boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H}\left[\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H} + \sigma_n^2\boldsymbol{I}\right]^{-1}\left(\boldsymbol{Y}_H - \bar{\boldsymbol{w}}^T\boldsymbol{\phi}_{\boldsymbol{X}_H}\right)^T, \tag{2-51}$$

and

$$\Sigma_{post}(\boldsymbol{X}_H, \boldsymbol{X}_H) = \boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H} - \boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H}\left[\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H} + \sigma_n^2\boldsymbol{I}\right]^{-1}\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H}. \tag{2-52}$$

Similarly, the posterior predictive distribution of (2-43)-(2-45) can be restated with the kernel notation. In particular, the posterior predictive mean is

$$m(\boldsymbol{x}_*) = \overline{\boldsymbol{w}}^T\boldsymbol{\phi}_{\boldsymbol{x}_*} + \boldsymbol{k}_{\boldsymbol{x}_*\boldsymbol{X}_H}\left[\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H} + \sigma_n^2\boldsymbol{I}\right]^{-1}\left(\boldsymbol{Y}_H - \overline{\boldsymbol{w}}^T\boldsymbol{\phi}_{\boldsymbol{X}_H}\right)^T \tag{2-53}$$

and the posterior predictive covariance is

$$\Sigma(\boldsymbol{x}_*, \boldsymbol{x}_*') = \boldsymbol{k}_{\boldsymbol{x}_*\boldsymbol{x}_*'} - \boldsymbol{k}_{\boldsymbol{x}_*\boldsymbol{X}_H}\left[\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H} + \sigma_n^2\boldsymbol{I}\right]^{-1}\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{x}_*'}. \tag{2-54}$$

A multitude of different functions can be used as kernels. The following example introduces a small selection. Figure 2-6 exemplarily shows three kernel functions, i.e. the polynomial kernel with degree 4, the *squared exponential* (SE) kernel and the periodic kernel. The first row showes the respective kernel functions for scalar inputs.

The second row depicts a draw from the GP $f(x) \sim GP\left(0, \boldsymbol{k}(x, x')\right)$ with its mean and standard deviation evaluated at 100 evenly distributed points in the set $x \in [-10,10]$. Note that the probability density function of the prior $p\left(f(\boldsymbol{x})\right)$ is depicted as a color map.

The last row shows the covariance matrix $\boldsymbol{k}(x, x')$ evaluated for the same set. The covariance matrix indicates the dependency between points and provides insight into the nature of the GP. For example, for the SE kernel, points which are spatially close are highly dependent. In contrast, spatially distant points only exhibit small dependence. Hence, the function values at two spatially distant points may vary strongly from each other. The effect is even more distinct for the periodic kernel. Its covariance matrix forms a repetitive pattern of dependence and independence, thus leading to a periodic function.

This thesis predominantly uses the SE kernel, since it possesses properties important for the forecasting algorithm discussed in Chapter 4.3. The SE kernel is given by

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2l^2}\right), \tag{2-55}$$

where the free parameter l defines the bandwidth of the kernel. Techniques to optimize the parameters based on gathered data are for example available in [173] and [21]. The interested reader is also referred to [59] for more information on kernel functions and their combinations.

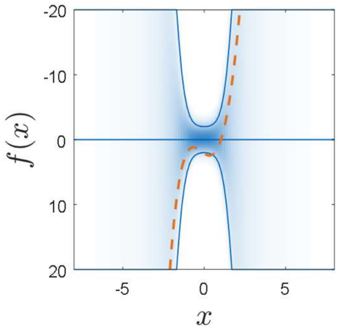
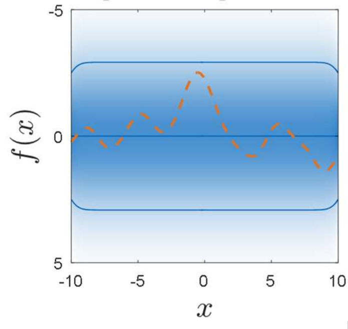
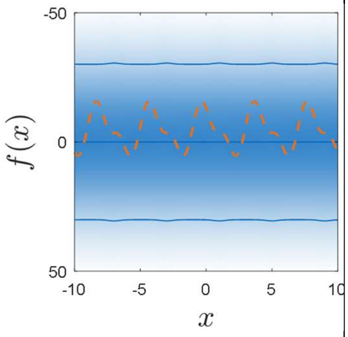
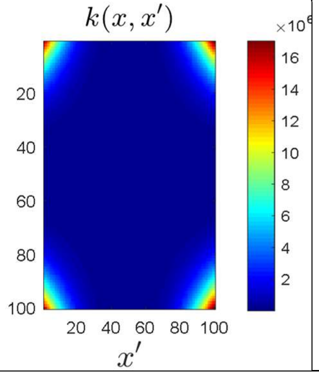
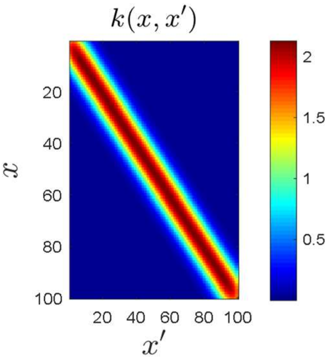
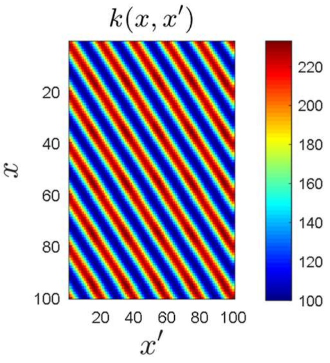
Type	Quartic Polynomial	Squared Exponential	Periodic
Equation	$\phi(x) = [x^0 \quad \dots \quad x^4]^T$ $k(x,x') = \phi^T(x)\Sigma\phi(x')$	$k(x,x') = \exp\left(-\frac{(x-x')^2}{2l^2}\right)$	$k(x,x')$ $= \exp\left(-\frac{2}{l^2}\sin^2\left(\pi\frac{x-x'}{p}\right)\right)$
Draw from the GP			
Covariance function			

Figure 2-6: Three examples for kernel functions - 4th order polynomial kernel (left), squared exponential kernel (middle) and periodic kernel (right)

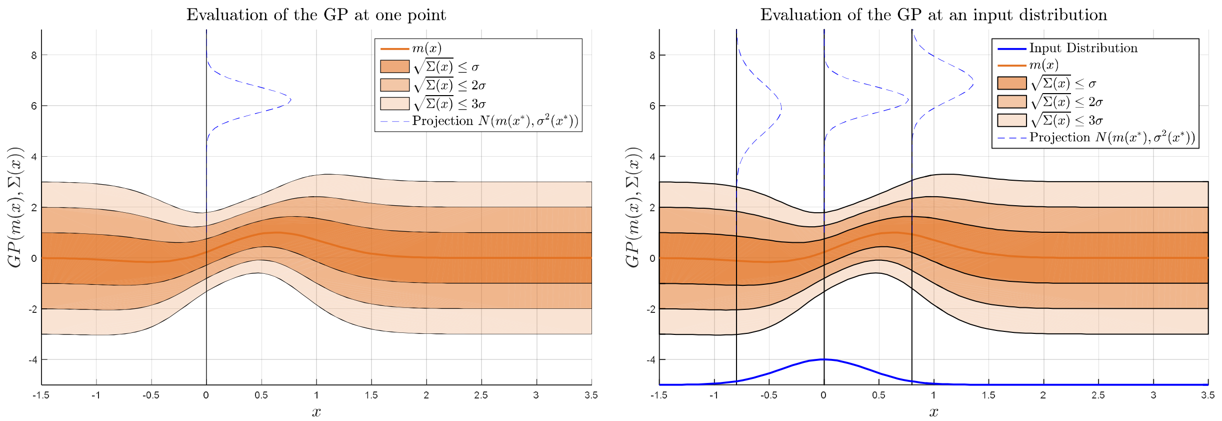


Figure 2-7: Evaluation of a GP at a single point and at an input distribution

2.3 Gaussian Process Evaluation at an uncertain input

By using (2-53) and (2-54), the evaluation of a GP at a single point $\boldsymbol{x}_*$ yields the parameters for the normal distribution of $f(\boldsymbol{x}_*)$ at said point. Until now, the test point $\boldsymbol{x}_*$ was assumed to be deterministic. However, in real world applications the test point often represents states of a system. Knowledge about system states originates from sensor measurements, which are often corrupted by noise. Hence, these applications require the evaluation of the GP model not at a single point but at an input distribution instead. Note, that sensor noise is often well represented by additive Gaussian white noise. Hence, this thesis only considers normally distributed $\boldsymbol{x}_*$.

Figure 2-7 shows the difference between the GP evaluation at a single point and at an input distribution. In the former case, the evaluation of the GP model yields a normal distribution for the variable modelled by the GP. Figure 2-7 (left) exemplarily shows the projected distribution of $f(\boldsymbol{x}_*)$ evaluated at $\boldsymbol{x}_* = 0$. In contrast, Figure 2-7 (right) depicts the case with input distribution $\boldsymbol{x}_* \sim N(0, 0.4^2)$. In order to facilitate understanding, the projected distributions of $f(\boldsymbol{x}_*)$ are exemplarily depicted at the mean $\mu = 0$ as well as at $\pm 2\sigma$. It can be seen, that while every projection in itself is a normal distributions, their parameters vary dependent on the considered point. Thus, the evaluation of the GP at an input distribution would require the combination of the projections of each possible test point weighted by its probability of occurrence. This leads to two general conclusions. First, evaluating the GP only at the mean of the input distribution leads to significant deviations from the real output, since the statistical nature of the input is neglected. Secondly, the resulting distribution of $f(\boldsymbol{x}_*)$ is most likely non-normal.

Consider the GP-model in (2-26). Evaluating the latter implies finding the marginal distribution $p(f_*)$, where $f_* = f(\boldsymbol{x}_*)$ for simplicity. In the case of a single deterministic input $\boldsymbol{x}_*$, the solution is given by (2-53) and (2-54). The challenge here is to find the marginal $p(f_*)$ if the input $\boldsymbol{x}_*$ is a random variable with the distribution $\boldsymbol{x}_* \sim N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$. In order to solve this problem, the joint distribution $p(f_*, \boldsymbol{x}_*)$ needs to be marginalized over the input $\boldsymbol{x}_*$:

$$p(f_*) = \int p(f_*, \boldsymbol{x}_*)\, d\boldsymbol{x}_* = \int p(f_* | \boldsymbol{x}_*) p(\boldsymbol{x}_*) d\boldsymbol{x}_*. \tag{2-56}$$

Often, solving (2-56) is analytically intractable. One way to marginalize $p(f_*, \boldsymbol{x}_*)$ is to use Monte-Carlo techniques for numerical approximation. However, the latter relies on sampling, which often demands the availability of large computational resources. Furthermore, the number of samples required to achieve a good fit may not be known a-priori.

Another approach is to approximate $p(f_*)$ as a normal distribution. In fact, the authors in [171], [71], [53] showed that if the employed kernel function in (2-50) is of the squared exponential type, the first and second moment of $p(f_*)$ can be computed analytically and exactly. In [28], Quiñonero-Candela mentions that the approach can also be applied for polynomial kernel functions. Ghassemi extended the technique to include periodic kernel functions ([71]). Amongst others, this technique has been applied for prediction ([85]) and filtering problems ([53]).

Define the mean and covariance of $p(f_*)$ to be represented by μ_f and Σ_f, respectively. The following sections depict how to compute μ_f and Σ_f analytically by leveraging the results of [171]. The sections are divided based on the employed kernel and whether a GP prior or regression model is considered. Note that only cases, which are important for this thesis, are considered. In particular, section 2.3.1 shows how to compute μ_f and Σ_f based on a prior GP model and a polynomial kernel function. In contrast, section 2.3.2 shows how to compute μ_f and Σ_f based on a GP regression model with squared exponential kernel functions.

2.3.1 Evaluation of a GP prior model with linear kernel functions at an uncertain input

Consider a prior GP model (2-26) with mean function (2-22) and covariance function (2-23). Furthermore, consider linear regressor functions $\boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{x}$ and a linear kernel of the form $k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^T \boldsymbol{\Sigma}_w \boldsymbol{x}'$. The derivation of the mean μ_f and covariance Σ_f are based on the law of iterated expectations and the law of total variance (see Appendix B.6).

Derivation of the mean μ_f

Applying the law of iterated expectations (see Appendix B.6) to (2-56) yields

$$\mu_f = \mathbb{E}_{\boldsymbol{x}_*}\left[\mathbb{E}_{f_*}[f_* | \boldsymbol{x}_*]\right]. \tag{2-57}$$

Remember, that the expectation of a GP is it's mean function $m(\boldsymbol{x})$. Hence, (2-57) becomes

$$\mu_f = \mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)]. \tag{2-58}$$

Employing the definition of the expected value operator leads to

$$\mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)] = \int m(\boldsymbol{x}_*) N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_*. \tag{2-59}$$

Inserting the mean of the prior model in (2-22) results in

$$\mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)] = \int \bar{\boldsymbol{w}}^T \boldsymbol{x}_* N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_* = \bar{\boldsymbol{w}}^T \underbrace{\int \boldsymbol{x}_* N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_*}_{\mathbb{E}_{\boldsymbol{x}_*}[\boldsymbol{x}_*]} = \bar{\boldsymbol{w}}^T \boldsymbol{\mu}_x. \tag{2-60}$$

Derivation of the covariance Σ_f

Similarly to the derivation of the mean, applying the law of the total variance (see Appendix B.6) to (2-56) yields

$$\Sigma_f = \mathbb{V}_{\boldsymbol{x}_*}\left[\mathbb{E}_{f_{\boldsymbol{x}_*}}\left[f_{\boldsymbol{x}_*}|\boldsymbol{x}_*\right]\right] + \mathbb{E}_{\boldsymbol{x}_*}\left[\mathbb{V}_{f_{\boldsymbol{x}_*}}\left[f_{\boldsymbol{x}_*}|\boldsymbol{x}_*\right]\right]. \tag{2-61}$$

Again, remember that the expectation of a GP is its mean function. Hence, $\mathbb{V}_{\boldsymbol{x}_*}\left[\mathbb{E}_{f_{\boldsymbol{x}_*}}\left[f_{\boldsymbol{x}_*}|\boldsymbol{x}_*\right]\right] = \mathbb{V}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)]$.

$$\mathbb{V}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)] = \mathbb{E}_{\boldsymbol{x}_*}\left[\left(m(\boldsymbol{x}_*) - \mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)]\right)\left(m(\boldsymbol{x}_*) - \mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)]\right)^T\right]. \tag{2-62}$$

Inserting (2-22) and (2-60) and rearranging terms yields

$$\begin{aligned} \mathbb{V}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)] &= \mathbb{E}_{\boldsymbol{x}_*}[(\bar{\boldsymbol{w}}^T\boldsymbol{x}_* - \bar{\boldsymbol{w}}^T\boldsymbol{\mu}_x)(\bar{\boldsymbol{w}}^T\boldsymbol{x}_* - \bar{\boldsymbol{w}}^T\boldsymbol{\mu}_x)^T] \\ &= \bar{\boldsymbol{w}}^T \underbrace{\mathbb{E}_{\boldsymbol{x}_*}[(\boldsymbol{x}_* - \boldsymbol{\mu}_x)(\boldsymbol{x}_* - \boldsymbol{\mu}_x)^T]}_{\mathbb{V}_{\boldsymbol{x}_*}[\boldsymbol{x}_*]=\boldsymbol{\Sigma}_x} \bar{\boldsymbol{w}} = \bar{\boldsymbol{w}}^T\boldsymbol{\Sigma}_x\bar{\boldsymbol{w}}. \end{aligned} \tag{2-63}$$

By using the variance of the GP prior model in (2-23), the second term in (2-61) becomes

$$\mathbb{E}_{\boldsymbol{x}_*}\left[\mathbb{V}_{f_{\boldsymbol{x}_*}}\left[f_{\boldsymbol{x}_*}|\boldsymbol{x}_*\right]\right] = \mathbb{E}_{\boldsymbol{x}_*}[\Sigma(\boldsymbol{x}_*, \boldsymbol{x}_*)] = \mathbb{E}_{\boldsymbol{x}_*}[\boldsymbol{x}_*^T\boldsymbol{\Sigma}_w\boldsymbol{x}_*]. \tag{2-64}$$

Remember that $\boldsymbol{\Sigma}_x$ is symmetric and positive definite. The solution for the expectation of the quadratic form $\mathbb{E}_{\boldsymbol{x}_*}[\boldsymbol{x}_*^T\boldsymbol{\Sigma}_w\boldsymbol{x}_*]$ is ([193, p. 9], Appendix B.10)

$$\mathbb{E}_{\boldsymbol{x}_*}\left[\mathbb{V}_{f_{\boldsymbol{x}_*}}\left[f_{\boldsymbol{x}_*}|\boldsymbol{x}_*\right]\right] = \mathbb{E}_{\boldsymbol{x}_*}[\boldsymbol{x}_*^T\boldsymbol{\Sigma}_w\boldsymbol{x}_*] = \mathrm{Tr}(\boldsymbol{\Sigma}_w\boldsymbol{\Sigma}_x) + \boldsymbol{\mu}_x^T\boldsymbol{\Sigma}_w\boldsymbol{\mu}_x. \tag{2-65}$$

Hence, equation (2-61) becomes

$$\Sigma_f = \bar{\boldsymbol{w}}^T\boldsymbol{\Sigma}_x\bar{\boldsymbol{w}} + \mathrm{Tr}(\boldsymbol{\Sigma}_w\boldsymbol{\Sigma}_x) + \boldsymbol{\mu}_x^T\boldsymbol{\Sigma}_w\boldsymbol{\mu}_x. \tag{2-66}$$

The following example is intended to give more insight into the approximation of $p(f_*)$ as a normal distribution.

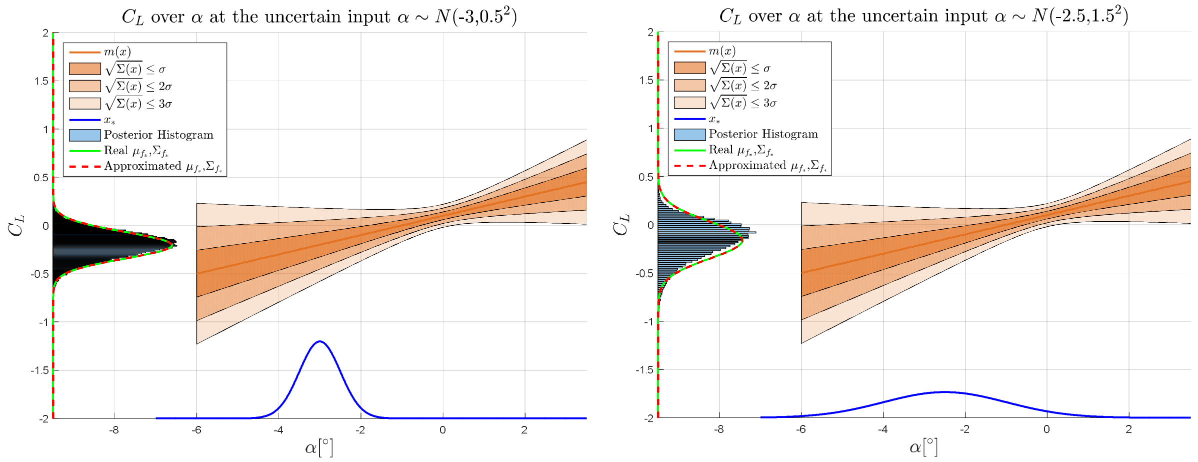


Figure 2-8: C_L over α modeled by a GP and evaluated at an input distribution

Example 2-3: Evaluation of a GP prior model with linear kernels at an uncertain input

Consider the C_L over α example in Example 2-2, which is modeled by a GP in (2-28). Instead of calculating C_L at a fixed input α, the angle of attack is assumed to be a rv itself. This example considers two scenarios. First, the GP shall be evaluated at the input distribution $\alpha_1 \sim N(-3, 0.5^2)$. Secondly, the input is changed to $\alpha_2 \sim N(-2.5, 1.5^2)$. Equations (2-60) and (2-66) are used for the exact analytical calculation of the mean μ_f and the covariance Σ_f. In order to build a reference, 50000 samples are drawn from α and the weights w and propagated through the model (2-28).

Figure 2-8 shows the results. The distribution of the input α is plotted on the x-axis on the bottom. The GP model (2-28) is depicted by the mean as well as the area representing 1,2 and 3 standard deviations, respectively. The bar graph on the y-axis represents the propagation results of the 50000 samples. Furthermore, the first and second moment of the propagated samples is computed. The solid line on the y-axis represents a normal distribution with the computed mean and covariance. Finally, the dashed line on the y-axis represents the approximating normal distribution by employing equations (2-60) and (2-66).

Figure 2-8 (left) shows the first case with $\alpha_1 \sim N(-3, 0.5^2)$. It can be seen that the distribution formed by the parameters of the propagated samples is nearly indistinguishable from the analytical calculation with equations (2-60) and (2-66). While the first two moments of the propagated distribution are calculated exactly, the approach does not give any hint whether the latter is well approximated by a normal distribution. Figure 2-8 (right) depicts this case with $\alpha_2 \sim N(-2.5, 1.5^2)$. It can be seen that the histogram of the propagated samples clearly forms a non-normal distribution. Still, the first two moments of this distribution can be calculated and used as the parameters of a normal distribution. The calculation rules of (2-60) and (2-66) then indeed result in the same parameters. However, the histogram is still not represented well. Considering the error between the approximation and the real distribution is subject to section 4.5.2.

2.3.2 Evaluation of a GP with squared exponential kernels at an uncertain input

Consider a prior GP model (2-26) with mean function (2-22) and covariance function (2-23). Furthermore, consider the SE kernel of (2-55). This thesis considers SE kernels for online regression problems only. It therefore assumes the prior distribution to have zero-mean. This is valid, since all physical information is already incorporated in the physical model. The derivation of the mean μ_f and covariance Σ_f are again based on the law of iterated expectations and the law of total variance (see Appendix B.6). The following approach is depicted in [171], [71] and [53].

Derivation of the mean μ_f

Similar to the linear kernel the approach facilitates the law of iterated expectations:

$$\mu_f = \mathbb{E}_{\boldsymbol{x}_*}\left[\mathbb{E}_{f_*}[f_*|\boldsymbol{x}_*]\right]. \qquad \textbf{(2-67)}$$

Again, the expectation of a GP is its mean function, such that $\mu_f = \mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)]$. The posterior predictive mean $m(\boldsymbol{x}_*)$ is given in (2-53). Considering a zero-mean prior and inserting (2-53) into (2-67) leads to

$$\mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)] = \int \boldsymbol{k}_{\boldsymbol{x}_*\boldsymbol{X}_H}\left[\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H} + \sigma_n^2\boldsymbol{I}\right]^{-1}\boldsymbol{Y}_H N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)d\boldsymbol{x}_*. \qquad \textbf{(2-68)}$$

For simplicity, define $\boldsymbol{B}_{\boldsymbol{X}_H\boldsymbol{Y}_H} = \left[\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H} + \sigma_n^2\boldsymbol{I}\right]^{-1}\boldsymbol{Y}_H^T$. Note that $\boldsymbol{B}_{\boldsymbol{X}_H\boldsymbol{Y}_H}$ does not depend on the predictive state $\boldsymbol{x}_*$, but only on the stored data. Therefore, it can be moved in front of the integral by first transposing $m(\boldsymbol{x}_*) = \boldsymbol{k}_{\boldsymbol{x}_*\boldsymbol{X}_H}\boldsymbol{B}_{\boldsymbol{X}_H\boldsymbol{Y}_H}$:

$$\mathbb{E}_{\mathbf{x}_*}[m(\boldsymbol{x}_*)] = \boldsymbol{B}_{\boldsymbol{X}_H\boldsymbol{Y}_H}^T \int \boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{x}_*} N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)d\boldsymbol{x}_*. \qquad \textbf{(2-69)}$$

Note, that $\boldsymbol{k}_{\boldsymbol{x}_*\boldsymbol{X}_H}^T = \boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{x}_*}$. The trick used in [171] in order to solve (2-69) is to note that the equation of a squared exponential kernel $\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{x}_*}$ is similar to that of a normal probability distribution. To see this, define $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times n}$ to contain the bandwidth of each kernel function on its main diagonal and expand the kernel matrix $\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{x}_*}$ for each i –th stored data point $\boldsymbol{X}_{H_i}$:

$$\boldsymbol{k}_{\boldsymbol{X}_{H_i}\boldsymbol{x}_*} = \frac{(2\pi)^{\frac{n}{2}}|\boldsymbol{\Lambda}|^{\frac{1}{2}}}{(2\pi)^{\frac{n}{2}}|\boldsymbol{\Lambda}|^{\frac{1}{2}}}\boldsymbol{k}_{\boldsymbol{X}_{H_i}\boldsymbol{x}_*} = (2\pi)^{\frac{n}{2}}|\boldsymbol{\Lambda}|^{\frac{1}{2}}\underbrace{\frac{1}{(2\pi)^{\frac{n}{2}}|\boldsymbol{\Lambda}|^{\frac{1}{2}}}e^{-\frac{1}{2}\left(\left(\boldsymbol{x}_*-\boldsymbol{X}_{H_i}\right)^T\boldsymbol{\Lambda}^{-1}\left(\boldsymbol{x}_*-\boldsymbol{X}_{H_i}\right)\right)}}_{N\left(\boldsymbol{X}_{H_i},\boldsymbol{\Lambda}\right)} \qquad \textbf{(2-70)}$$

$$= (2\pi)^{\frac{n}{2}}|\boldsymbol{\Lambda}|^{\frac{1}{2}}N\left(\boldsymbol{X}_{H_i}, \boldsymbol{\Lambda}\right).$$

Hence, $(2\pi)^{-\frac{n}{2}}|\boldsymbol{\Lambda}|^{-\frac{1}{2}}\boldsymbol{k}_{\boldsymbol{X}_{H_i}\boldsymbol{x}_*}$ as a function of $\boldsymbol{x}_*$ forms a normal distribution with mean $\boldsymbol{X}_{H_i}$ and covariance $\boldsymbol{\Lambda}$. The trick uses this reformulation in order to employ the same calculation rules as for a sum of i.i.d. normally distributed random variables.

Following the calculation rules in Appendix C.4, the multiplication of two normal distributions $N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$ and $N(\boldsymbol{X}_{H_i}, \boldsymbol{\Lambda})$ yields

$$N(\boldsymbol{X}_{H_i}, \boldsymbol{\Lambda}) \cdot N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$$

$$= \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Lambda} + \boldsymbol{\Sigma}_x|^{\frac{1}{2}}} e^{-\frac{1}{2}\left(\left(\mu_x - X_{H_i}\right)^T (\boldsymbol{\Lambda}+\boldsymbol{\Sigma}_x)^{-1} \left(\mu_x - X_{H_i}\right)\right)} N(\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3). \qquad \textbf{(2-71)}$$

It is not necessary to calculate the pdf $N(\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$ explicitly, since it will be integrated out in the next step. The i-th solution l_i for the integral in (2-69) is obtained by inserting (2-71) into (2-69):

$$l_i = \int \boldsymbol{k}_{X_{H_i} x_*} N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_*$$

$$= \frac{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Lambda}|^{\frac{1}{2}}}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Lambda} + \boldsymbol{\Sigma}_x|^{\frac{1}{2}}} e^{-\frac{1}{2}\left(\left(\mu_x - X_{H_i}\right)^T (\boldsymbol{\Lambda}+\boldsymbol{\Sigma}_x)^{-1} \left(\mu_x - X_{H_i}\right)\right)} \underbrace{\int N(\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3) d\boldsymbol{x}_*}_{=1} \qquad \textbf{(2-72)}$$

$$= |\boldsymbol{I} + \boldsymbol{\Lambda}^{-1}\boldsymbol{\Sigma}_x|^{-\frac{1}{2}} e^{-\frac{1}{2}\left(\left(\mu_x - X_{H_i}\right)^T (\boldsymbol{\Lambda}+\boldsymbol{\Sigma}_x)^{-1} \left(\mu_x - X_{H_i}\right)\right)}.$$

Combining all i solutions in the vector $\boldsymbol{l}$ results in a solution for (2-69):

$$\mu_f = \mathbb{E}_{\mathbf{x}_*}[m(\boldsymbol{x}_*)] = \boldsymbol{B}_{X_H Y_H}^T \boldsymbol{l}. \qquad \textbf{(2-73)}$$

The following part deals with the derivation of the covariance Σ_f.

Derivation of the covariance Σ_f

Similarly to the derivation of the mean, applying the law of the total variance (see Appendix B.6) to (2-56) yields

$$\Sigma_f = \mathbb{V}_{\boldsymbol{x}_*}\left[\mathbb{E}_{f_{\boldsymbol{x}_*}}[f_{\boldsymbol{x}_*} | \boldsymbol{x}_*]\right] + \mathbb{E}_{\boldsymbol{x}_*}\left[\mathbb{V}_{f_{\boldsymbol{x}_*}}[f_{\boldsymbol{x}_*} | \boldsymbol{x}_*]\right]. \qquad \textbf{(2-74)}$$

Again, remember that the expectation of a GP is its mean function. With the definition of the covariance operator, the first term in (2-74) resolves to

$$\mathbb{V}_{\boldsymbol{x}_*}\left[\mathbb{E}_{f_{\boldsymbol{x}_*}}[f_{\boldsymbol{x}_*} | \boldsymbol{x}_*]\right] = \mathbb{V}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)] = \mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*) m^T(\boldsymbol{x}_*)] - \mathbb{E}_{\boldsymbol{x}_*}^2[m(\boldsymbol{x}_*)]. \qquad \textbf{(2-75)}$$

With the application of the expected value operator, the first term in (2-75) becomes

$$\mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*) m^T(\boldsymbol{x}_*)] = \int m(\boldsymbol{x}_*) m^T(\boldsymbol{x}_*) N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_*. \qquad \textbf{(2-76)}$$

Inserting the posterior predictive mean $m(\boldsymbol{x}_*)$ of (2-53) and moving $\boldsymbol{B}_{X_H Y_H}$ outside of the integral yields

$$\mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*) m^T(\boldsymbol{x}_*)] = \boldsymbol{B}_{X_H Y_H}^T \int \boldsymbol{k}_{X_H x_*} \boldsymbol{k}_{x_* X_H} N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_* \, \boldsymbol{B}_{X_H Y_H}. \qquad \textbf{(2-77)}$$

In (2-77) the product of two kernel functions needs to be integrated. Again, the trick used in [171] forms the basis for solving (2-77). Remember the kernel function expansion in (2-70). Hence, the problem comes down to solving the multiplication of the three normal densities $N(\boldsymbol{X}_{H_i},\boldsymbol{\Lambda})$, $N\left(\boldsymbol{X}_{H_j},\boldsymbol{\Lambda}\right)$ and $N(\boldsymbol{\mu}_x,\boldsymbol{\Sigma}_x)$ for each pair of stored data points $\left(\boldsymbol{X}_{H_i},\boldsymbol{X}_{H_j}\right)$. The derivation is considerably easier if the two densities $N(\boldsymbol{X}_{H_i},\mathbf{\Lambda})$ and $N\left(\boldsymbol{X}_{H_j},\mathbf{\Lambda}\right)$ are multiplied first, since they share a common covariance matrix. Following the calculation rules in Appendix C.4, the multiplication of $N(\boldsymbol{X}_{H_i},\boldsymbol{\Lambda})$ and $N\left(\boldsymbol{X}_{H_j},\boldsymbol{\Lambda}\right)$ yields

$$\begin{aligned} &N(\boldsymbol{X}_{H_i},\mathbf{\Lambda})\cdot N\left(\boldsymbol{X}_{H_j},\mathbf{\Lambda}\right) \\ &= \frac{1}{(2\pi)^{\frac{n}{2}}|2\mathbf{\Lambda}|^{\frac{1}{2}}} e^{-\frac{1}{2}\left(\left(\boldsymbol{X}_{H_i}-\boldsymbol{X}_{H_j}\right)^T(2\mathbf{\Lambda})^{-1}\left(\boldsymbol{X}_{H_i}-\boldsymbol{X}_{H_j}\right)\right)} N(\boldsymbol{\mu}_3,\mathbf{\Sigma}_3), \end{aligned} \tag{2-78}$$

where

$$\begin{aligned} \boldsymbol{\mu}_3 &= \frac{1}{2}\left(\boldsymbol{X}_{H_i}-\boldsymbol{X}_{H_j}\right) \\ \mathbf{\Sigma}_3 &= \frac{1}{2}\mathbf{\Lambda}. \end{aligned} \tag{2-79}$$

Again, applying the calculation rules in Appendix C.4, the multiplication of all three densities yields

$$\begin{aligned} &N(\boldsymbol{X}_{H_i},\mathbf{\Lambda})\cdot N\left(\boldsymbol{X}_{H_j},\mathbf{\Lambda}\right)\cdot N\left(\boldsymbol{X}_{H_j},\mathbf{\Lambda}\right) \\ &= \frac{1}{(2\pi)^{\frac{n}{2}}|2\mathbf{\Lambda}|^{\frac{1}{2}}} e^{-\frac{1}{2}\left(\left(\boldsymbol{X}_{H_i}-\boldsymbol{X}_{H_j}\right)^T(2\mathbf{\Lambda})^{-1}\left(\boldsymbol{X}_{H_i}-\boldsymbol{X}_{H_j}\right)\right)} \\ &\cdot \frac{1}{(2\pi)^{\frac{n}{2}}\left|\frac{1}{2}\mathbf{\Lambda}+\mathbf{\Sigma}_x\right|^{\frac{1}{2}}} e^{-\frac{1}{2}\left((\boldsymbol{\mu}_x-\boldsymbol{\mu}_3)^T\left(\frac{1}{2}\mathbf{\Lambda}+\mathbf{\Sigma}_x\right)^{-1}(\boldsymbol{\mu}_x-\boldsymbol{\mu}_3)\right)}\cdot N(\boldsymbol{\mu}_4,\mathbf{\Sigma}_4). \end{aligned} \tag{2-80}$$

Again the parameters of the density $N(\boldsymbol{\mu}_4,\mathbf{\Sigma}_4)$ do not need to be calculated explicitly, since it integrates out in the next step. The ij-th solution $\boldsymbol{L}_{ij}$ for the integral in (2-77) is obtained by simplifying the expression above and inserting it into (2-77):

$$\begin{aligned} \boldsymbol{L}_{ij} &= \int \boldsymbol{k}_{X_H x_*}\boldsymbol{k}_{x_* X_H} N(\boldsymbol{\mu}_x,\mathbf{\Sigma}_x)d\boldsymbol{x}_* \\ &= \frac{1}{|2\mathbf{\Sigma}_x+\mathbf{I}|^{\frac{1}{2}}} e^{-\frac{1}{2}\left(\left(\boldsymbol{X}_{H_i}-\boldsymbol{X}_{H_j}\right)^T(2\mathbf{\Lambda})^{-1}\left(\boldsymbol{X}_{H_i}-\boldsymbol{X}_{H_j}\right)+(\boldsymbol{\mu}_x-\boldsymbol{\mu}_3)^T\left(\frac{1}{2}\mathbf{\Lambda}+\mathbf{\Sigma}_x\right)^{-1}(\boldsymbol{\mu}_x-\boldsymbol{\mu}_3)\right)} \underbrace{\int N(\boldsymbol{\mu}_4,\mathbf{\Sigma}_4)d\boldsymbol{x}_*}_{=1}. \end{aligned} \tag{2-81}$$

Hence, equation (2-76) becomes

$$\mathbb{E}_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)m^T(\boldsymbol{x}_*)] = \boldsymbol{B}_{X_H Y_H}^T \boldsymbol{L} \boldsymbol{B}_{X_H Y_H}. \tag{2-82}$$

The second term in (2-75) follows from leveraging the results of the derivation of the mean $\boldsymbol{\mu}_f$. Using (2-73), $\mathbb{E}^2_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)]$ becomes

$$\mathbb{E}^2_{\boldsymbol{x}_*}[m(\boldsymbol{x}_*)] = \left(\boldsymbol{B}^T_{X_H Y_H}\boldsymbol{l}\right)^2 = \boldsymbol{B}^T_{X_H Y_H}\boldsymbol{l}\boldsymbol{l}^T\boldsymbol{B}_{X_H Y_H}. \tag{2-83}$$

The second term in (2-74)

$$\mathbb{E}_{\boldsymbol{x}_*}\left[\mathbb{V}_{f_{\boldsymbol{x}_*}}\left[f_{\boldsymbol{x}_*}|\boldsymbol{x}_*\right]\right] = \mathbb{E}_{\boldsymbol{x}_*}[\boldsymbol{\Sigma}(\boldsymbol{x}_*)]. \tag{2-84}$$

Employing the expected value operator and using the predictive posterior covariance of (2-54), (2-84) becomes

$$\mathbb{E}_{\boldsymbol{x}_*}[\boldsymbol{\Sigma}(\boldsymbol{x}_*)] = \int \boldsymbol{k}_{\boldsymbol{x}_*\boldsymbol{x}_*} N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_* - \int \boldsymbol{k}_{\boldsymbol{x}_* X_H}\left[\boldsymbol{k}_{X_H X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\boldsymbol{k}_{X_H \boldsymbol{x}_*} N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_*. \tag{2-85}$$

Independent of the numerical value of the input $\boldsymbol{x}_*$, the SE kernel $\boldsymbol{k}_{\boldsymbol{x}_*\boldsymbol{x}_*}$ evaluates to 1. Hence, the solution of the first integral in (2-85) is

$$\int \boldsymbol{k}_{\boldsymbol{x}_*\boldsymbol{x}_*} N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_* = 1. \tag{2-86}$$

The second integral in (2-85) is solved by leveraging the properties of the trace operator (see Appendix A.1):

$$\begin{aligned} &\int \boldsymbol{k}_{\boldsymbol{x}_* X_H}\left[\boldsymbol{k}_{X_H X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\boldsymbol{k}_{X_H \boldsymbol{x}_*} N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_* \\ &= \int Tr\left[\left[\boldsymbol{k}_{X_H X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\boldsymbol{k}_{X_H \boldsymbol{x}_*}\boldsymbol{k}_{\boldsymbol{x}_* X_H}\right] N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_*. \end{aligned} \tag{2-87}$$

The structure of the integral is similar to the one in (2-81) and can be solved in an equal way by expanding the kernel functions and multiplying the three normal densities. Using the result in (2-81), (2-87) becomes

$$\int Tr\left[\left[\boldsymbol{k}_{X_H X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\boldsymbol{k}_{X_H \boldsymbol{x}_*}\boldsymbol{k}_{\boldsymbol{x}_* X_H}\right] N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) d\boldsymbol{x}_* = Tr\left[\left[\boldsymbol{k}_{X_H X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\boldsymbol{L}\right]. \tag{2-88}$$

Inserting (2-86) and (2-88) in (2-85) yields

$$\mathbb{E}_{\boldsymbol{x}_*}[\boldsymbol{\Sigma}(\boldsymbol{x}_*)] = 1 - Tr\left[\left[\boldsymbol{k}_{X_H X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\boldsymbol{L}\right]. \tag{2-89}$$

Finally, by inserting the results of (2-82), (2-83) and (2-89) in (2-74) the covariance Σ_f is

$$\Sigma_f = 1 + \boldsymbol{B}^T_{X_H Y_H}\boldsymbol{L}\boldsymbol{B}_{X_H Y_H} - \boldsymbol{B}^T_{X_H Y_H}\boldsymbol{l}\boldsymbol{l}^T\boldsymbol{B}_{X_H Y_H} - Tr\left[\left[\boldsymbol{k}_{X_H X_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}\boldsymbol{L}\right]. \tag{2-90}$$

The following example uses the introductory example of section 2.3 to demonstrate the evaluation of a GP with SE kernels at an uncertain input.

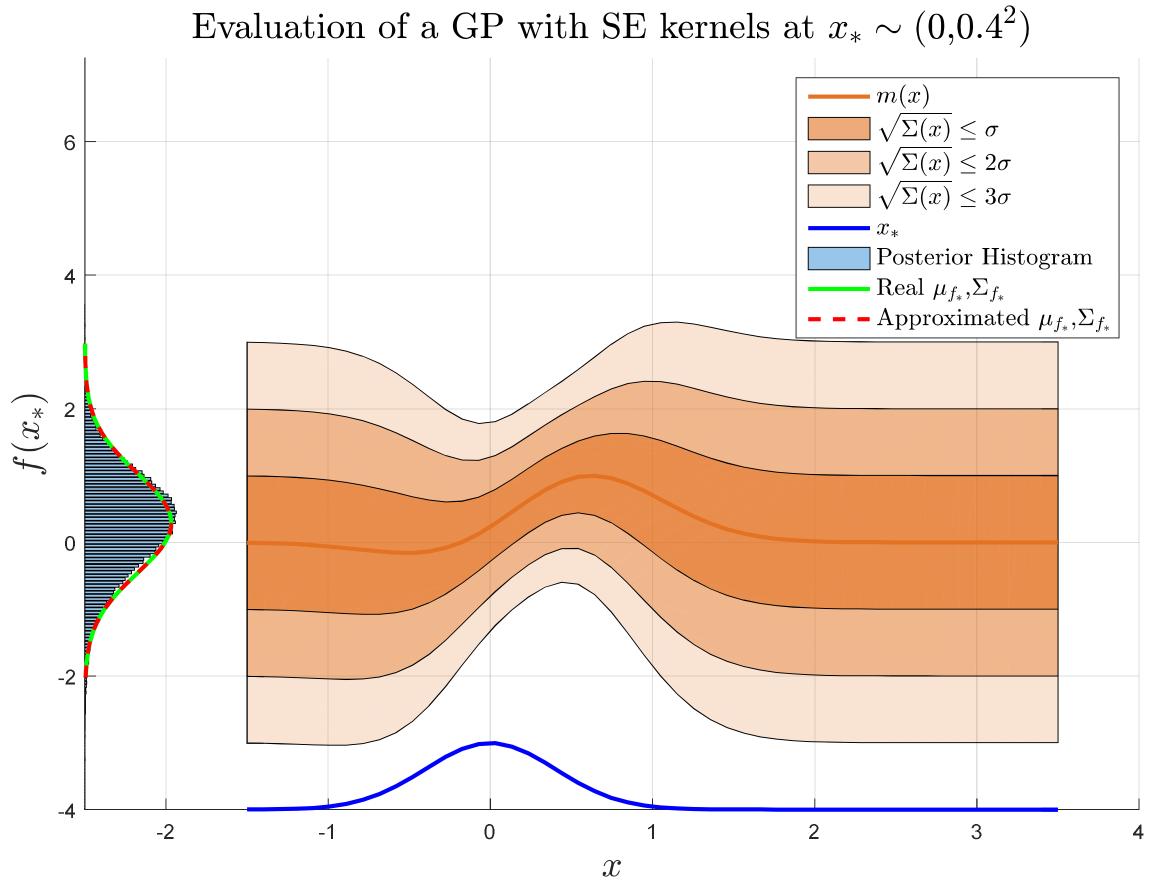


Figure 2-9: Evaluation of a GP with SE kernels at the uncertain input

Example 2-4: **Evaluation of a GP formed by SE kernels at an uncertain input**

Consider the generic introductory example of section 2.3. The GP model is selected to have a zero-mean prior distribution. The kernel functions are selected to be of the SE type as in (2-55) with bandwidth $l = 0.5$. The standard deviation of the zero-mean additive measurement noise is chosen to be $\sigma_n = 0.65$. Two data points are part of the history stacks, namely $\boldsymbol{X}_H = [0\ 0.5]^T$ and $\boldsymbol{Y}_H = [0\ 1.5]^T$. The input distribution is $x_* \sim N(0, 0.4^2)$. Again, in order to build a reference, 100000 samples are drawn from x_* and propagated through the GP.

Figure 2-9 shows the result of the evaluation of the GP model. The distribution of the input x is plotted on the x-axis on the bottom. The GP model is depicted by the mean as well as the area representing 1,2 and 3 standard deviations, respectively. The bar graph on the y-axis represents the propagation results of the 100000 samples. Furthermore, the first and second moment of the propagated samples are computed. The solid line on the y-axis represents a normal distribution with the computed mean and covariance. Finally, the dashed line on the y-axis represents the approximating normal distribution by employing equations (2-73) and (2-90).

As for the linear kernels, the distribution formed by the parameters of the propagated samples is nearly indistinguishable from the analytical calculation with equations (2-73) and (2-90). This demonstrates that the first two moments of $p(f_{x_*})$ can be exactly calculated by (2-73) and (2-90). Again, note that this approach does not relay information, whether $p(f_{x_*})$ is well approximated by a normal distribution in the first place.

3 System Description and Control Design

This section presents the system description and control architecture used throughout the thesis. The outline of this section is as follows: Section 3.1 defines the system, which is to be controlled. Section 3.2 introduces the baseline control architecture used throughout this thesis. Section 3.3 augments the control architecture by a MRAC. Finally, section 3.4 summarizes gaps, which prevent a successful certification of adaptive controllers.

3.1 System Definition

Let $\boldsymbol{x}_P \in D_x \subset \mathbb{R}^n$ be the state vector, let $\boldsymbol{u} \in \mathbb{R}^l$ be the control input and consider the following class of control-affine nonlinear dynamical systems:

$$\dot{\boldsymbol{x}}_P = \boldsymbol{f}(\boldsymbol{x}_P) + \boldsymbol{G}(\boldsymbol{x}_P)\boldsymbol{u} \tag{3-1}$$

The functions $\boldsymbol{f}(\boldsymbol{x}_P)$, with $\boldsymbol{f}(\boldsymbol{0}) = \boldsymbol{0}$, and $\boldsymbol{G}(\boldsymbol{x}_P)$ are fully or partially unknown, but assumed to be Lipschitz over D_x and $\boldsymbol{G}(\mathrm{x}_\mathrm{P})$ is assumed to be nonsingular for all $\boldsymbol{x}_P \in D_x$. The control input $\boldsymbol{u}$ is assumed to be piecewise continuous. The system states $\boldsymbol{x}_P$ are assumed to be available for full state feedback. Note that the class of systems in (3-1) can be extended to include non-affine systems if the input is replaceable by an invertible function (see i.a. [197]).

Directly designing a control algorithm for the system in (3-1) is seldom possible or advisable. Rather, transformations are sought which bring the dynamics into a form, which is easier to handle, without introducing additional simplifications. For the purpose of this thesis the existence of a transformation is assumed, which brings the system dynamics into the form

$$\dot{\boldsymbol{x}}_P = \boldsymbol{A}_P\boldsymbol{x}_P + \boldsymbol{B}_P\big(\boldsymbol{u} - \boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})\big). \tag{3-2}$$

Here, the system matrix $\boldsymbol{A}_P \in \mathbb{R}^{n\times n}$ and input matrix $\boldsymbol{B}_P \in \mathbb{R}^{n\times l}$ are known and the pair $(\boldsymbol{A}_P, \boldsymbol{B}_P)$ is controllable. The term $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$ represents the modelling uncertainty, which is a result of the system reformulation. For the sake of simplicity assume $(\boldsymbol{A}_P, \boldsymbol{B}_P)$ to be in controllable canonical form, i.e.:

$$\boldsymbol{A}_P = \begin{bmatrix} \boldsymbol{A}_{P,1} & 0 & \dots & \dots & 0 \\ 0 & \boldsymbol{A}_{P,2} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & \cdots & \boldsymbol{A}_{P,r} \end{bmatrix}, \boldsymbol{B}_P = \begin{bmatrix} \boldsymbol{b}_{P,1} & 0 & \dots & \dots & 0 \\ 0 & \boldsymbol{b}_{P,2} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & \cdots & \boldsymbol{b}_{P,r} \end{bmatrix}. \tag{3-3}$$

Here, the i-th submatrix of $\boldsymbol{A}_P$ and $\boldsymbol{B}_P$ is given by

$$\boldsymbol{A}_{P,i} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ \vdots & 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & 0 & 1 \\ 0 & 0 & \cdots & \cdots & 0 \end{bmatrix}, \boldsymbol{b}_{P,i} = \begin{bmatrix} 0 \\ \vdots \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{3-4}$$

A candidate approach for transforming (3-1) into (3-2) is approximate dynamic inversion or feedback linearization (see i.a. [118], [95], [41], [114], [149]). The underlying idea is to invert the system dynamics in (3-1) in order to bring the nonlinear system into an equivalent linear form. However, since the nonlinear dynamics are at most partially known, an approximate model of (3-1) has to be used. The difference between the dynamics in (3-1) and the approximate model inevitably introduces uncertainties to the system in (3-2).

Chowdhary et al. divide the uncertainty $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$ into two classes. A structured uncertainty is linearly parametrized and its basis vector is known. In the more general case the basis of the uncertainty is unknown and it needs to be approximated. In this case, the uncertainty is labeled unstructured. Definitions 3.1 and 3.2 formally define the two concepts.

Definition 3.1 – Structured Uncertainty ([41])

There exist ideal unknown weights $\boldsymbol{\Theta}^* \in \mathbb{R}^{k \times l}$ and a vector of *known* basis functions $\boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u}) \in \mathbb{R}^k$ such that the uncertainty $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$ can be uniquely represented by

$$\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u}) = \boldsymbol{\Theta}^{*T} \boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u}). \tag{3-5}$$

Definition 3.2 – Unstructured Uncertainty ([41])

There exist ideal unknown weights $\boldsymbol{\Theta}^* \in \mathbb{R}^{k \times l}$ and a vector of basis functions $\boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u}) \in \mathbb{R}^k$ such that the uncertainty $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$ can be represented by

$$\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u}) = \boldsymbol{\Theta}^{*T} \boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u}) + \boldsymbol{\eta}(\boldsymbol{x}_P, \boldsymbol{u}), \tag{3-6}$$

Where the representation error $\bar{\eta} = \sup_{x_P \in D_x, \boldsymbol{u} \in D_u} \|\boldsymbol{\eta}(\boldsymbol{x}_P, \boldsymbol{u})\|$ is bounded over D_x and D_u.

The optimal parameters $\boldsymbol{\Theta}^*$ are assumed to be time invariant throughout this work. For further information about adaptive control in the presence of time variant uncertainties refer i.a. to [131], [140], [209]. A large class of uncertainties satisfies Definition 3.1 (see e.g. the nonlinear wingrock dynamics model [196], [142]). Table 3-1 lists some general examples for structured uncertainties $\boldsymbol{\Delta}(\boldsymbol{x_P}, \boldsymbol{u})$.

Type of uncertainty	Mathematical Expression for $\Delta(\mathrm{x_P}, \mathbf{u})$
Stabilizing Feedback	$\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u}) = \boldsymbol{\Theta}^{*T} \boldsymbol{x}_P$
Parametrized Uncertainty	$\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u}) = \boldsymbol{\Theta}^{*T} \boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u})$
Constant Disturbance	$\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u}) = \boldsymbol{\Theta}^{*T}$
Input Uncertainty	$\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u}) = \boldsymbol{\Theta}^{*T} \boldsymbol{u}$

Table 3-1: Examples for uncertainties $\Delta(x_P, u)$ satisfying Definition 3.1

The remainder of this chapter focusses on unstructured uncertainties as it constitutes the more general case. In this, the basis $\boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u})$ is formed out of universally approximating functions, such as Gaussian radial basis functions ([130], [124]). Note, that picking the basis $\boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u})$ such that $\bar{\eta} = 0$, implies that the uncertainty approximation is exact in D_x and D_u. However, this requires infinitely many approximating functions ([69]), which is infeasible for practical applications.

3.2 Control Architecture

Figure 3-1 shows the control architecture employed in this thesis. The control task is divided into two parts, the baseline and the adaptive controller. The objective of the baseline controller is to ensure the desired performance of the closed loop system in the absence of the uncertainty $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$. In turn the adaptive controller aims at approximating the uncertainty and reducing its effect on the plant dynamics.

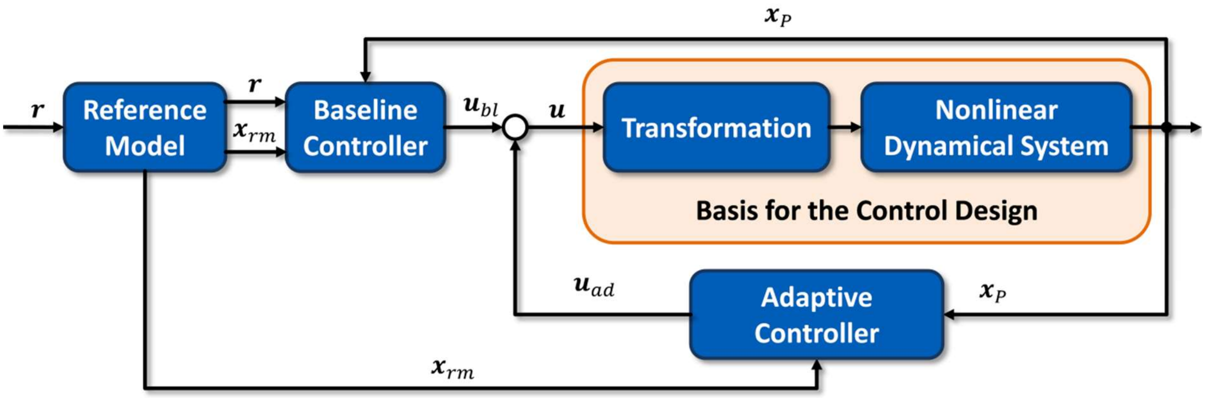


Figure 3-1: General control architecture with baseline and adaptive controller

A basic and frequently implemented approach for the baseline controller design is to use pole placement. However, for many missions it is desirable to let the output of the system follow a given reference trajectory instead of regulating it to the origin. The reference trajectory is generated by a reference model, which encompasses the designers' objectives such as performance criteria and state constraints. An imperative requirement for the reference model design is for the plant to be able to physically track the reference states exactly. Due to this prerequisite and a frequently experienced lack of knowledge of the system performance, reference models tend to be linear and conservative.

Let $\boldsymbol{x}_{rm} \in D_x \subset \mathbb{R}^n$ denote the state of the reference model dynamics

$$\dot{\boldsymbol{x}}_{rm} = \boldsymbol{A}_{rm}\boldsymbol{x}_{rm} + \boldsymbol{B}_{rm}\boldsymbol{r}. \tag{3-7}$$

The pair $(\boldsymbol{A}_{rm}, \boldsymbol{B}_{rm})$, with $\boldsymbol{A}_{rm} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{B}_{rm} \in \mathbb{R}^{n\times l}$, defines the desired response characteristics of the reference model. The system matrix $\boldsymbol{A}_{rm}$ is chosen to be Hurwitz. Here, $\boldsymbol{r} \in D_r \subset \mathbb{R}^l$ denotes the bounded and piecewise continuous reference command. For the purpose of this thesis assume that the reference model can be brought into the controllable canonical form of (3-3) such that

$$\dot{\boldsymbol{x}}_{rm} = \boldsymbol{A}_P\boldsymbol{x}_{rm} + \boldsymbol{B}_P\boldsymbol{u}_{rm}. \tag{3-8}$$

The internal reference model control law

$$\boldsymbol{u}_{rm} = \boldsymbol{K}_{rm,x}^T\boldsymbol{x}_{rm} + \boldsymbol{K}_{rm,r}^T\boldsymbol{r} \tag{3-9}$$

ensures that $\boldsymbol{A}_{rm} = \boldsymbol{A}_P + \boldsymbol{B}_P\boldsymbol{K}_{rm,x}^T$ and $\boldsymbol{B}_{rm} = \boldsymbol{B}_P\boldsymbol{K}_{rm,r}^T$. The matrices $\boldsymbol{K}_{rm,x} \in \mathbb{R}^{n\times l}$ and $\boldsymbol{K}_{rm,r} \in \mathbb{R}^{l\times l}$ contain the feedback and feedforward gains, respectively.

The tracking error between the system and reference model states is defined as

$$\boldsymbol{e} = \boldsymbol{x}_P - \boldsymbol{x}_{rm}. \tag{3-10}$$

As part of the control concept, the control input $\mathbf{u}$ consists of a term $\boldsymbol{u}_{bl}$ for the baseline controller and a term $\boldsymbol{u}_{ad}$ for the adaptive controller, respectively:

$$\boldsymbol{u} = \boldsymbol{u}_{bl} + \boldsymbol{u}_{ad}. \tag{3-11}$$

The control law for the baseline controller is defined as

$$\boldsymbol{u}_{bl} = \boldsymbol{u}_{rm} + \boldsymbol{K}_e^T \boldsymbol{e}, \tag{3-12}$$

where the matrix $\boldsymbol{K}_e \in \mathbb{R}^{n\times l}$ contains the feedback gains for the tracking error and $\boldsymbol{u}_{rm}$ is the reference model control law in (3-9). Define $\boldsymbol{A}_e = \boldsymbol{A}_P + \boldsymbol{B}_P \boldsymbol{K}_e^T$. The feedback gains $\boldsymbol{K}_e$ are chosen such that $\boldsymbol{A}_e$ is Hurwitz. Inserting (3-9), (3-11) and (3-12) into the plant dynamics of (3-2) gives

$$\dot{\boldsymbol{x}}_P = \boldsymbol{A}_P \boldsymbol{x}_P + \boldsymbol{B}_P \boldsymbol{K}_e^T \boldsymbol{e} + \boldsymbol{B}_P \boldsymbol{K}_{rm,x}^T \boldsymbol{x}_{rm} + \boldsymbol{B}_P \boldsymbol{K}_{rm,r}^T \boldsymbol{r} + \boldsymbol{B}_P\big(\boldsymbol{u}_{ad} - \boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})\big). \tag{3-13}$$

Note, that by choosing $\boldsymbol{K}_e = \boldsymbol{K}_{rm,x}$ the baseline tracking controller would be equal to a pole placement controller. Differentiating (3-10) with respect to time and inserting (3-7) and (3-13) yields the error dynamics

$$\begin{aligned} \dot{\boldsymbol{e}} &= \boldsymbol{A}_P \boldsymbol{x}_P + \boldsymbol{B}_P \boldsymbol{K}_e^T \boldsymbol{e} + \boldsymbol{B}_P \boldsymbol{K}_{rm,x}^T \boldsymbol{x}_{rm} + \underbrace{\boldsymbol{B}_P \boldsymbol{K}_{rm,r}^T}_{\boldsymbol{B}_{rm}} \boldsymbol{r} + \boldsymbol{B}_P\big(\boldsymbol{u}_{ad} - \boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})\big) \\ &\quad - \underbrace{\boldsymbol{A}_{rm} \boldsymbol{x}_{rm}}_{=(\boldsymbol{A}_P + \boldsymbol{B}_P \boldsymbol{K}_{rm,x}^T)\boldsymbol{x}_{rm}} - \boldsymbol{B}_{rm} \boldsymbol{r} \\ &= \underbrace{\boldsymbol{A}_P \boldsymbol{x}_P - \boldsymbol{A}_P \boldsymbol{x}_{rm} + \boldsymbol{B}_P \boldsymbol{K}_e^T \boldsymbol{e}}_{\boldsymbol{A}_e \boldsymbol{e}} + \boldsymbol{B}_P\big(\boldsymbol{u}_{ad} - \boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})\big) \\ &= \boldsymbol{A}_e \boldsymbol{e} + \boldsymbol{B}_P\big(\boldsymbol{u}_{ad} - \boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})\big). \end{aligned} \tag{3-14}$$

Note that if the adaptive controller is able to cancel the uncertainty $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$ uniformly in D_x and D_u, the tracking error dynamics reduce to $\dot{\boldsymbol{e}} = \boldsymbol{A}_e \boldsymbol{e}$. Since $\boldsymbol{A}_e$ is Hurwitz, the closed loop system is exponentially stable and the plant tracks the reference model exponentially fast. The following section focusses on the design of an adaptive control algorithm, which approximates the uncertainty $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$ and reduces its effect on the tracking error dynamics in (3-14).

3.3 Model Reference Adaptive Control

MRAC aims at cancelling the effect of the uncertainty on the tracking error dynamics of (3-14) and letting the plant track the reference model. This is achieved by emulating the structure of the uncertainty and approximating it using a weighted combination of regressor functions. The numerical values of the adaptive parameters are adjusted online based on the minimization of a quadratic cost. The underlying assumption is that if the adaptive gains do converge to their ideal values, the uncertainty is cancelled and the plant tracks the reference model.

Fundamentally, most MRAC approaches can be classified based on the chosen structure of the adaptive element and the parameter update scheme. In the former case, if the uncertainty is structured (as per Definition 3.1), the adaptive element directly emulates its composition. I.e. the regressor functions of the adaptive element equal the basis functions of the uncertainty. If the adaptive weights do converge, the uncertainty is cancelled uniformly over time. In the unstructured case (as per Definition 3.2) the basis functions of the uncertainty are unknown. Universal approximators represent one way of reducing the effect of the uncertainty on the plant nonetheless. One class of universal approximators, which has been used successfully, are neural networks (see i.a. [110], [92], [113]). The underlying idea is to approximate the modelling uncertainty by a large number of regressor functions, such as radial basis functions. Since the problem is over-determined, a uniform cancellation becomes nearly impossible, which in turn results in an approximation error acting on the error dynamics. However, if this error is small enough, the system can still be shown to be stable and track the reference model adequately.

Based on the type of adaptation laws, MRAC techniques can be divided into direct and indirect approaches ([17], [58], [156]). Direct MRAC adjusts the adaptive parameters online, in order to minimize a cost function on the tracking and parameter error. In contrast, indirect MRAC aims at identifying the parameters of the plant by utilizing an identification model. Instead of the tracking error, the identification error between the identification model and the plant drives the update of the adaptive gains. A detailed overview of indirect MRAC can be found i.a. in [18], [17], [58], [156]. Indirect MRAC forms the basis for some advanced strategies, such as $\mathcal{L}_1$ adaptive control or Predictor based MRAC. Both lead to smoother transients, while conserving a fast adaptation speed.

Numerous modifications and extensions exist within the mentioned families of MRAC architectures. Considering all available modifications exceeds the scope of this work. Hence, this thesis picks one representative of the direct and one of the indirect family in order to present and apply the online monitoring approach. The former family will be represented by direct MRAC with a gradient based parameter update law. The latter family, will be represented by Predictor based MRAC (PMRAC), which recently attracted increased attention.

The outline of this chapter is as follows. Section 3.3.1 introduces direct MRAC with gradient based update laws. Section 3.3.2 introduces PMRAC. Section 3.3.2 gives a concise overview of modifications, extensions and alternatives to the presented MRAC cases.

3.3.1 Direct MRAC with gradient based update laws

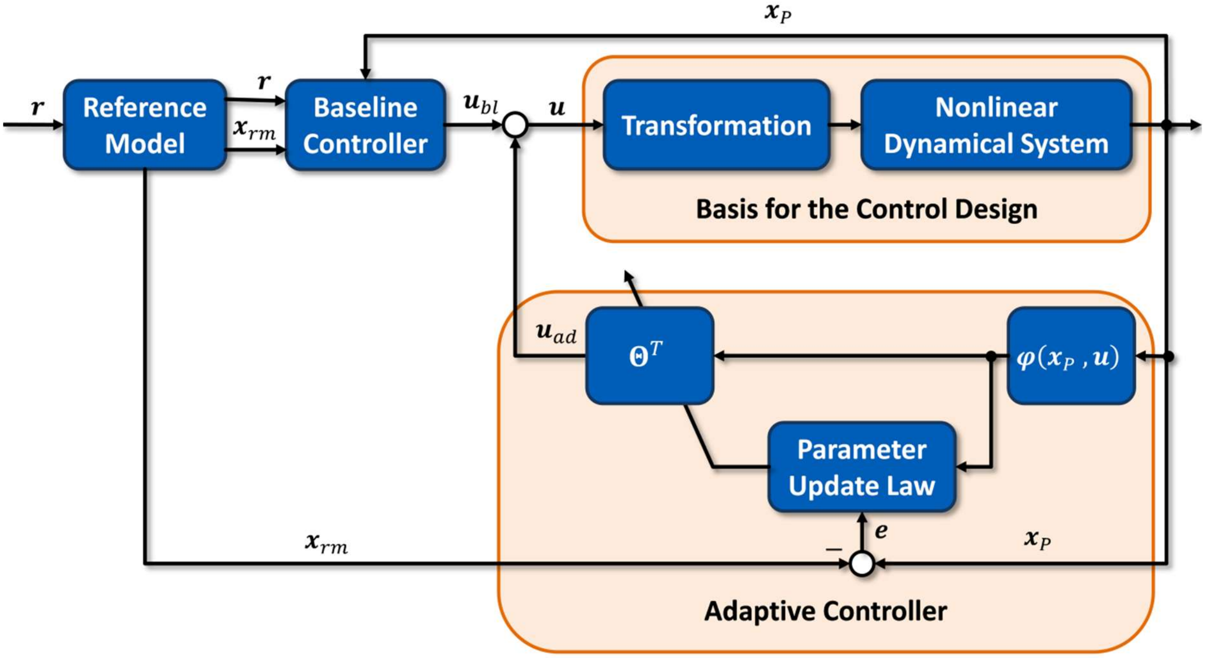


Figure 3-2: Control Architecture with Direct Model Reference Adaptive Controller

Figure 3-2 shows the control architecture with direct MRAC augmentation. The premise of the latter is to cancel the uncertainty directly in the input channel. Therefore, the control signal adopts the structure of the uncertainty: Similar to Definition 3.1 and Definition 3.2, the adaptive signal is a linear combination of the adaptive parameters $\boldsymbol{\Theta} \in \mathbb{R}^{k\times l}$ and basis functions $\boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u})$ such that

$$\boldsymbol{u}_{ad} = \boldsymbol{\Theta}^T \boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u}). \tag{3-15}$$

Since $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$ is a function of $\boldsymbol{u}_{ad}$ as per (3-11) and (3-13), and $\boldsymbol{u}_{ad}$ needs to be designed to cancel $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$, the existence and uniqueness of a fixed-point solution to $\boldsymbol{u}_{ad} = \boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$ is assumed. Sufficient conditions for satisfying this assumption are available in [123]. Define the parameter error $\widetilde{\boldsymbol{\Theta}} \in \mathbb{R}^{k\times l}$ between the current weights $\boldsymbol{\Theta}$ and the ideal weights $\boldsymbol{\Theta}^*$ to be

$$\widetilde{\boldsymbol{\Theta}} = \boldsymbol{\Theta} - \boldsymbol{\Theta}^*. \tag{3-16}$$

Considering the more general uncertainty representation of Definition 3.2 and inserting (3-15) into the error dynamics in (3-14) yields

$$\dot{\boldsymbol{e}} = \boldsymbol{A}_e \boldsymbol{e} + \boldsymbol{B}_P\left(\boldsymbol{\Theta}^T \boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u}) - \boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})\right) = \boldsymbol{A}_e \boldsymbol{e} + \boldsymbol{B}_P \widetilde{\boldsymbol{\Theta}}^T \boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u}) + \boldsymbol{B}_P \boldsymbol{\eta}(\boldsymbol{x}_P, \boldsymbol{u}). \tag{3-17}$$

Since $\boldsymbol{A}_e$ is Hurwitz, for any positive definite matrix $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ there exists a positive definite symmetric solution $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ for the Lyapunov equation [84, p. 173]

$$\boldsymbol{0} = \boldsymbol{A}_e^T \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}_e + \boldsymbol{Q}. \tag{3-18}$$

The task of the adaptive controller is to reduce the impact of the uncertainty $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$ on the error dynamics (3-14). MRAC achieves this by updating the numerical values of the adaptive parameters online in order to minimize a cost function on the tracking and parameter error. In most direct MRAC architectures, the update on the adaptive weights is driven by the instantaneous tracking error $\boldsymbol{e}$ and given by

$$\dot{\mathbf{\Theta}} = -\mathbf{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P,\boldsymbol{u})\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P - \mathbf{\Gamma}k(\mathbf{\Theta}). \tag{3-19}$$

The matrix $\mathbf{\Gamma} \in \mathbb{R}^{k\times k}$ contains the learning rate. Often, $\mathbf{\Gamma}$ is chosen to be diagonal. Furthermore, $k(\mathbf{\Theta})$ denotes an update law modification, which ensures stability of the closed loop system in the presence of disturbances. Let $c_i, \mathrm{i} \in (0,1,2)$, be constants with $c_0 > 0$. The update law modification $k(\mathbf{\Theta})$ takes on the form

$$k(\mathbf{\Theta}) = c_0\tilde{\mathbf{\Theta}} \pm c_1\bar{\mathbf{\Theta}} \pm c_2\boldsymbol{I}. \tag{3-20}$$

Update law modifications, which satisfy (3-20) include amongst others Ioannou's σ-Modification ([102]) or concurrent learning by Chowdhary ([41], [35], [42]). Here $\bar{\mathbf{\Theta}} \in \mathbb{R}^{k\times l}$ denotes an additional, constant term, which comprises parameter errors resulting from wrong assumptions or false initialization. For example, σ-Modification requires a preselected reference damping value, which is often chosen to be 0, since no further information is available. Section 3.3.2 elaborates on update law modifications. Remember that the optimal adaptive parameters $\mathbf{\Theta}^*$ are assumed to be time invariant. Hence, using (3-19) the parameter error dynamics are

$$\dot{\tilde{\mathbf{\Theta}}} = \dot{\mathbf{\Theta}} - \dot{\mathbf{\Theta}}^* = -\mathbf{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P,\boldsymbol{u})\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P - \mathbf{\Gamma}k(\mathbf{\Theta}). \tag{3-21}$$

The following theorem establishes the well known stability proof for MRAC with the gradient decent based update laws of (3-19) (see e.g. [155], [8], [204]).

Theorem 3.1 – Uniform Ultimate Boundedness of MRAC

Consider the system dynamics in (3-2), the reference model dynamics in (3-7), the control law in (3-11), the tracking error dynamics (3-17), the parameter error dynamics (3-21) and Definition 3.2. Then the instantaneous parameter update law in (3-19) guarantees that the zero solution $(\boldsymbol{e},\tilde{\mathbf{\Theta}}) = 0$ of the closed loop system (3-17) and (3-21) is globally uniformly ultimately bounded.

Proof:

Consider the following Lyapunov function candidate

$$V(\boldsymbol{e},\tilde{\mathbf{\Theta}}) = \frac{1}{2}\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{e} + \frac{1}{2}\mathrm{Tr}(\tilde{\mathbf{\Theta}}^T\mathbf{\Gamma}^{-1}\tilde{\mathbf{\Theta}}). \tag{3-22}$$

Note that (3-22) is positive definite and radially unbounded as $V(\boldsymbol{e},\tilde{\mathbf{\Theta}}) = 0$ iff $\boldsymbol{e} = \mathbf{0}$ and $\tilde{\mathbf{\Theta}} = 0$, and $V(\boldsymbol{e},\tilde{\mathbf{\Theta}}) \to \infty$ for $\boldsymbol{e} \to \infty$ and $\tilde{\mathbf{\Theta}} \to \infty$.

Taking the time derivative of (3-22) along the tracking and parameter error dynamics of (3-17) and (3-21) yields

$$\dot{V}\left(\boldsymbol{e},\tilde{\boldsymbol{\Theta}}\right)=\frac{1}{2}\boldsymbol{e}^T\boldsymbol{P}\dot{\boldsymbol{e}}+\frac{1}{2}\dot{\boldsymbol{e}}^T\boldsymbol{P}\boldsymbol{e}+\mathrm{Tr}\left(\tilde{\boldsymbol{\Theta}}^T\boldsymbol{\Gamma}^{-1}\dot{\boldsymbol{\Theta}}\right)$$

$$=\frac{1}{2}\boldsymbol{e}^T\underbrace{\left(\boldsymbol{A}_e^T\boldsymbol{P}+\boldsymbol{P}\boldsymbol{A}_e\right)}_{-\boldsymbol{Q}}\boldsymbol{e}+\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P\boldsymbol{\eta}(\boldsymbol{x}_P,\boldsymbol{u})+\mathrm{Tr}\left(\tilde{\boldsymbol{\Theta}}^T\boldsymbol{\Gamma}^{-1}\dot{\boldsymbol{\Theta}}+\tilde{\boldsymbol{\Theta}}^T\boldsymbol{\varphi}(\boldsymbol{x}_P,\boldsymbol{u})\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P\right) \qquad \textbf{(3-23)}$$

$$=\frac{1}{2}\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e}+\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P\boldsymbol{\eta}(\boldsymbol{x}_P,\boldsymbol{u})-\mathrm{Tr}\left(c_0\tilde{\boldsymbol{\Theta}}^T\tilde{\boldsymbol{\Theta}}\pm c_1\tilde{\boldsymbol{\Theta}}^T\bar{\boldsymbol{\Theta}}\pm c_2\tilde{\boldsymbol{\Theta}}^T\right).$$

Constructing an upper bound on $\dot{V}\left(\boldsymbol{e},\tilde{\boldsymbol{\Theta}}\right)$ gives

$$\dot{V}\left(\boldsymbol{e},\tilde{\boldsymbol{\Theta}}\right)\leq-\frac{1}{2}\lambda_{\min}(\boldsymbol{Q})\|\boldsymbol{e}\|^2-c_0\left\|\tilde{\boldsymbol{\Theta}}\right\|^2+\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|\|\boldsymbol{e}\|+(c_1\|\bar{\boldsymbol{\Theta}}\|+c_2)\left\|\tilde{\boldsymbol{\Theta}}\right\|. \qquad \textbf{(3-24)}$$

For the ease of readability define $c_3=c_1\|\bar{\boldsymbol{\Theta}}\|+c_2$. In order to proof Theorem 3.1 we need to construct a positively invariant set with respect to $V\left(\boldsymbol{e},\tilde{\boldsymbol{\Theta}}\right)$ using (3-24). Consider the following compact set, which originates from completing the squares in (3-24)

$$\Omega=\left\{\left(\boldsymbol{e},\tilde{\boldsymbol{\Theta}}\right)\middle|\frac{1}{2}\lambda_{min}(\boldsymbol{Q})\left[\|\boldsymbol{e}\|-\frac{\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|}{\lambda_{min}(\boldsymbol{Q})}\right]^2+c_0\left[\left\|\tilde{\boldsymbol{\Theta}}\right\|-\frac{c_3}{2c_0}\right]^2\leq\frac{(\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|)^2}{2\lambda_{min}(\boldsymbol{Q})}+\frac{c_3^2}{4c_0}\right\}. \qquad \textbf{(3-25)}$$

Then $\dot{V}\left(\boldsymbol{e},\tilde{\boldsymbol{\Theta}}\right)<0$ outside of Ω. The set Ω is not necessarily invariant with respect to $V\left(\boldsymbol{e},\tilde{\boldsymbol{\Theta}}\right)$. Hence, consider the contour of Ω resulting from setting (3-24) equal to zero:

$$\frac{1}{2}\lambda_{min}(\boldsymbol{Q})\left[\|\boldsymbol{e}\|-\frac{\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|}{\lambda_{min}(\boldsymbol{Q})}\right]^2+c_0\left[\left\|\tilde{\boldsymbol{\Theta}}\right\|-\frac{c_3}{2c_0}\right]^2-\frac{(\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|)^2}{2\lambda_{min}(\boldsymbol{Q})}-\frac{c_3^2}{4c_0}=0. \qquad \textbf{(3-26)}$$

The largest tracking error e_Ω on the contour of Ω is found by setting $\left\|\tilde{\boldsymbol{\Theta}}\right\|=\frac{c_3}{2c_0}$ and solving (3-26) to $\|\mathbf{e}\|$

$$e_\Omega=\frac{\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|+\sqrt{(\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|)^2+\frac{\lambda_{min}(\boldsymbol{Q})c_3^2}{2c_0}}}{\lambda_{min}(\boldsymbol{Q})}. \qquad \textbf{(3-27)}$$

The largest parameter error $\tilde{\Theta}_\Omega$ on the contour of Ω is found by setting $\|\boldsymbol{e}\|=\frac{\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|}{\lambda_{min}(\boldsymbol{Q})}$ and solving (3-26) to $\left\|\tilde{\boldsymbol{\Theta}}\right\|$

$$\tilde{\Theta}_\Omega=\frac{c_3+\sqrt{c_3^2+\frac{2c_0(\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|)^2}{\lambda_{min}(\boldsymbol{Q})}}}{2c_0}. \qquad \textbf{(3-28)}$$

Let

$$\beta=\frac{1}{2}\lambda_{max}(\boldsymbol{P})e_\Omega^2+\frac{1}{2}\lambda_{max}(\boldsymbol{\Gamma}^{-1})\tilde{\Theta}_\Omega^2 \qquad \textbf{(3-29)}$$

and define the compact set

$$\Omega_\beta = \{(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}}) | V(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}}) \le \beta\}. \tag{3-30}$$

Note that $\Omega \subseteq \Omega_\beta$ with $\dot{V}(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}}) < 0$ outside of Ω_β. It follows that Ω_β is positively invariant with respect to $V(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}})$. Following LaSalle's invariance principle in Theorem A.1 the zero solution $(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}}) = 0$ of the closed loop system (3-17) and (3-19) is uniformly ultimately bounded. Since $V(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}})$ is radially unbounded, this result is global.

□

Figure 3-3 shows a geometrical interpretation of the stability proof in Theorem 3.1. The set Ω is not necessarily invariant with respect to $\mathrm{V}(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}})$ as the trajectory of $\left(\boldsymbol{e}(\mathrm{t}), \widetilde{\boldsymbol{\Theta}}(\mathrm{t})\right)$ can leave Ω even though $\dot{\mathrm{V}}(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}}) < 0$. Hence, a set $\Omega_\mathrm{\beta}$ needs to be found, whose level set corresponds to a Lyapunov level $\mathrm{V}(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}}) = \mathrm{\beta}$ and which is a superset of Ω in order to ensure $\dot{\mathrm{V}}(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}}) < 0$ outside of $\Omega_\mathrm{\beta}$. Therefore, the maximum tracking and parameter error $(e_\Omega, \widetilde{\Theta}_\Omega)$ on the contour of Ω are sought. Since Ω is convex, the Lyapunov function value $\mathrm{V}(e_\Omega, \widetilde{\Theta}_\Omega) = \mathrm{\beta}$ defines the level set for $\Omega_\mathrm{\beta}$ such that $\Omega \subseteq \Omega_\beta$ holds. As a result Ω_β is positive invariant, which ensures that every solution that starts in Ω_β stays in this set $\forall t \ge t_0$. As a result it is possible to calculate the maximum encountered tracking and parameter errors.

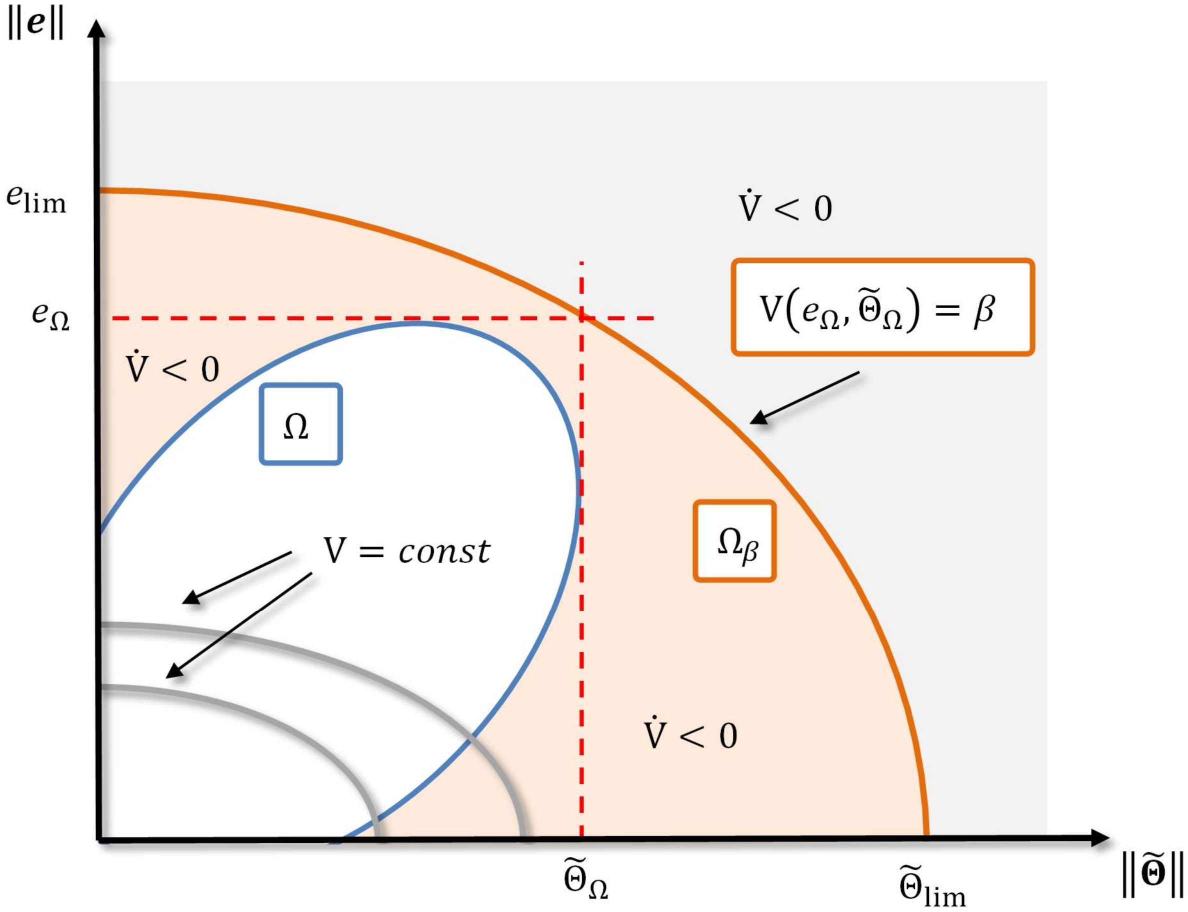

$||\boldsymbol{e}||$
$e_{\lim}$
e_Ω
$\dot{\mathrm{V}} < 0$
$\dot{\mathrm{V}} < 0$
$\mathrm{V}(e_\Omega, \widetilde{\Theta}_\Omega) = \beta$
Ω
$\mathrm{V} = const$
Ω_β
$\dot{\mathrm{V}} < 0$
$\widetilde{\Theta}_\Omega$
$\widetilde{\Theta}_{\lim}$
$||\widetilde{\boldsymbol{\Theta}}||$

Figure 3-3: Geometricl Interpretation of the stability proof in Theorem 3.1

Corollary 3.1

If Theorem 3.1 holds and furthermore $\left(\boldsymbol{e}(t_0), \widetilde{\boldsymbol{\Theta}}(t_0)\right) \in \Omega_\beta$, then the tracking and parameter error are bounded by $\|\boldsymbol{e}(\mathrm{t})\| \leq e_{lim}$ and $\left\|\widetilde{\boldsymbol{\Theta}}(\mathrm{t})\right\| \leq \widetilde{\Theta}_{lim} \forall \mathrm{t} > \mathrm{t}_0$, where $e_{lim} = \sqrt{\frac{2\beta}{\lambda_{max}(\boldsymbol{P})}}$ and $\widetilde{\Theta}_{lim} = \sqrt{\frac{2\beta}{\lambda_{max}(\boldsymbol{\Gamma}^{-1})}}$.

Proof:

Consider the level set $\Omega_{l\beta} = \left\{\left(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}}\right) \middle| \mathrm{V}\left(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}}\right) = \beta\right\}$ and the definition of β in (3-29). For $\left(\boldsymbol{e}(t_0), \widetilde{\boldsymbol{\Theta}}(t_0)\right) \in \Omega_\beta$, Theorem 3.1 ensures that $\left(\boldsymbol{e}(t), \widetilde{\boldsymbol{\Theta}}(t)\right) \in \Omega_{l\beta} \forall t \geq t_0$. Hence, $e_{lim} = \max_{\boldsymbol{e} \in \Omega_{l\beta}} \|\boldsymbol{e}\|$, which yields

$$e_{lim} = \sqrt{\frac{2\beta}{\lambda_{max}(\boldsymbol{P})}}. \tag{3-31}$$

Similarly, $\widetilde{\Theta}_{lim} = \max_{\widetilde{\boldsymbol{\Theta}} \in \Omega_{l\beta}} \left\|\widetilde{\boldsymbol{\Theta}}\right\|$, which yields

$$\widetilde{\Theta}_{lim} = \sqrt{\frac{2\beta}{\lambda_{max}(\boldsymbol{\Gamma}^{-1})}}. \tag{3-32}$$

□

An a-priori analysis based on the presented Lyapunov approach is often difficult or even impossible. While the Lyapunov analysis guarantees boundedness of all system signals, these bounds might be physically unreasonable or cannot be computed at all. E.g. in order to derive the maximum tracking and parameter error, knowledge about the optimal parameters $\boldsymbol{\Theta}^*$ is necessary. Even if an estimate of $\boldsymbol{\Theta}^*$ was available, the Lyapunov analysis projects multidimensional signals, i.e. $\boldsymbol{e}$ and $\widetilde{\boldsymbol{\Theta}}$, onto a scalar $\mathrm{V}\left(\boldsymbol{e}, \widetilde{\boldsymbol{\Theta}}\right)$ by using worst case mappings. Hence, even if a bound on $\boldsymbol{e}$ is computable, it might already exceed structural or operational limits. This is one reason for the lack of certifiability of adaptive controllers as discussed in section 3.4.

The presented direct MRAC approach still has been applied to a huge variety of systems. Thereby, direct MRAC has shown promising results in practice. If the limits of the Lyapunov analysis can be overcome, it would help adaptive controllers on the road to certification and thus increase its applicability.

A multitude of variations of the presented approach exist, which aim at improving certain aspects of the closed loop behavior. Section 3.3.3 summarizes the (in the view of the author) most important modifications. The following section introduces Predictor based MRAC as a representative of the family of indirect MRAC approaches.

3.3.2 Predictor based MRAC

Predictor based MRAC (PMRAC) ([128]) aims at improving the transient performance characteristics of the classical direct MRAC approach of section 3.3.2. PMRAC introduces a state predictor to the closed loop. Instead of updating the adaptive parameters dependent on the tracking error between the system and the reference model only, the new update laws also depend on the prediction error between system and predictor states. This concept reduces transient oscillations and ensures a smoother response ([128]). Figure 3-4 shows the PMRAC control architecture with the state predictor.

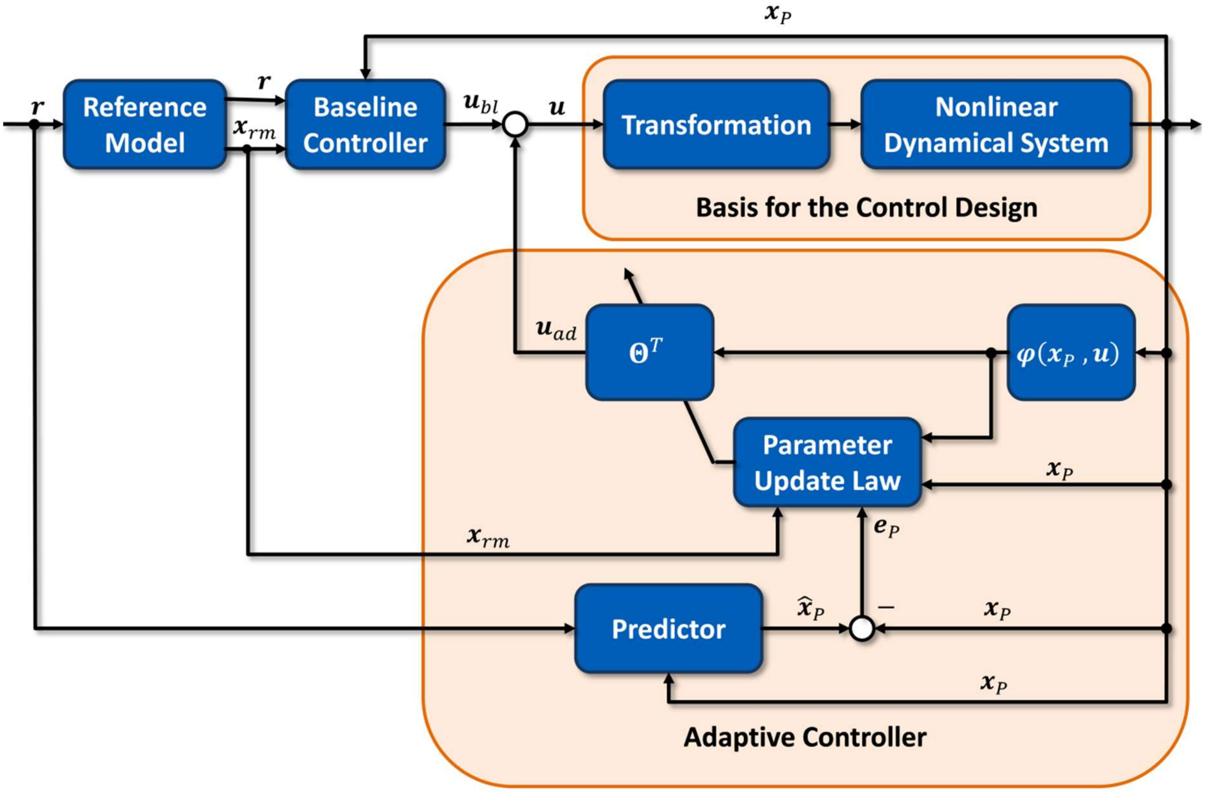


Figure 3-4: Control Architecture augmented by Predictor-Based Model Reference Adaptive Control

Let $\hat{\boldsymbol{x}}_P \in \mathbb{R}^n$ denote the states of the state predictor. The predictor dynamics are ([128]):

$$\dot{\hat{\boldsymbol{x}}}_P = \boldsymbol{A}_{prd}(\hat{\boldsymbol{x}}_P - \boldsymbol{x}_P) + \boldsymbol{A}_{rm}\boldsymbol{x}_P + \boldsymbol{B}_{rm}\boldsymbol{r}. \tag{3-33}$$

The system matrix of the state predictor $\boldsymbol{A}_{prd} \in \mathbb{R}^{n\times n}$ is chosen to be Hurwitz. Define the prediction error $\boldsymbol{e}_P \in \mathbb{R}^n$ to be

$$\boldsymbol{e}_P = \hat{\boldsymbol{x}}_P - \boldsymbol{x}_P. \tag{3-34}$$

Consequently, the predictor error dynamics are

$$\begin{aligned}\dot{\boldsymbol{e}}_P = \dot{\hat{\boldsymbol{x}}}_P - \dot{\boldsymbol{x}}_P = \boldsymbol{A}_{prd}(\hat{\boldsymbol{x}}_P - \boldsymbol{x}_P) + \boldsymbol{A}_{rm}\boldsymbol{x}_P + \boldsymbol{B}_{rm}\boldsymbol{r} - \boldsymbol{A}_P\boldsymbol{x}_P - \boldsymbol{B}_P\boldsymbol{K}_e^T\boldsymbol{e} - \boldsymbol{B}_P\boldsymbol{K}_{rm,x}^T\boldsymbol{x}_{rm} \\ - \boldsymbol{B}_P\boldsymbol{K}_{rm,r}^T\boldsymbol{r} - \boldsymbol{B}_P\big(\boldsymbol{u}_{ad} - \boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})\big).\end{aligned} \tag{3-35}$$

For the ease of depiction, assume $\boldsymbol{K}_{rm,x} = \boldsymbol{K}_e$. As a result $\boldsymbol{A}_{rm} = \boldsymbol{A}_e$ and the predictor error dynamics of (3-35) become

$$\begin{aligned}\dot{\boldsymbol{e}}_P &= \boldsymbol{A}_{prd}\boldsymbol{e}_P + \boldsymbol{A}_{rm}\boldsymbol{x}_P \underbrace{-\boldsymbol{A}_P\boldsymbol{x}_P \underbrace{-\boldsymbol{B}_P\boldsymbol{K}^T\boldsymbol{e} - \boldsymbol{B}_P\boldsymbol{K}^T\boldsymbol{x}_{rm}}_{=-\boldsymbol{B}_P\boldsymbol{K}^T\boldsymbol{x}_P}}_{=-\boldsymbol{A}_{rm}\boldsymbol{x}_P} - \boldsymbol{B}_P\big(\boldsymbol{u}_{ad} - \boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})\big) \\ &= \boldsymbol{A}_{prd}\boldsymbol{e}_P - \boldsymbol{B}_P\big(\boldsymbol{u}_{ad} - \boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})\big).\end{aligned} \tag{3-36}$$

Since $\boldsymbol{A}_{prd}$ is Hurwitz, for any positive definite matrix $\boldsymbol{Q}_{prd} \in \mathbb{R}^{n\times n}$ there exists a positive definite symmetric solution $\boldsymbol{P}_{prd} \in \mathbb{R}^{n\times n}$ for the Lyapunov equation ([84, p. 173])

$$\boldsymbol{0} = \boldsymbol{A}_{prd}^T\boldsymbol{P}_{prd} + \boldsymbol{P}_{prd}\boldsymbol{A}_{prd} + \boldsymbol{Q}_{prd} \tag{3-37}$$

In PMRAC the update of the adaptive weights is not only driven by the instantaneous tracking error $\boldsymbol{e}$ but also by the prediction error $\boldsymbol{e}_P$:

$$\dot{\boldsymbol{\Theta}} = -\boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u})(\boldsymbol{e}^T\boldsymbol{P} - \boldsymbol{e}_P^T\boldsymbol{P}_P)\boldsymbol{B}_P - \boldsymbol{\Gamma}k(\boldsymbol{\Theta}). \tag{3-38}$$

Consequently, equation (3-38) also represents the parameter error dynamics. The following theorem establishes the stability proof for PMRAC with the gradient decent based update laws of (3-19) (see [128]).

Theorem 3.2 – Uniform Ultimate Boundedness of PMRAC (based on [128])

Consider the system dynamics in (3-2), the reference model dynamics in (3-7), the predictor dynamics in (3-33), the control law in (3-11), the tracking error dynamics (3-17), the predictor error dynamics (3-36), the parameter error dynamics (3-38) and Definition 3.2. Then the instantaneous parameter update law in (3-38) guarantees that the zero solution $\big(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\big) = 0$ of the closed loop system (3-17), (3-36) and (3-38) is globally uniformly ultimately bounded.

Proof:

The proof largely follows the same arguments as Theorem 3.1. Consider the following Lyapunov function candidate

$$V\big(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\big) = \frac{1}{2}\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{e} + \frac{1}{2}\boldsymbol{e}_P^T\boldsymbol{P}_{prd}\boldsymbol{e}_P + \frac{1}{2}\mathrm{Tr}\big(\widetilde{\boldsymbol{\Theta}}^T\boldsymbol{\Gamma}^{-1}\widetilde{\boldsymbol{\Theta}}\big). \tag{3-39}$$

Note that (3-39) is positive definite and radially unbounded as $V\big(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\big) = 0$ iff $\boldsymbol{e} = \boldsymbol{0}, \boldsymbol{e}_P = \boldsymbol{0}$ and $\widetilde{\boldsymbol{\Theta}} = 0$, and $V\big(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\big) \to \infty$ for $\boldsymbol{e} \to \infty, \boldsymbol{e}_P \to \infty$ and $\widetilde{\boldsymbol{\Theta}} \to \infty$. Taking the time derivative of (3-39) along the tracking and prediction error dynamics of (3-17) and (3-36) yields

$$\begin{aligned}\dot{V}\big(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\big) &= \frac{1}{2}\boldsymbol{e}^T\boldsymbol{P}\dot{\boldsymbol{e}} + \frac{1}{2}\dot{\boldsymbol{e}}^T\boldsymbol{P}\boldsymbol{e} + \frac{1}{2}\dot{\boldsymbol{e}}_P^T\boldsymbol{P}_{prd}\boldsymbol{e}_P + \frac{1}{2}\boldsymbol{e}_P^T\boldsymbol{P}_{prd}\dot{\boldsymbol{e}}_P\mathrm{Tr}\big(\widetilde{\boldsymbol{\Theta}}^T\boldsymbol{\Gamma}^{-1}\dot{\boldsymbol{\Theta}}\big) \\ &= \frac{1}{2}\boldsymbol{e}^T\underbrace{(\boldsymbol{A}_e^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}_e)}_{-\boldsymbol{Q}}\boldsymbol{e} + \boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P\boldsymbol{\eta}(\boldsymbol{x}_P, \boldsymbol{u}) + \frac{1}{2}\boldsymbol{e}_P^T\underbrace{\big(\boldsymbol{A}_{prd}^T\boldsymbol{P}_{prd} + \boldsymbol{P}_{prd}\boldsymbol{A}_{prd}\big)}_{-\boldsymbol{Q}_{prd}}\boldsymbol{e}_P \\ &\quad +\mathrm{Tr}\big(\widetilde{\boldsymbol{\Theta}}^T\boldsymbol{\Gamma}^{-1}\dot{\boldsymbol{\Theta}} + \widetilde{\boldsymbol{\Theta}}^T\boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u})\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P - \widetilde{\boldsymbol{\Theta}}^T\boldsymbol{\varphi}(\boldsymbol{x}_P, \boldsymbol{u})\boldsymbol{e}_P^T\boldsymbol{P}\boldsymbol{B}_P\big).\end{aligned} \tag{3-40}$$

Inserting the parameter update laws of (3-38) results in

$$\dot{V}\left(\boldsymbol{e},\boldsymbol{e}_P,\widetilde{\boldsymbol{\Theta}}\right)=-\frac{1}{2}\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e}-\frac{1}{2}\boldsymbol{e}_P^T\boldsymbol{Q}_{prd}\boldsymbol{e}_P+\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P\boldsymbol{\eta}(\boldsymbol{x}_P,\boldsymbol{u})$$
$$-\mathrm{Tr}\left(c_0\widetilde{\boldsymbol{\Theta}}^T\widetilde{\boldsymbol{\Theta}}\pm c_1\widetilde{\boldsymbol{\Theta}}^T\overline{\boldsymbol{\Theta}}\pm c_2\widetilde{\boldsymbol{\Theta}}^T\right). \tag{3-41}$$

Constructing an upper bound on $\dot{V}\left(\boldsymbol{e},\boldsymbol{e}_P,\widetilde{\boldsymbol{\Theta}}\right)$ gives

$$\dot{V}\left(\boldsymbol{e},\boldsymbol{e}_P,\widetilde{\boldsymbol{\Theta}}\right)\leq-\frac{1}{2}\lambda_{min}(\boldsymbol{Q})\|\boldsymbol{e}\|^2-\frac{1}{2}\lambda_{min}\left(\boldsymbol{Q}_{prd}\right)\|\boldsymbol{e}_P\|^2-c_0\left\|\widetilde{\boldsymbol{\Theta}}\right\|^2+\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|\|\boldsymbol{e}\|$$
$$+(c_1\|\overline{\boldsymbol{\Theta}}\|+c_2)\left\|\widetilde{\boldsymbol{\Theta}}\right\|. \tag{3-42}$$

For the ease of readability define $c_3=c_1\|\overline{\boldsymbol{\Theta}}\|+c_2$. The proof follows by constructing a positively invariant set with respect to $V\left(\boldsymbol{e},\boldsymbol{e}_P,\widetilde{\boldsymbol{\Theta}}\right)$ using (3-42). Consider the following compact set, which originates from completing the squares in (3-42)

$$\Omega=$$
$$=\left\{\begin{matrix}\left(\boldsymbol{e},\boldsymbol{e}_P,\widetilde{\boldsymbol{\Theta}}\right)\Big|\frac{1}{2}\lambda_{min}(\boldsymbol{Q})\left[\|\boldsymbol{e}\|-\frac{\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|}{\lambda_{min}(\boldsymbol{Q})}\right]^2+c_0\left[\left\|\widetilde{\boldsymbol{\Theta}}\right\|-\frac{c_3}{2c_0}\right]^2+\frac{1}{2}\lambda_{min}\left(\boldsymbol{Q}_{prd}\right)\|\boldsymbol{e}_P\|^2\\ \leq\frac{(\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|)^2}{2\lambda_{min}(\boldsymbol{Q})}+\frac{c_3^2}{4c_0}\end{matrix}\right\}. \tag{3-43}$$

Then $\dot{V}\left(\boldsymbol{e},\boldsymbol{e}_P,\widetilde{\boldsymbol{\Theta}}\right)<0$ outside of Ω. The set Ω is not necessarily invariant with respect to $V\left(\boldsymbol{e},\boldsymbol{e}_P,\widetilde{\boldsymbol{\Theta}}\right)$. Hence, consider the contour of Ω resulting from setting (3-42) equal to zero:

$$\frac{1}{2}\lambda_{min}(\boldsymbol{Q})\left[\|\boldsymbol{e}\|-\frac{\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|}{\lambda_{min}(\boldsymbol{Q})}\right]^2+\frac{1}{2}\lambda_{min}\left(\boldsymbol{Q}_{prd}\right)\|\boldsymbol{e}_P\|^2+c_0\left[\left\|\widetilde{\boldsymbol{\Theta}}\right\|-\frac{c_3}{2c_0}\right]^2$$
$$-\frac{(\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|)^2}{2\lambda_{min}(\boldsymbol{Q})}-\frac{c_3^2}{4c_0}=0. \tag{3-44}$$

The largest tracking error e_Ω on the contour of Ω is found by setting $\left\|\widetilde{\boldsymbol{\Theta}}\right\|=\frac{c_3}{2c_0}$, $\|\boldsymbol{e}_P\|=0$ and solving (3-44) to $\|\mathbf{e}\|$

$$e_\Omega=\frac{\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|+\sqrt{(\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|)^2+\frac{\lambda_{min}(\boldsymbol{Q})c_3^2}{2c_0}}}{\lambda_{min}(\boldsymbol{Q})}. \tag{3-45}$$

The largest parameter error $\widetilde{\Theta}_\Omega$ on the contour of Ω is found by setting $\|\boldsymbol{e}\|=\frac{\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|}{\lambda_{min}(\boldsymbol{Q})}$, $\|\boldsymbol{e}_P\|=0$ and solving (3-44) to $\left\|\widetilde{\boldsymbol{\Theta}}\right\|$

$$\widetilde{\Theta}_\Omega=\frac{c_3+\sqrt{c_3^2+\frac{2c_0(\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|)^2}{\lambda_{min}(\boldsymbol{Q})}}}{2c_0}. \tag{3-46}$$

Finally the largest prediction error $e_{P,\Omega}$ on the contour of Ω is found by setting $\|\boldsymbol{e}\| = \frac{\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|}{\lambda_{min}(\boldsymbol{Q})}$, $\|\widetilde{\boldsymbol{\Theta}}\| = \frac{c_3}{2c_0}$ and solving (3-44) to $\|\boldsymbol{e}_P\|$

$$e_{P,\Omega} = \sqrt{\frac{(\bar{\eta}\|\boldsymbol{P}\boldsymbol{B}_P\|)^2}{\lambda_{min}(\boldsymbol{Q})\lambda_{min}(\boldsymbol{Q}_{prd})} + \frac{c_3^2}{2c_0\lambda_{min}(\boldsymbol{Q}_{prd})}} \tag{3-47}$$

Let

$$\beta = \frac{1}{2}\lambda_{max}(\boldsymbol{P})e_{\Omega}^2 + \frac{1}{2}\lambda_{max}(\boldsymbol{P}_{Prd})e_{P,\Omega}^2 + \frac{1}{2}\lambda_{max}(\boldsymbol{\Gamma}^{-1})\widetilde{\Theta}_{\Omega}^2 \tag{3-48}$$

and define the compact set

$$\Omega_{\beta} = \left\{\left(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\right) \middle| V\left(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\right) \leq \beta\right\}. \tag{3-49}$$

Note that $\Omega \subseteq \Omega_{\beta}$ with $\dot{V}\left(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\right) < 0$ outside of Ω_{β}. It follows that Ω_{β} is positively invariant with respect to $V\left(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\right)$. Following LaSalle's invariance principle in Theorem A.1, the zero solution $\left(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\right) = 0$ of the closed loop system (3-17), (3-36) and (3-38) is uniformly ultimately bounded. Since $V\left(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\right)$ is radially unbounded, this result is global.

□

Similar to direct MRAC, the maximum tracking, prediction and parameter error can be calculated using the following corollary.

Corollary 3.2

If Theorem 3.2 holds and furthermore $\left(\boldsymbol{e}(t_0), \boldsymbol{e}_P(t_0), \widetilde{\boldsymbol{\Theta}}(t_0)\right) \in \Omega_{\beta}$, then the tracking, prediction and parameter errors are bounded by $\|\boldsymbol{e}(\mathrm{t})\| \leq e_{\mathrm{lim}}$, $\left\|\boldsymbol{e}_p(t)\right\| \leq e_{\mathrm{P},lim}$ and $\left\|\widetilde{\boldsymbol{\Theta}}(\mathrm{t})\right\| \leq \widetilde{\Theta}_{lim} \forall \mathrm{t} > \mathrm{t}_0$, where $e_{lim} = \sqrt{\frac{2\beta}{\lambda_{max}(\boldsymbol{P})}}$, $e_{\mathrm{P},lim} = \sqrt{\frac{2\beta}{\lambda_{max}(\boldsymbol{P}_{prd})}}$ and $\widetilde{\Theta}_{lim} = \sqrt{\frac{2\beta}{\lambda_{max}(\boldsymbol{\Gamma}^{-1})}}$.

Proof:

Consider the level set $\Omega_{l\beta} = \left\{\left(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\right) \middle| \mathrm{V}\left(\boldsymbol{e}, \boldsymbol{e}_P, \widetilde{\boldsymbol{\Theta}}\right) = \beta\right\}$ and the definition of β in (3-48). For $\left(\boldsymbol{e}(t_0), \boldsymbol{e}_P(t_0), \widetilde{\boldsymbol{\Theta}}(t_0)\right) \in \Omega_{\beta}$, Theorem 3.2 ensures that $\left(\boldsymbol{e}(t), \boldsymbol{e}_P(t), \widetilde{\boldsymbol{\Theta}}(t)\right) \in \Omega_{l\beta}\ \forall t \geq t_0$. Hence, $e_{lim} = \max_{\boldsymbol{e} \in \Omega_{l\beta}} \|\boldsymbol{e}\|$, which yields

$$e_{lim} = \sqrt{\frac{2\beta}{\lambda_{max}(\boldsymbol{P})}}. \tag{3-50}$$

Similarly, $e_{\mathrm{P},lim} = \max_{\boldsymbol{e}_P \in \Omega_{l\beta}} \|\boldsymbol{e}_P\|$, which yields

$$e_{\mathrm{P},lim} = \sqrt{\frac{2\beta}{\lambda_{max}\left(\boldsymbol{P}_{prd}\right)}} \tag{3-51}$$

And, $\widetilde{\Theta}_{lim} = \max_{\widetilde{\boldsymbol{\Theta}} \in \Omega_{l\beta}} \left\|\widetilde{\boldsymbol{\Theta}}\right\|$, which yields

$$\widetilde{\Theta}_{lim} = \sqrt{\frac{2\beta}{\lambda_{max}(\boldsymbol{\Gamma}^{-1})}}. \tag{3-52}$$

□

Similar to the direct MRAC case of section 3.3.1, the derived bounds on the tracking, prediction and parameter error might not be computable or might lack physical significance. Still PMRAC has shown to yield promising results in application. Hence, overcoming the limited expressiveness of the Lyapunov analysis would benefit its applicability.

The following section deals with selected update law modifications, extensions as well as alternative approaches to the presented MRAC cases.

3.3.3 Modifications, Extensions and Alternatives

Numerous variations to the MRAC approaches of section 3.3.1 and 3.3.2 exist. They can loosely be classified into update law modifications, reference model modifications and alternative approaches. The following sections provide a concise overview of the most important modifications and approaches as seen by the author.

3.3.3.1 Update Law Modifications

Update law modifications change the gradient descent based learning law of (3-19) in order to alter properties of the closed loop behavior. Table 3-2 lists the (in the view of the author) most important update law modifications.

In the absence of a modification term $k(\boldsymbol{\Theta})$ in (3-19), a disturbance would cause the closed loop system to be unstable. In particular, the adaptive parameters start to diverge causing the tracking error to grow unbounded, which eventually leads to a system crash. This phenomenon is called parameter drift. In order to ensure boundedness of the adaptive parameters and therefore stability of the closed loop dynamics multiple robustness modifications were proposed.

The dead zone modification ([102], [100]) stops the adaptation once the tracking error $||\boldsymbol{e}||$ falls below a certain threshold $||\boldsymbol{e}|| \leq \epsilon$. The magnitude of ϵ is therefore directly proportional to the remaining tracking error. Since ϵ is chosen based on a Lyapunov argument, the bounds are inherently conservative leading to an often undesirably large tracking error.

In contrast, σ-Modification [101], [102]) and e-Modification ([154]) add a damping term to the update law, which is dependent on the current parameter estimate $\boldsymbol{\Theta}$. These modifications ensure boundedness of the parameter and tracking error. However, the damping terms try to force the weights back to a reference $\boldsymbol{\Theta}_0$, often chosen to be zero. Hence, even in the absence of disturbances, the adaptive parameters do not converge to their optimal values. Furthermore, these terms act as forgetting factors in local approximators such as Radial Basis Function Neural Networks ([183]), i.e. previously learned local estimates are forgotten.

The optimal modification ([159]) follows a slightly different strategy compared to the former approaches. It adds a damping term to the update law, which is proportional to the amount of persistency of excitation (see Definition A.1 in Appendix A.3) in the basis vector. As a result, the optimal modification allows for fast adaptation, while keeping undesired oscillations at a minimum. However, only boundedness of the product of weights and basis vector can be established. The parameters themselves are only bounded if the vector of basis functions is persistently excited.

Q-Modification ([214], [212], [213], [224]) employs a moving window of the integrated system uncertainty in order to identify the optimal parameters, while simultaneously suppressing the uncertainty. The approach is shown to ensure closed loop stability.

The projection algorithm ([74]) exerts yet a different concept. Instead of adding damping to the update law, it redirects the parameter update vector such that the adaptive weights do not leave an a-priori chosen convex set. This is achieved by scaling the component of the weight update law which is orthogonal to the set boundary if the expected weight trajectory threatens to leave it. If in addition the optimal weights are within the preselected convex set, the parameters are bounded and closed loop stability can be established. Even though the former condition requires some knowledge about the optimal parameters, the author in [17] argues that for real world applications a valid convex set can be derived from expected uncertainties.

Robustness modifications primarily aim at bounding the adaptive parameters and thus establishing stability of the closed loop system. Yet other alterations pursue the goal of improving the closed loop properties of an already stabilized system.

The goal of Adaptive Loop Recovery [27] is to recover the loop transfer properties of a reference model. In particular, the reference model stability margins, i.e. gain and time delay margin, should be preserved to the extent possible. This is achieved by adding a damping term to the update law, which is proportional to the derivative of the basis functions with respect to their dependent variables. Adaptive loop recovery ensures closed-loop stability if the uncertainty is linearly parametrized. In the presence of disturbances or approximation errors the authors in [27] suggest the additional use of robustness modifications such as σ-Modification, or e-Modification.

The K-Modification ([122], [121]) aims at obtaining smooth transient performance of the weight error dynamics. The underlying idea is to add a stiffness term to the parameter update law. Therefore, the adaptive parameters are integrated over a moving time window. In fact, combining K-Modification with an additional damping term, such as σ-Modification, or e-Modification, the gradient based update laws form a second order system. The damping and stiffness coefficients are dependent on the modification terms and adjustable by the designer. A desired behavior of the transient parameter error dynamics can therefore be prescribed.

The primary goal of most MRAC applications is to achieve asymptotic tracking of the reference model. In this, the behavior of the adaptive weights is of minor interest as long as they are bounded. However, parameter convergence exhibits some benefits over the pure tracking problem formulation. While partial asymptotic stability ensures convergence of the tracking error to the origin, it remains short on guarantees for the transient performance. Hence, the tracking error might rise to critical values and lead to a system failure. In contrast, exponential parameter convergence leads to exponential error convergence with guaranteed exponentially bounded transient performance ([185], [35]). Furthermore, if the adaptive parameters do converge to their true values the uncertainty is uniformly approximated. The resulting error dynamics are linear. If in addition the error controller is chosen such that $\boldsymbol{K}_e = \boldsymbol{K}_{rm,x}$, the plant recovers the phase and gain margins of the reference model. This facilitates the evaluation of the performance of the controller using well understood linear stability metrics ([3]).

To accomplish parameter convergence, the vector of basis functions needs to be persistently exciting ([204], [23], [155]). For a linear basis Boyd and Sastry have shown that the condition on persistency of excitation can be related to a condition on the spectral properties of the exogenous reference command ([23]). In particular for aircraft applications, this leads to a waste of fuel, puts additional stress on the system and might be operationally undesirable. Furthermore, the basis functions are often highly nonlinear, thus the approach lacks a verifiable condition for compliance.

To counter the disadvantage of persistency of excitation, Chowdhary et al. developed concurrent learning ([42], [41], [35]). The underlying idea is to update the adaptive parameters using instantaneous data as well as specifically stored data concurrently. If the data points were stored at a time when the vector of basis functions was excited, this information can be used in future updates to achieve parameter convergence. Chowdhary et al. introduced a verifiable condition on the linear independence of the stored data and showed exponential tracking and parameter convergence in the absence of the approximation error $\boldsymbol{\eta}(\boldsymbol{x}_P, \boldsymbol{u})$. If $\boldsymbol{\eta}(\boldsymbol{x}_P, \boldsymbol{u}) \neq 0$, concurrent learning still ensures exponential convergence of the parameters to a compact domain around their optimal values.

3.3.3.2 Reference Model Modifications and Extensions

Apart from alterations to the parameter update laws, also modifications for the reference model exist. They leverage online available information in order to alter the dynamics of the reference model.

Special kinds of uncertainties are neglected actuator dynamics. In this case, the reference model often demands a trajectory which is unachievable by the plant. In particular, if the actuator states saturate, the tracking error is not reduced by further parameter updates. Since the weight update laws are driven by the tracking error, the adaptive parameters grow unbounded. In order to counter this effect and allow adaptation in the presence of actuator saturation, Johnson developed *Pseudo Control Hedging* (see e.g. [111], [112], [113], [92]). The underlying idea is to slow down the reference model by a measure of the expected plant reaction deficit. Therefore, it requires knowledge about the actuator states, which are either measurable or can be estimated by employing models of the actuators ([92]). While Pseudo Control Hedging has demonstrated good performance in application ([110], [114]), it still abides a universal stability proof. The reason for this is that the reference model is not necessarily stable given the state feedback. However, Bierling showed stability for the special case of known control effectiveness by connecting Pseudo Control Hedging to $\mathcal{L}_1$ adaptive control ([19]). Furthermore, this modification changes the response characteristics of the reference model drastically. Hence, even if the plant tracks the reference model perfectly, the trajectory does not necessarily abide a-priori formulated requirements.

Closed Loop Reference Models ([73], [72], [225]) try to shape the transient performance of the tracking error dynamics. Therefore, feed the tracking error back into the reference model. The underlying idea is to pull the reference trajectory closer to the plant. This reduces the tracking error and in turn yields a smoother adaptation. In fact, Gibson derived bounds on the transient performance ([73]). However, similar to Pseudo Control Hedging, the response characteristics of the reference model change drastically. Thus a-priori formulated requirements may be violated.

The Reference Governor [76], [75], [12], [56] aims at shaping the reference trajectory in order to achieve desired performance characteristics, while simultaneously abiding input and state constraints. The basic idea is to use a model based approach, such as Model Predictive Control, in order to solve an optimal control problem. The mentioned approaches often assume linear plant dynamics or the existence of a stabilizing controller. The authors in ([145], [149]) extended the approach and applied it to reference models in MRAC architectures. The hypothesis is that if the closed loop system tracks the reference model exactly, and the reference model is controlled optimally, the plant recovers the same properties. The main drawback of MPC is its requirement on high computational power. However, note that the reference model dynamics are known a-priori and do not change during application. As a result, solving a multiparametric quadratic programming problem

([151], [14], [208], [80]) for the optimal control of the reference model yields an explicit solution. The online effort reduces to a function evaluation, which enables the use of the reference governor on systems with fast dynamics and low computational resources.

Name	Update Law Modification
Dead Zone Modification [102], [100]	Stop Adaptation if the tracking error $\lVert \boldsymbol{e} \rVert$ is below a certain threshold $\lVert \boldsymbol{e} \rVert \leq \epsilon$. $k(\boldsymbol{e},\boldsymbol{\Theta}) = \begin{cases} -\boldsymbol{\varphi}(\boldsymbol{x}_P,\boldsymbol{u})\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P & \text{if } \lVert \boldsymbol{e} \rVert \leq \epsilon \\ \boldsymbol{0} & \text{otherwise} \end{cases}$
σ**-Modification** [101], [102]	Add damping to the update law: $k(\boldsymbol{\Theta}) = \sigma(\boldsymbol{\Theta} - \boldsymbol{\Theta}_0), \sigma > 0$
e**-Modification** [154]	Add damping to the update law, which is proportional to the tracking error: $k(\boldsymbol{e},\boldsymbol{\Theta}) = \sigma\lVert \boldsymbol{e} \rVert(\boldsymbol{\Theta} - \boldsymbol{\Theta}_0), \sigma > 0$
Optimal Modification [159]	Add damping proportional to $\boldsymbol{\varphi}(\boldsymbol{x}_P,\boldsymbol{u})\boldsymbol{\varphi}^T(\boldsymbol{x}_P,\boldsymbol{u})$ to the update law: $k(\boldsymbol{x}_P,\boldsymbol{\Theta}) = \sigma\boldsymbol{\varphi}(\boldsymbol{x}_P,\boldsymbol{u})\boldsymbol{\varphi}^T(\boldsymbol{x}_P,\boldsymbol{u})(\boldsymbol{\Theta} - \boldsymbol{\Theta}_0)\boldsymbol{B}_P^T\boldsymbol{P}\boldsymbol{A}_{rm}^{-1}\boldsymbol{B}_P, \sigma > 0$
Q**-Modification** [214], [212], [213], [224]	Add a time history of the state and the control input to the update law: $k(\boldsymbol{\Theta}) = \left(\boldsymbol{\Theta}^T\boldsymbol{q}(t,t-t_d) - \boldsymbol{c}(t,t-t_d)\right)\boldsymbol{q}(t,t-t_d)$ $\boldsymbol{q}(t,t-t_d) = \int_{\mathrm{t}-t_d}^{\mathrm{t}} \boldsymbol{\varphi}\left(\boldsymbol{x}_P(s),\boldsymbol{u}(s)\right)ds\ , t_d > 0$ $\boldsymbol{c} = \boldsymbol{B}_P\left(\boldsymbol{e}(t) - \boldsymbol{e}(t-t_d) - \int_{t-t_d}^{t} \boldsymbol{A}_{rm}\boldsymbol{e}(s)\right) + \int_{t-t_d}^{t} \boldsymbol{u}_{ad}(s)ds$
Parameter Projection [74]	Define a convex set in the parameter space. Use a projection algorithm on the parameter update law if it threatens to cause the adaptive weights to leave the convex set.
Adaptive Loop Transfer Recovery [27]	Let $\boldsymbol{z} = (\boldsymbol{x}_P,\boldsymbol{u})$. Add a damping term, which is proportional to $\boldsymbol{\varphi}_z(\boldsymbol{z})\boldsymbol{\varphi}_z^T(\boldsymbol{z})$, to the update law $k(\boldsymbol{\Theta}) = \sigma\boldsymbol{\varphi}_z(\boldsymbol{z})\boldsymbol{\varphi}_z^T(\boldsymbol{z})\boldsymbol{\Theta},$ $\boldsymbol{\varphi}_z(\boldsymbol{z}) = \frac{d\boldsymbol{\varphi}(\boldsymbol{z})}{d\boldsymbol{z}}, \sigma > 0$
K**-Modification** [122], [121]	Add stiffness to the update law: $k(\boldsymbol{\Theta}) = \sigma \int_{t-t_d}^{t} \boldsymbol{\Theta}(s)ds\, , \sigma > 0, t_d > 0$
Concurrent Learning [42], [41], [35]	Add an update based on stored data to the update law: $k(\mathbf{x}_\mathrm{P},\boldsymbol{\Theta}) = \sum_{i=1}^{p} \boldsymbol{\varphi}_i(\boldsymbol{x}_P,\boldsymbol{u})\left[\boldsymbol{\varphi}_i^T(\boldsymbol{x}_P,\boldsymbol{u})\boldsymbol{\Theta} - \boldsymbol{\Delta}_i^T(\boldsymbol{x}_P,\boldsymbol{u})\right]$ $\boldsymbol{\varphi}_i(\boldsymbol{x}_P,\boldsymbol{u})$ and $\boldsymbol{\Delta}_i$ denote recorded data points from past time instants t_i.

Table 3-2: Update Law Modifications; based on [223, p. 68] and [17]

3.3.3.3 Alternative Approaches

The previously mentioned approaches share the MRAC structure as well as the fact that the adaptive weights are adjusted based on a gradient descent based update law. Alternative approaches which deviate from these two properties exist and found increasing interest in recent time. The following selection gives a brief overview of two further MRAC options.

Classic static Radial Basis Function Neural Network approaches priorly distribute the regressor functions throughout a region of the state space. Once the state leaves the prescribed domain, network approximation is impossible and system failures can occur. Overcoming this problem not only requires knowledge about the expected range of each state, but also leads to a high number of regressor functions, consequently requiring high computational power. Chowdhary et al. introduced GP based MRAC ([38], [124], [37]) in order to abrogate the inflexibility of static neural networks. The underlying idea is to employ results from computer science in order to learn the uncertainty from online gathered data only and cancel its effects on the plant dynamics. Each such data point is associated with a kernel function, which in total aim at approximating the uncertainty. The difference to static networks is that kernel functions are placed on the fly, their number and location in the state space are not fixed a-priori. However, just storing each point quickly leads to congestion of the storage capabilities. Consequently, Chowdhary et al. introduce a budget on the number of data points and elaborate on suitable methods of how to choose those points ([38]). GP MRAC repeals the need for update laws. Rather, the adaptive signal is directly computed from the predictive posterior distribution, which in turn is a result of Bayesian inference using the online gathered data. Furthermore, GP MRAC supplies a belief on the quality of the uncertainty approximation in the form of a variance.

In general, MRAC applications require high learning rates in order to achieve fast adaptation performance. However, this induces oscillations in the input channel, which can lead to undesirable system behavior or instability ([29]). $\mathcal{L}_1$ adaptive control ([97], [221]) was developed as a derivative of indirect MRAC in order to abrogate the undesirable result of high adaptation gains. It aims at decoupling robustness from the adaptation, i.e. control from estimation ([17, p. 163]). The underlying idea is to add a low pass filter to the input channel and therefore limit its bandwidth. As a result, high frequencies from high adaptation gains in the estimation loop are prevented from entering the plant. $\mathcal{L}_1$ adaptive control has been successfully applied to a huge variety of systems and problems including tailless fixed-wing aircraft ([166]), space launch vehicles ([119]) or degraded multirotor systems ([148]). However, note that $\mathcal{L}_1$ adaptive control focusses on the cancellation of the uncertainty pointwise in time, i.e. information is not conserved. If a certain region in the state space is revisited, the uncertainty needs to be relearned.

3.4 The Certification Challenge of Model Reference Adaptive Controllers

Despite the promising advantages of adaptive controllers to compensate uncertainties and react to unforeseen situations, they are not used in commercial aviation. This is mainly due to a missing decision on the correct means to routinely verify, validate and thus certify adaptive flight control algorithms (see i.a. [104], [105], [169], [170], [205], [220], [49], [187]).

Airborne systems are extensively regulated by governments. For the commercial use of aerospace systems, certification is required. The legal framework is provided by aviation authorities. Within the European Union, the European Aviation Safety Agency (EASA) is in charge of certification. Its counterpart in the United States is the Federal Aviation Adminsitration (FAA).

EASA provides the legal framework in order to certify aircraft. These are summarized in the so called certification specifications (CS). For normal, utility, aerobatic and commuter airplanes of a specific size CS23 ([47]) applies. The latter consists of two parts. The first book covers the airworthiness code, which specifies requirements the aircraft has to fulfill. The second book contains Acceptable Means of Compliance (AMC), which act as ways to show compliance with the stated requirements.

Paragraph CS23.1309 ([47]) states requirements for the certification of flight control algorithms. However, the second book in [47] does not state any AMC to comply with §1309. During the course of this thesis, the 5th Amendment of the CS23 was published ([47b]). The latter lifts numerous requirements and allows for new design solutions, which were previously not applicable. With this, safety is formulated in a more risk-based manner ([135b]). However, the freedom of design decisions also raises the question of how the development process shall be designed in the end.

For the former version [47], the EASA CM SWCEH-002 ([60]) states that the standard DO-178B/C ([176]) has been approved as an acceptable means of compliance. In general, the DO-178B/C provides "[…] guidance for the production of software for airborne systems and equipment that performs its intended function with a level of confidence in safety that complies with airworthiness requirements" [176, p. 1]. In order to achieve this, the DO-178B/C requires the precise, a-priori specification of the software by using a set of requirements ([87]). Suitable requirements can be found in e.g. the AS94900 ([181]), the ARP94910 ([180]) or in MIL-HDBK-1797 ([141]).

Verification and Validation (V&V) are key processes in the development of control algorithms and are intended to check for the correctness of the software. With regard to the DO-178B/C Jacklin states that "[…] verification is the process of testing the software at each stage of its development to make sure it has been programmed as specified in the software requirements document" [104, p. 2]. Furthermore, "[v]alidation comprises the testing effort to assure that the verified software is able to accomplish the purpose as stated in the software requirements document" [104, p. 2]. In the context of control algorithms, this includes guarantees about the performance and behavior of a controlled system. Thus, from an abstract point of view, the V&V process is intended to generate trust in a successful operation of the flight control software.

Linear control systems are designed with a fixed system configuration and a static environment in mind. Note, that this does not indicate the absence of disturbances. Rather, a model of the environment is selected, which is kept for the whole development process. The controller gains are selected to be constant and the complete system can be tested against suitable requirements. The latter are provided by i.a. the AS94900 and the ARP94910. Furthermore, robustness metrics, such as the gain and phase margin, help to determine the robustness of the closed loop system against changes within the specified environment. This analysis is performed a-priori to the controller deployment. Hence, guarantees and the expected behavior of the closed-loop system are known before the system is even used.

In contrast, adaptive systems adjust their parameters online. As a result they are able to compensate modeling uncertainties and unforeseen configurational or environmental changes. However, several technical and formal drawbacks exist that prevent the direct adaptation of the V&V process for linear systems to adaptive controllers (see i.a. [105]).

Guarantees for the convergence of the adaptive weights only exist for a small class of systems. They generally do not hold for systems subject to disturbance. Similarly, in the case that the uncertainty is to be approximated by an over-determined control structure, such as neural networks, the adaptive parameters do not converge either. As a result, the uncertainty is not cancelled uniformly and it can still affect the plant adversely. To account for this, a Lyapunov analysis is usually intended to demonstrate stability of the closed loop system (see e.g. [8], [155], [204]).

Even though the Lyapunov based stability proofs usually ensure boundedness of all system signals, these bounds are often utterly conservative, physically unreasonable and might cause a system to crash due to the violation of structural or operational constraints. Heise considerably tightens the analytically computed bounds in [87], but states that the results are still too conservative to be used universally. Furthermore, most stability proofs for adaptive control developments only take a part of the system into account. In particular, subsystems, such as actuator dynamics, and effects from the environment are often neglected. Hence, the stability proof might not even hold in the presence to small changes in the system or when transported to an experimental system.

A crucial point in the V&V process of linear flight controllers are analytically derived performance and robustness guarantees in the form of metrics, such as phase and gain margins. Unfortunately, such universal metrics do not exist for nonlinear controllers. Recently, the time-delay margin has been used for adaptive systems ([57], [158], [160]). It quantifies the time-delay the closed-loop system is able to sustain before becoming unstable. In this, it is similar to the phase margin in linear control theory. The analytic computation of the time-delay margin is only possible for simple systems. Apart from that, it can be estimated by performing numerical simulation. Hence, a concise a-priori analytical evaluation of the adaptive controller is not possible with the current state-of-the-art. The same argument is extendable to other metric types such as performance metrics.

Apart from the technical difficulties, also formal barriers prevent a successful certification of adaptive flight control algorithms. The existing support documents, such as the AS94900 and the ARP94910, are designed with linear control theory in mind. For instance, several requirements are formulated in the frequency domain. Since a learning based controller as presented in section 3.3.1, is inherently nonlinear, a frequency domain analysis is not applicable. Furthermore, the spirit of the DO-178B/C demands testing all requirements against all possible parameter combinations. Wilkinson ([220, p. 14]) and Dahm ([49, p. 13]) argue that an adaptive system can assume an infinite number of possible parameters. Even an active restriction of the adaptive weights to a preselected region (with e.g. parameter projection) does not solve this problem. In fact, it is not guaranteed that the worst case performance is obtained by using adaptive parameters on the boundary of the specified region. Consequently, the requirements would have to be tested against an infinite amount of parameter combinations.

Overcoming these drawbacks and providing a concise V&V process for complex adaptive systems is seen as one of the top challenges for autonomous systems ([157]). A variety of authors proposed extensions and alterations to the current certification process in order to include adaptive control algorithms. Due to the changing nature of adaptive controllers, there seems to be an emerging consensus to move the verification from an a-priori analysis to a verification during the runtime of the system (see i.a. [105], [169], [170], [205], [220], [187], [177], [178]). An integral part of run time verification is seen to be the establishment of an online monitoring tool that surveils the adaptive control algorithm. In fact, the new framework set by the 5th Amendment of the CS23 ([47b]) enables such an approach.

Even though the monitoring algorithm is envisioned to play an integral role for runtime verification, it is not intended to solve the certification problem of adaptive flight control algorithms alone. Rather, Schumann sees the monitor as an extension that still requires “intelligent validation” and a “static mathematical analysis” [187]. In that, the technique is not intended to find software failure and supersede a rigorous design and verification, but to evaluate the system. Amongst others, in [105] Jacklin lists additional gaps that need to be overcome in order to successfully certify adaptive flight control algorithms.

Against this background, the remainder of this thesis deals with the development of an online monitoring tool, which is intended to assess the performance of the adaptive controller during flight and indicate imminent off-nominal and undesired behavior.

4 Run-Time Monitoring of Adaptive Control Algorithms

An online monitor is seen as an integral part for future runtime verification of adaptive flight control algorithms by a variety of authors (see i.a. [105], [187]). Thereby, Taylor et al. state that a run-time monitor "[…] would need to be developed as part of the system, perhaps beginning as early as the requirement stage of the project" [205]. The question arises what a runtime monitor should be able to do and what information it should relay. Depending on the application, different points of view exist in the literature.

For Jacklin the online monitoring tool shall "monitor adaptive controller performance after deployment" [105]. Similarly, Phillips and Blackburn state that run time verification methods "[…] defin[e] a model or meta-model of acceptable system states and behaviors" [169]. Taylor et al. state that runtime monitors "[…] can provide fault detection and can identify divergent behaviors that could lead to failures" [205]. Other authors are more specific about their goals. Within the IRAC project, Krishnakumar expects the monitoring algorithm to find the "[t]ime (in seconds) to predict loss of stability as flight parameters change […]" [125, p. 27]. In [222] Yeramella lists the following questions the monitoring system has to answer: "Does online learning tend to diverge?", "Is it possible to detect this behavior?", "Does the learning converge to a stable state?", "How long does it take for the learning to do so?". Summarizing the mentioned statements leads to two common goals for the online monitor: First, detect diverging learning behavior of the adaptive controller. Secondly, predict imminent system failure before it occurs.

Some statements above primarily focus on the learning behavior of the adaptive control algorithm. Naturally, the evolution of the plant states is tightly coupled with the evolution of the adaptive parameters. As a consequence, neglecting one aspect in the monitor can lead to significantly wrong conclusions. On the one side, focusing only on correct learning neglects the impact of the control signal on the plant. E.g. even correct learning can lead to states exceeding operational or structural limits in transients. On the other hand, if the evolution of the adaptive parameters is neglected or oversimplified, its impact on the plant dynamics is captured wrongly. One such example is parameter drift. While the modeling uncertainty might be represented well, the adaptive parameters drift apart. Once the flight conditions change, the adaptive parameters lead to an enlarged control input, which can cause the system to crash.

The monitoring approach proposed in this thesis is guided by the structure of certification procedures. In that, undesired events are formulated with an acceptable probability of occurrence. These events are translated into requirements for the aircraft. The task of the control system is to control the state of the aircraft and ensure stability, robustness against disturbances as well as good tracking performance. Hence, from the control perspective, requirements on undesired events directly induce requirements on the system states. The goal of the proposed monitoring approach is therefore the certainty evaluation of near-future state requirement violations.

The monitor is envisioned to achieve this by using a model of the plant and parameter dynamics in order to predict the evolution of the system and controller states. The controlled system exhibits three properties, which place essential restrictions on the design of the monitor. First, real world systems are subject to stochastic uncertainties originating from i.a.

noisy state measurements, distributed system parameters or external disturbances. Secondly, the closed loop dynamics are highly nonlinear. Even if the plant was linear, the adaptive component still adds a nonlinear feedback to the control signal. Lastly, the monitor shall be implemented on unmanned aircraft. The latter exhibit fast dynamics, while the online available computational resources are usually low. As a direct consequence, these three conditions impose requirements on the monitoring approach. In particular, the monitor shall take the stochastic nature of the system into account, be able to predict the state and parameter distribution even if the underlying system is nonlinear, and be deployable on standard UAS hardware.

This chapter introduces a monitoring algorithm based on analytic parameter and state distribution forecast, which is envisioned to meet these requirements. The underlying idea is to model the initial state distribution with a GMM. The modelling uncertainties are represented by GPs. Propagating the uncertainty through the system reduces to a linearization of the nonlinear dynamics at each mixture component and evaluating the GPs at each mixture element, respectively. Since GMMs can assume the shape of arbitrary continuous distributions, they are envisioned to be suited for long-term propagation in nonlinear systems. The following sections detail the structure and design choices of the proposed monitoring technique.

The outline of this chapter is as follows. Section 4.1 surveys existing monitoring algorithms for adaptive controllers as well as techniques for uncertainty propagation and prediction applied to dynamical systems. Section 4.2 details the structure of the proposed monitoring strategy. Section 4.3 shows how to use GMMs in order to propagate uncertain states through a nonlinear system in the absence of modelling uncertainties. Section 4.4 shows how to include GPs in the GMM prediction in order to represent modeling uncertainties. Section 4.5 details the monitoring output as well as identifies errors occurring in the prediction and introduces a confidence measure for the forecast. Section 4.6 lists important implementation aspects. This chapter is concluded in section 4.7 with a discussion of the proposed monitoring algorithm including a distinction from existing approaches.

The fundamentals of probability theory are assumed to be known. However, the most important aspects are summarized in Appendices B and C. Preliminary results of this chapter appeared in [146].

4.1 Existing Monitoring and Prediction Strategies

This section first provides an overview of existing monitoring strategies for adaptive control algorithms. The second part surveys prediction algorithms as well as techniques for uncertainty propagation.

4.1.1 Existing Monitoring Strategies for Adaptive Controllers

There exists a vast literature of performance monitoring and assessment techniques for control applications. This is particularly true for the field of industrial applications and process dynamics. Since these approaches primarily focus on linear control design, they are omitted here. For the interested reader, three survey papers shall find attention here. Harris et al. gives an early overview [86] of performance monitoring for univariate and multivariate control systems. Qin assesses various control performance monitors in [109], including time-delayed systems. Jelali gives an overview of control performance assessment technologies in industrial applications, including statistical approaches ([107]). The following part focusses on existing monitoring strategies for adaptive controllers.

4.1.1.1 Evaluation of the Lyapunov Function

Taylor et al. sees potential in using the Lyapunov stability analysis during runtime ([205]). A straight forward approach is to employ the error bounds derived from the Lyapunov analysis in (3-31) and (3-32) or (3-50)-(3-52), respectively. However, as mentioned before, the Lyapunov analysis is utterly conservative. Furthermore, various properties aggravate the inference of conclusions on the state and parameter limits from the Lyapunov analysis alone. One such example is that the optimal adaptive parameters are usually unknown and can only be guessed.

4.1.1.2 Oscillation detection

In [44] Chowdhary et al. developed a technique, which approximates the frequency content of measured data by a finite Fourier basis. The underlying goal is to detect undesired oscillations and bursting caused by an adaptive control algorithm. In particular, Chowdhary et al. use a moving window of data and analyse it using the discrete counterpart of the Fourier transform. If the Fourier coefficients exceed a predefined threshold, a warning is issued. While this method was shown to successfully detect oscillations once they occur, predictive behavior of the system is not possible.

4.1.1.3 Monitors developed for neural networks

Various approaches to the monitoring of controllers with a neural network as an adaptive element exist. Leonard et al. ([129]) developed Validity Index Networks in order to determine, whether there is adequate training data in RBF neural networks. In essence, Validity Index Networks, observe the data density for each node in RBF neural networks. The local density allows for the derivation of confidence limits on the neural network output given the current input.

The Validity Index Networks approach is extended by Liu et al. in [133], [134], [132], [188] to Dynamic Cell Structures. Thereby, the input space to the network is fractured into Voronoi regions. Each region is associated with a neuron. Every data input is associated with the neuron closest and second-closest to it. The approach of Liu et al. then computes the local variance of the network residual for each region using only the locally classified data and neighbouring regions. From the local variance, Liu et al derive an estimate of the standard error of the network output.

In [222], [68], Yerramalla et al. use Lyapunov-like functions in order to detect deterioration from a previously classified stable learning behaviour. In particular, Yerramalla introduces four tests that evaluate the deviation between the specified behaviour and the current neural network performance for networks based on dynamic cell structures.

In [189], Schumann and Liu highlight the similarity between neural network and the Kalman Filter and use an *Extended Kalman Filter* (EKF) in order to update the adaptive weights. The covariance of the EKF is used as a measure of confidence in the learning.

In [189], [188] Schumann and Liu also suggest the use of a parameter sensitivity analysis on the adaptive weights. The approach can be extended by treating the network inputs as rvs. A decreasing sensitivity indicates convergence of the neural network parameters.

In [186], [186], [82], [188] the authors learn the input-output map of a RBF neural network using Bayesian regression for a moving window of data. If the true output of the adaptive controller significantly differs from the learned model, a warning is issued. The authors in [202] extend this method to include parametrized adaptive elements in general. In [150], this technique has been shown to detect diverging state trajectories of an adaptively controlled quadcopter. However, conclusions about when the state trajectories exceed a critical limit are not possible.

Rule extraction techniques follow the idea of pre-evaluating a neural network that has been trained a-priori ([50], [6], [5]). Based on the input data a set of mathematical rules describing the pre-trained neural network is extracted. The underlying idea is to use the derived rules online, in order to check their adherence by the evolving neural network given new data.

The mentioned approaches have shown to yield good performance if the adaptive element is a neural network. However, most of the approaches are restricted to neural networks only. Furthermore, while they detect undesired behaviour in the learning capabilities of the adaptive element, they don’t relay information about an imminent requirement violation. That is, a neural network might work as intended, but lead to a system state exceeding a predefined limit. Hence, an integral monitoring approach is necessary, which considers both, the adaptive elements as well as the state evolution it drives.

4.1.2 Approaches for Prediction and Uncertainty Propagation

The goal of the proposed monitoring technique is to predict the future state and parameter evolution based on a model of the system. This section surveys approaches, which could be used in model based prediction tasks and analysis their abidance of the formulated requirements for the monitor design. As a reminder, the latter comprise the need of the monitor to be applicable to systems, which are highly nonlinear, subject to stochastic uncertainty and exhibit limited computational power.

Prediction plays a vital role in a wide span of applications. The class of Bayesian Filters ([106], [83], [34]) usually is comprised of a prediction step followed by an update based on gathered measurement. This type of filter finds ample utilization in navigation modules, which estimate the states of dynamical systems (see i.a. [67]). Remaining useful life predictions are essential in manufacturing and the service industry, as they estimate the time till a product or tool degrades beyond usability ([162]). The underlying idea is to organize replacements proactively in order to not risk e.g. production standstill. An application of prediction encountered on a daily basis is weather forecasts ([136]). Lately, the strong interest in autonomous driving led to increased research in the estimation of future vehicle trajectories ([120], [85]).

Techniques for model based prediction can be loosely classified into deterministic and stochastic approaches. The former assume either an upper bound on the uncertainty, a deterministic uncertainty or no uncertainty at all. One such example is *Model Predictive Control* (MPC; see e.g. [13], [216], [117]). Based on the forecasted system dynamics, a control law is extracted, which abides desired tracking and constraint properties. MPC has been used for a variety of approaches, such as process dynamics, and UAV control ([149], [145]). On the other hand, stochastic approaches consider the stochastic nature of the uncertainty in the forecast. As a result, the prediction outcome itself is a rv. This approach allows the inclusion of the certainty of events into the forecast without the need of prior bounding of the uncertainty. Within the prediction scenario, uncertainty propagation is concerned with the question of how the distribution of the system states changes due to the dynamics and other effects. Note that this thesis uses the term stochastic prediction synonymously with the term uncertainty propagation.

The adaptively controlled system is subject to stochastic uncertainties. By imposing an upper bound on the uncertainty, deterministic approaches can still be applied. However, this would introduce a high level of conservatism into the forecast, which can lead to continuous predicted requirement violation, even though the system works as intended. Hence, this thesis focusses on techniques for stochastic prediction.

This thesis loosely classifies the family of uncertainty propagation techniques into two categories: Approaches that require sampling and analytic techniques. The following sections give a brief overview of both classes. This chapter uses the term ‘prior’ to indicate the pdf, which is propagated through the system. Consequently, the term ‘posterior’ indicates the result of the propagation. Note, that this is not contrary to the formulation of prior and posterior pdfs in other chapters, but mentioned for clarity.

4.1.2.1 Sampling based uncertainty propagation

Sampling based approaches, or Monte Carlo methods (see e.g. [77], [174], [89]), comprise a family of algorithms, which rely on sampling in order to generate a belief on the input-output relation of a system. In particular, samples are drawn from an initial probability distribution. In a second step, a deterministic operation, such as function propagation, is performed on the samples. The results are statistically analysed. The underlying idea is that if a sufficient number of samples are generated from the prior pdf, the propagation result forms a new pdf, which expresses the effect of the system on the input distribution. This way, inherently nonlinear systems and non-Gaussian probability distributions can be evaluated.

Monte Carlo methods span a wide variety of applications including the capability analysis of aircraft ([135]), medical research [175] and financial engineering ([24]). Several modifications of the classical Monte Carlo idea exist. Most famously, sequential Monte Carlo shall be mentioned here as it forms the basis for the famous particle filter. Amongst others, the latter has been applied for online fault diagnosis and failure prognosis ([163]).

Even though Monte Carlo methods are applicable to nonlinear and non-Gaussian systems, they exhibit two major disadvantages. First, it is usually not known a-priori how many samples are required in order to approximate the target distribution. Secondly, they are computationally expensive as a huge number of samples usually need to be drawn in the first place in order to represent the probability distribution well. Thus they do not abide the third requirement on the online monitor, which assumes low computational power on the monitored system. As a consequence, they are not further considered in this thesis.

4.1.2.2 Analytic uncertainty propagation

Analytic uncertainty propagation summarizes the family of techniques that calculate the posterior pdf or its moments in an input-output system directly without relying on sample based inference. In this, the term ‘analytic’ refers to the intent of obtaining a closed-form solution for the posterior distribution (see e.g. [53]). The foundation of the uncertainty propagation follows a Bayesian formulation ([218]). This means, the relation between the predicted posterior and the prior pdf is modelled using Bayes’ Theorem. The applicability of these techniques strongly depends on the properties of the system as well as the prior distribution.

One of the most famous approaches, which rely on this concept is the Kalman Filter ([115], [144], [106]; Appendix E.1). The Kalman Filter consists of two parts, a prediction step and a correction step. The prediction step uses a model of the system in order to predict the state distribution after one time step. The correction step updates the prediction by a new measurement. The Kalman Filter has been applied to a huge variety of systems, including object tracking ([45]), navigation ([143]) or finance ([217]).

The Kalman Filter relies on a normally distributed prior pdf and linear system dynamics. In the prediction step, the prior pdf is propagated through the linear system, resulting in a normally distributed posterior. A logical thought arising from the two-step approach is to omit the correction step and recursively apply the prediction step, in order to arrive at a model based state prediction. While this applies in principle, real-world systems are seldom linear or affected by normally distributed uncertainties. In particular, the negligence of the system nonlinearity quickly leads to huge prediction errors as the latter are propagated through the system. Several extensions exist, which try to overcome these disadvantages.

The prediction step of the EKF (see e.g. [106]) represents one of the most famous advancements of the Kalman Filter. The underlying idea is to linearize the nonlinear system around the mean of the prior pdf. The resulting system is linear and used in order to propagate the state pdf. As a result, the posterior pdf is also normally distributed. The approach has shown good performance in various applications. However, some disadvantages arise. First, the linearized surrogate model has to be valid over most of the pdf of the prior. If this is not guaranteed, the linearized model needs to be averaged over all surrogate models governed by the prior distribution. Secondly, the EKF still assumes the prior to be normally distributed. This assumption usually does not hold for long-term predictions (see Example 4-2 in section 4.2).

To overcome the disadvantages of the EKF, the GMM filter ([199]) approximates the prior pdf with a GMM. A GMM is able to represent non-normal distributions allowing to deal with nonlinear state evolutions. The underlying idea is to linearize the nonlinear system dynamics at the mean of each mixture component. Each mixture is then propagated through its respective surrogate model ([206]). The posterior is again a GMM. The hypothesis is that the support of the mixture components is small enough in order to ensure validity of the single linearized models. Since GMMs are applicable to nonlinear systems and are able to deal with non-normal state distributions, they will play a key role in the proposed monitoring algorithm.

Various other alternatives to the basic Kalman Filter algorithm and its prediction step exist. Most notably, this includes the Sigma Point Kalman Filter [211] or the Ensemble Kalman Filter ([116]). However, these techniques require sampling at different points of the algorithm and are hence neglected in this thesis.

The propagation approaches mentioned above implicitly require the system dynamics to be known. However, this might not be the case. Recently, GP models for uncertainty modelling attracted increased attention ([215], [54], [53], [59]). Thereby, the system dynamics are represented by a GP, which is either available in form of a prior model or inferred from online gathered data. Propagating a normally distributed state distribution through a GP becomes analytically intractable ([53], [71]).

Candela showed in [171] how to analytically and exactly calculate the first two moments of the posterior if the prior pdf is Gaussian and the GP model is represented by squared exponential kernels. This technique has been applied for prediction ([85]) and filtering problems ([53]).

A combination of GPs and GMMs emerges as a promising candidate for the basis of the proposed monitoring approach as it fulfils the three main requirements. First, GPs are able to take the stochastic nature of the system dynamics into account. Secondly, GMMs can be applied to propagate the system state distribution through nonlinear dynamics. Lastly, the reliance on analytic uncertainty propagation promises to meet the requirement on low computational effort.

4.1.2.3 Additional uncertainty propagation techniques

The previous sections gave an overview of techniques for sampling based and analytic uncertainty propagation. Additionally, techniques exist, which, dependent on their manifestation, rely on either analytic or sampling based techniques, but cannot be classified into one field. Two methods shall find attention here, the sensitivity analysis and polynomial chaos.

The sensitivity analysis "[…] investigates the connection between inputs and outputs of a (computational) model" [98, p. 50]. In particular, it analysis the impact a change in the input has on the output. Here, an input can also refer to a change of the system parameters. Within the context of sensitivity analysis, multiple methodologies exist. A famous approach is to calculate local sensitivity by analysing the partial derivatives of the system dynamics. A special kind is the stochastic sensitivity analysis ([62, 103]). Here, the input is assumed to follow a probability distribution. Sampling based approaches rely on sampling the input distributions ("one at a time") and propagating them through the system. The results for e.g. the scatter plot approach ([33]) are sensitivities, which are encoded in scatterplots and can be further analysed. In contrast, variance-based methods ([182]) aim at decomposing the output variance into parts, which can be attributed to the individual inputs. The sensitivity analysis allows the generation of good understanding for a dynamical system. However, it suffers from two main disadvantages. First, the sensitivity analysis relies on static system dynamics. However, adaptive systems inherently change over time. Furthermore, inverse approaches, such as GP regression, allow updating the system model during its runtime. Secondly, to get a good understanding of nonlinear systems, the complete input space needs to be taken into account. However, adaptive controllers assume an infinite amount of possible parameter combinations. This renders testing against a large amount of such combinations during the system runtime infeasible.

Polynomial chaos originates from the field of fluid dynamics and was intended to model turbulence ([219]). In particular, polynomial chaos aims at analysing the evolution of the state distribution in a dynamical system, which exhibits uncertain parameters ([153]). One technique to achieve this is *Polynomial Chaos Expansion* (PCE). PCE "[…] is a way of representing an arbitrary random variable of interest as a function of another random variable with a given distribution, and of representing that function as a polynomial expansion" ([161, p. 3]). The underlying basis of the function is chosen to consist of orthogonal weighted polynomials with respect to the distribution of the rv. As a consequence each polynomial is a rv. Since they are constructed to be orthogonal, the polynomials are uncorrelated.

To propagate a PCE, the output of the system dynamics is constructed as a PCE with the same polynomials as the input rv. The prediction step amounts to finding the weights of the output PCE. In general, approaches to solve this problem are classified into intrusive and non-intrusive methods. The former apply e.g. the Galerkin projection to find the weights of the output PCE (see e.g. [164]). However, this requires knowledge about the system dynamics. In contrast, non-intrusive approaches rely on sampling and use i.a. Monte Carlo or Bayesian Regression to calculate the weights.

While PCE has found increased attention in recent years, it exhibits some disadvantageous properties.. First, non-intrusive approaches to retrieve the output PCE weights rely on sampling, which is computationally expensive. As a result, intrusive approaches are favoured. However, O'Hagan argues that intrusive methods require a high complexity to model the system and are not easily adapted to changes ([161]). Furthermore, Lu et. al argue in [138] that PCE approaches for inverse problems can be extremely inaccurate. Hence, they tend to be inadequate for inferring a model from online gathered data. As a result, the mentioned properties render the current form of PCE inapplicable for the presented monitoring problem.

4.2 Proposed Monitoring Strategy

This section introduces the proposed system monitor and elaborates on its key components. The underlying idea of the monitor is to predict the state and parameter evolution in order to assess future behaviour of the closed loop system. Figure 4-1 depicts the proposed monitoring strategy. Starting with a new measurement at each time step, the monitor predicts the state and weight trajectories for a certain future time frame, the prediction horizon. Measurements suffer from noise and thus the initial state for the prediction is often corrupted. However, the pdf of the installed sensors is often well known by the manufacturers. Hence, instead of a deterministic forecast, the proposed monitoring algorithm leverages stochastic propagation techniques in order to predict probabilistic state and weight trajectories. Within the prediction horizon, the monitor then assesses with which probability a-priori formulated state requirements are violated. If the probable requirement violation exceeds a certain limit, the monitor issues a warning such that the pilot or the control system can initiate counter measures.

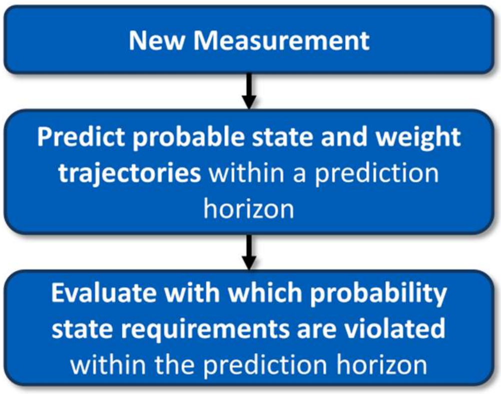

New Measurement
Predict probable state and weight trajectories within a prediction horizon
Evaluate with which probability state requirements are violated within the prediction horizon

Figure 4-1: Proposed Monitoring Strategy

In order to perform predictions, the monitor employs the closed loop plant dynamics (3-13), the reference model dynamics in (3-7) and the parameter update law in (3-19) (or (3-33) and (3-38) respectively). If the initial values $\left(\boldsymbol{x}_P(t_0), \widetilde{\boldsymbol{\Theta}}(t_0)\right)$ are exactly known and the system is not subject to disturbances or uncertainties, the state and parameter trajectory prediction is obtained by recursive evaluation of the mentioned models. However, the initial state $\boldsymbol{x}_P(t_0)$ is often not exactly known in a real system, due to measurement noise and only the output $\boldsymbol{y}(t_0) \in \mathbb{R}^n$ with

$$\boldsymbol{y}(t_0) = \boldsymbol{x}_P(t_0) + \boldsymbol{\varepsilon} \tag{4-1}$$

is available. Here, $\boldsymbol{v}$ denotes the measurement noise, which is assumed to be iid. normal with zero mean such that $\boldsymbol{\varepsilon} \sim N(0, \sigma_n^2 \boldsymbol{I})$. Here, σ_n denotes the standard deviation of the measurement noise, which can for example be obtained from data sheets of the respective sensor. As a result, the initial value of the prediction is normally distributed with $\boldsymbol{y}(t_0) \sim N(\boldsymbol{x}_P(t_0), \sigma^2 \boldsymbol{I})$. Note that this formulation equals the definition of the output equation in the regression model (2-20). Note that stochastic uncertainties acting on the plant are not exclusive to the measurement noise. Further stochastic uncertainties originate i.a. from process noise or probabilistically modelled disturbances such as turbulence.

Let $\boldsymbol{x}_s$ denote the vector combining all system and controller states. For the direct MRAC case this includes the plant states $\boldsymbol{x}_P$ as well as the adaptive parameters $\boldsymbol{\Theta}$. In the PMRAC case, $\boldsymbol{x}_s$ also contains the predictor states $\widehat{\boldsymbol{x}}_P$. Let $\boldsymbol{x}_{s,k}$ denote $\boldsymbol{x}_s(t_k)$ at the time instant t_k. Predicting $\boldsymbol{x}_{s,k+1}$ comes down to evaluating the Chapman-Kolmogorov equation (see e.g. [4]):

$$p(\boldsymbol{x}_{s,k+1}) = \int p(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})p(\boldsymbol{x}_{s,k})d\boldsymbol{x}_{s,k}. \tag{4-2}$$

Note, that placing a probability distribution on $\boldsymbol{x}_{s,0}$ represents an abuse of notation, as e.g. the adaptive parameters as well as the predictor states are exactly known at the initial time. For $k > 0$, the adaptive parameters as well as the predictor states are uncertain. The conditional pdf $p(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})$ corresponds to the evaluation of the system dynamics in (3-13). If the system (3-13) was linear with fixed adaptive gains, a recursive prediction could be achieved straight forward by propagating the state distribution through the linear dynamics. The basic idea is that the linear transformation of a normally distributed rv is still normal. This implies that both $p(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})$ and $p(\boldsymbol{x}_{s,k})$ are normal pdfs. As a result also $p(\boldsymbol{x}_{s,k+1})$ is normal. A normal distribution is exactly described by its first two moments. Hence, predicting state and parameter trajectories reduces to a recursive application of the summation and multiplication rules for normal distributions (see Appendix C.4). This paradigm is also the basis for the classical Kalman Filter algorithm.

However, by using the nonlinear closed loop system (3-13) and the parameter update laws of (3-19) or (3-38) for predictions two major problems arise. First, the modelling uncertainty $\boldsymbol{\Delta} = \boldsymbol{\Theta}^{*T}\boldsymbol{\varphi}$ is usually unknown. It can have a significant impact on the plant dynamics and even cause the system to crash. Usually, its impact depends on the quality of the plant model. E.g., for civil transport aircraft, significant effort is put into establishing a model, which represents the real plant well. Exemplary uncertainties for this kind of model are e.g. parameter uncertainties in the aerodynamic coefficients. In contrast, for small UAV significantly less effort is put into establishing a good model. Hence, the encountered uncertainties are expected to have a serious impact on the plant dynamics. Without knowledge about the uncertainty $\boldsymbol{\Delta}$, predictions of the state and parameter trajectories will be significantly flawed.

Secondly, the system in (3-13) is highly nonlinear. Note, that even if the regressor function $\boldsymbol{\varphi}$ was linear, the multiplication with the time varying adaptive weights renders the whole term nonlinear. A linear system is only achieved if the regressor represents a state independent constant, i.a. a bias. The propagation of a normally distributed rv through a nonlinear function does not necessarily result in a normally distributed posterior. Rather, the resulting rv can be extremely different to a rv with a normal distribution. This implies that $p(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})$ in (4-2), and therefore $p(\boldsymbol{x}_{s,k+1})$, are not necessarily normal pdfs. The resulting pdf $p(\boldsymbol{x}_{s,k+1})$ could still be exactly described by its moments. However, usually it is unknown how many moments are necessary. Furthermore, their exact calculation may be difficult or even impossible. Example 4-1 illustrates the former arguments, by simulating a nonlinear plant with normally distributed initial system state. Furthermore, it approximates the propagated rv by a normally distributed one in order to highlight that treating $p(\boldsymbol{x}_{s,k+1})$ as a normal despite better knowledge can result in extreme errors.

Example 4-1: **Trajectory distribution for a nonlinear simulation**

Consider the simplified roll dynamics of a fixed-wing aircraft:

$$\dot{p}_B = B_P\big(u - \Delta(p_B)\big). \tag{4-3}$$

Here, p_B denotes the roll rate and u represents the system input, which is given by the aileron deflection in reality. The input matrix B_P represents the roll moment generated by an aileron deflection. Assume a generic uncertainty of the form

$$\Delta(p_B) = \Theta^* cos(4p_B). \tag{4-4}$$

The input p_B to the uncertainty is assumed to be given in degrees per second. Consider a reference model of the form (3-7) as well as the control law in (3-11) and the direct MRAC update laws with σ-modification of (3-19). Table H-1 in Appendix H.1 lists all relevant parameters in order to recreate the simulation.

The simulation runs for a total of $12s$ with a time step of $0.001s$. The reference command is held constant at $r = 30\frac{\circ}{s}$, the initial condition of the reference model is $p_{rm}(t_0) = 0\frac{\circ}{s}$. The initial condition of the plant states follows a normal distribution with $p_B(t_0) \sim N\left(-20\frac{\circ}{s}, \left(5.73\frac{\circ}{s}\right)^2\right)$. To generate a distribution over the state trajectories, the simulation is repeated 2000 times.

Figure 4-2 shows the reference command, the state of the reference model as well as the evolution of the state trajectories over the simulation horizon. Thereby, the plot depicts the area in which 67%, 90% and 99% of all state trajectories reside in at each time instant. It can be seen that these area shrink and expand but generally approach the reference model state over the course of the simulation. This indicates that the single state trajectories track the reference model. Most importantly, the areas are seen to be asymmetrical. This indicates that the distribution over the trajectories significantly differs from a Gaussian distribution.

Figure 4-3 shows the state trajectory distribution as a histogram at selected time instants, namely after $0.4s$, $2.0s$, $8.0s$ and at $12.0s$. Furthermore, the approximating normal distribution is depicted. The later originates from taking the mean and variance of the state trajectory distribution at each time instant. It indicates the result if the data is interpreted to be normally distributed. It can be seen that even after $0.4s$ the state distribution significantly differs from a normal distribution. Furthermore, the distribution is multimodal. During the simulation the distribution becomes unimodal again, but skewed as the largest mode changes from the right side to the left side. At the end of the simulation it becomes multimodal again. In each case, the true distribution is badly approximated by a single normal distribution. This indicates that significant errors could be made by neglecting the nonlinear nature of the system during the prediction.

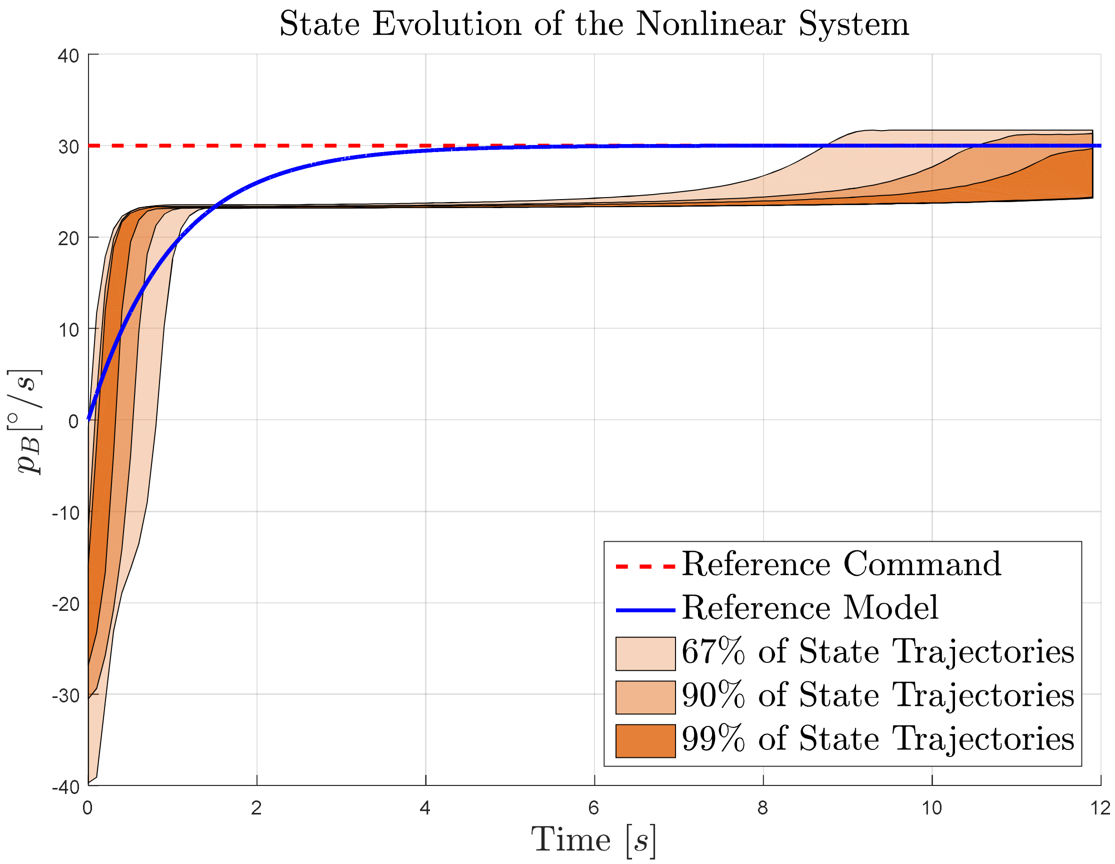


Figure 4-2: Areas where 67%, 90% and 99% of all state trajectories reside in

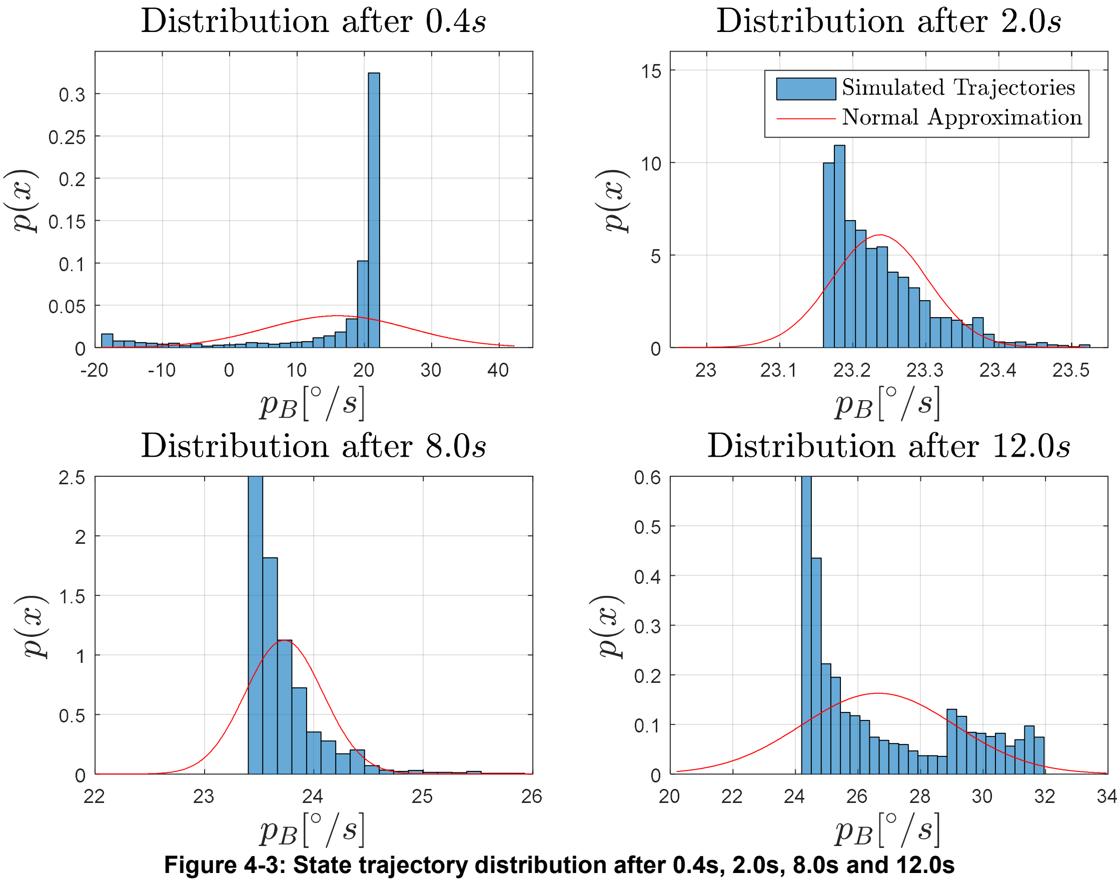


Figure 4-3: State trajectory distribution after 0.4s, 2.0s, 8.0s and 12.0s

In the absence of the residual error $\widetilde{\mathbf{\Theta}}^T\boldsymbol{\varphi}$ the plant in (3-13) becomes linear and independent of the adaptive parameters. Then, the theory about the propagation of normally distributed rv through linear functions can be applied. Hence, a first attempt would suggest at focusing on the impact of the residual error $\widetilde{\mathbf{\Theta}}^T\boldsymbol{\varphi}$ on the otherwise linear plant dynamics. However, the optimal parameters $\mathbf{\Theta}^*$ are unknown and both, the parameter error $\widetilde{\mathbf{\Theta}}$ and the regressor vector $\boldsymbol{\varphi}$ are time varying. Instead, assessing the adaptive signal $\boldsymbol{u}_{ad} = \mathbf{\Theta}^T\boldsymbol{\varphi}$ and the uncertainty $\mathbf{\Delta} = \mathbf{\Theta}^{*T}\boldsymbol{\varphi}$ separately exhibits some advantages, which are discussed in the following.

The initial value of the adaptive signal $\boldsymbol{u}_{ad}$ is exactly known, since it is generated in the flight control computer. Furthermore, the regressor function $\boldsymbol{\varphi}$, as well as the update law in (3-19) or (3-38), are known. The problem of predicting $\boldsymbol{u}_{ad}$ reduces to the evaluation of known nonlinear functions for a rv $\boldsymbol{x}_s$ with known initial distribution. Several techniques to propagate a rv through a nonlinear function exist. One of the most famous representatives is the EKF ([106]). The prediction step linearizes the nonlinear dynamics around the current mean of the state distribution. The result is used to propagate the state. As a consequence, $p(\boldsymbol{x}_{s,k+1})$ will represent a normal pdf. Again, predicting state and parameter trajectories reduces to a recursive application of the summation and multiplication rules for normal distributions. While the EKF has shown good performance in various applications, some disadvantages arise from the linearization. First, the linearized model needs to be valid over most of the pdf of $\boldsymbol{x}_s$. Second, the EKF assumes a normally distributed state $\boldsymbol{x}_s$. While, this may be approximately true for one prediction step, it usually does not hold for a long-term forecast. Example 4-2 gives insight into the former arguments by propagating a normally distributed rv through a sine function using the linearization approach.

To overcome these disadvantages the proposed algorithm employs GMMs as introduced in section 2.1. The basic idea is to approximate the initial state distribution $p(\boldsymbol{x}_s)$ ($p(\boldsymbol{y})$ respectively) with a weighted sum of multiple normal distributions such that

$$\hat{p}(\boldsymbol{x}_s) = \sum_{i=1}^{m} \alpha_i N(\boldsymbol{\mu}_i, \mathbf{\Sigma}_i). \tag{4-5}$$

Here, $\boldsymbol{\mu}_i$, $\mathbf{\Sigma}_i$ denote the mean and covariance of the i-th mixture component. The a-priori known nonlinear functions $\boldsymbol{\varphi}$ are approximated by a first order Taylor series expansion at the means of each GMM component, instead of linearizing the nonlinear function at one point only. The underlying idea is that if the largest part of the support of the mixture components is in turn small enough, the validity of the single Taylor series approximations can be preserved. An important property is that the former is designer chosen by selecting the standard deviation of each component during the approximation of the initial state distribution. During the prediction, each component of the GMM is then propagated linearly. As a result, $p(\boldsymbol{x}_{s,k+1})$ will be represented by a GMM again. Still, a GMM is able to also represent non-normal distributions allowing to deal with nonlinear state evolutions. This strategy has been successfully used i.a. in GMM filtering ([199]).

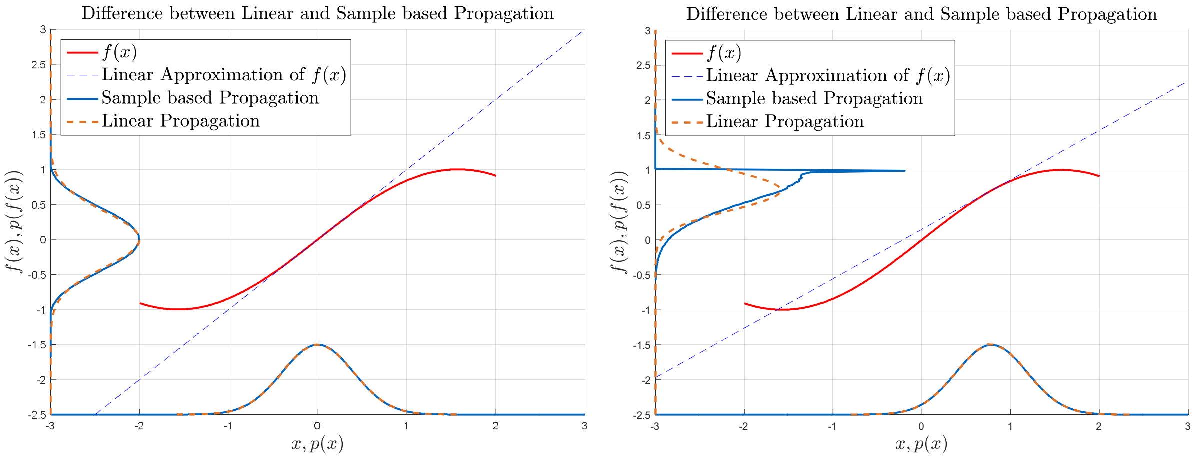


Figure 4-4: Difference between linear and sample based propagation

Example 4-2: Propagation of a normally distributed random variable through a sine function by sampling and linear approximation

Let X be a random variable with a normal pdf such that $X \sim N(\mu, \sigma^2)$. The standard deviation is chosen to be $\sigma = 0.4$. Furthermore, let $f(x) = sin(x)$ be a nonlinear function, which the random variable is propagated through. In order to generate the results for the nonlinear propagation, 100000 samples are taken from $N(\mu, \sigma^2)$ and individually propagated through $f(x)$. In turn, for the linear approximation, the function $f(x)$ is approximated by a first order Taylor series at the mean μ. Two cases with different means are considered. In the first case a mean of $\mu = 0$ is chosen. In the second case the mean is shifted by $\frac{\pi}{4}$. In order to compare the results, the Kullback-Leibler-divergence is used ([127], [126]; see Appendix B.9).

Figure 4-4 shows the difference between a nonlinear and a linearly approximated propagation for the two cases. Thereby, the distributions on the x −axis show the prior pdf of x, whereas the distribution on the y −axis depicts the predictive. Figure 4-4 (left) depicts the first case. A first order Taylor series approximates the sine function well within an interval of $(-0.75 \ldots 0.75)$. About 95% of the prior pdf lies within this area. Hence, it is not surprising that the resulting predictive distribution is well approximated by a normal. Note, that the approximation is not perfect. As a result, a residual error remains. The Kullback-Leibler divergence between the linearly and nonlinearly propagated posterior is $D_{KL} = 1.42$.

Figure 4-4 (right) depicts the second case. A first order Taylor series approximates the sine function well within an interval of $(0.60 \ldots 1.00)$. However, this amounts to less than 50% of the prior pdf lying within this area. Hence, the resulting predictive distribution shows significant difference to a normal pdf. The Kullback-Leibler divergence for the second case is $D_{KL} = 26.58$ and therefore around 18 times higher when compared to the first case.

The presented example showed that the approximation of the predictive as a normal pdf is well warranted in the first case. However, a simple shift of the prior mean can render this strategy unreliable. A similar result is obtained if the standard deviation is varied instead of the mean. In the presented example, decreasing the former leads to a better estimation of the predictive as a normal pdf. This is due to an increasing area of the pdf lying within the part of the nonlinear function, which is well approximated by a first order Taylor series.

In order to incorporate the uncertainty $\boldsymbol{\Delta}$ in the prediction, the proposed monitoring approach models $\boldsymbol{\Delta}$ by a GP as introduced in section 2.2. In that, it considers two cases. First, remember that for a variety of applications a-priori information about $\boldsymbol{\Delta}$ is available in the form of a prior GP model. One such case is the C_L over α model of Example 2-2, which is inferred by extensive experimental analysis. The underlying idea is to evaluate such a prior model at each mixture component of the GMM. This is achieved by employing the calculation rules in section 2.3.1. One key aspect is that the technique in section 2.3.1 calculates the first two moments of the propagated distribution exactly. By neglecting higher order moments, the GMM retains its structure. Similar to before, the underlying idea is that if the mixture components are narrow enough, the approximation of the propagated component by a normal is justified. This strategy has been successfully used i.a. in k-step ahead prediction problems for GPs representing car dynamics ([85]).

The second case assumes no prior knowledge about $\boldsymbol{\Delta}$. Remember that the optimal parameters $\boldsymbol{\Theta}^*$ are unknown but assumed to be time invariant and that the regressor vector $\boldsymbol{\varphi}$ is known. Hence, if a sufficient number of measurements of $\boldsymbol{\Delta}$ were available, the application of regression techniques could lead to an estimate of $\boldsymbol{\Delta}(\boldsymbol{x}_P)$. However, calculating only the most probable answer for $\boldsymbol{\Delta}(\boldsymbol{x}_P)$ under the given data is not sufficient. Rather, also a confidence in the estimate is required. Otherwise, an inaccurate estimate of $\boldsymbol{\Delta}(\boldsymbol{x}_P)$ can lead to predicted trajectories, which are not conservative enough and disregard an impeding requirement violation. One such technique is GP Regression (see i.a. [173], [21], [152]) as introduced in section 2.2.1. It constructs a stochastic model of $\boldsymbol{\Delta}(\boldsymbol{x}_P)$ and uses measured data points in order to update it. The model can then be evaluated at points of interest using the calculation rules in section 2.3.2. The benefit of GP regression is that it does not only supply a most likely estimate, but also provides a level of confidence in the form of a predictive posterior variance.

The proposed approach relies on a serious of approximations. The most important ones include the linearization of the plant dynamics and the approximation of the evaluated GP as a normal distribution. In contrast to filter algorithms, pure predictions cannot be corrected based on incoming measurements. As a result, errors occurring due to the mentioned approach, such as approximation inaccuracies, are propagated through the system. While the result of the algorithm might still indicate that no requirement violation is imminent, built-up propagation errors can significantly falsify this assessment. In order to deal with this arising problem, a measure of confidence is necessary. This measure is intended to relay information about how certain the result of the algorithm is based on the encountered errors.

Figure 4-5 depicts the detailed strategy of the proposed monitoring algorithm incorporating the preceding arguments. Based on a new measurement, if no prior model is available, the GP model $\boldsymbol{\Delta}$ is updated using Bayesian inference. Three steps are performed in each prediction step. First, the uncertainty model is evaluated for each mean of the current GMM components. Second, the plant dynamics including the adaptive controller are linearized at the same means. At last, the mixture components are propagated through the linearized as well as the GP model. The updated state distribution recursively serves as a new input. These steps are repeated within the prediction horizon. Afterwards, the algorithm assesses the probability of a requirement violation within the prediction horizon. At last, a confidence measure, which incorporates propagation and approximation errors, is calculated.

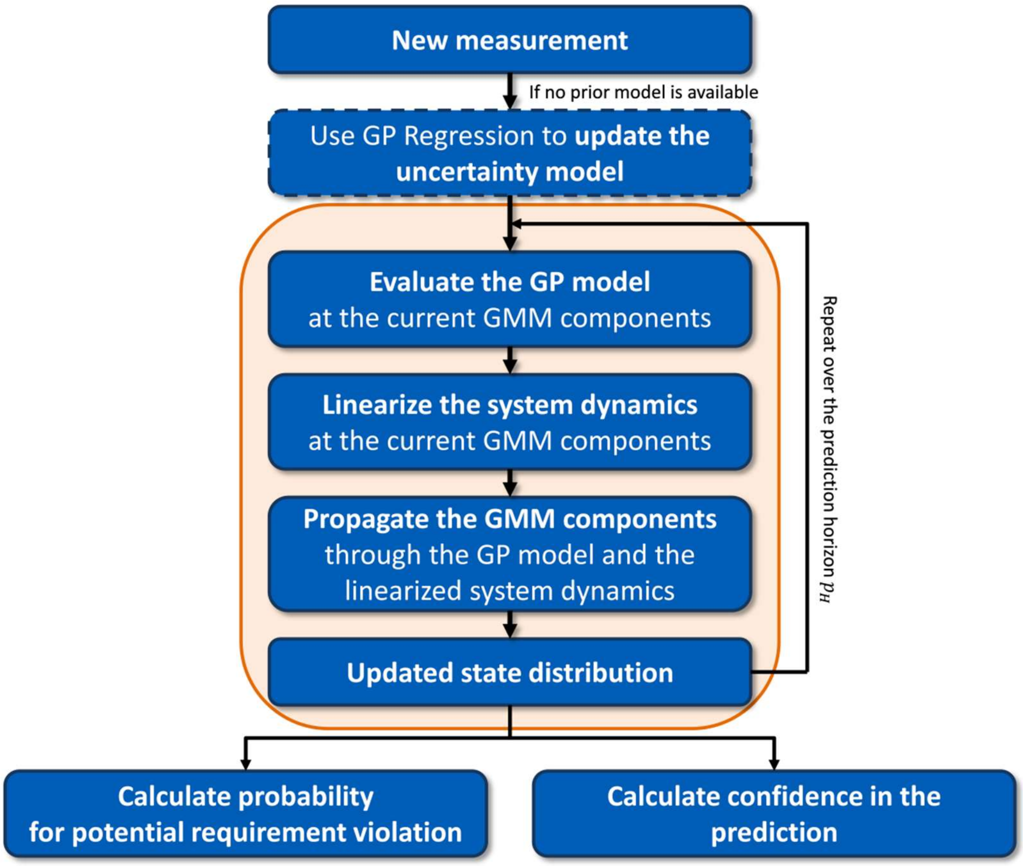

New measurement
If no prior model is available
Use GP Regression to update the uncertainty model
Evaluate the GP model at the current GMM components
Linearize the system dynamics at the current GMM components
Propagate the GMM components through the GP model and the linearized system dynamics
Updated state distribution
Repeat over the prediction horizon p_H
Calculate probability for potential requirement violation
Calculate confidence in the prediction

Figure 4-5: Proposed strategy for state and parameter monitoring of Model Reference Adaptive Controllers

The following sections elaborate on the single parts of the approach in detail. In particular, section 4.3 depicts how to use GMMs in order to propagate the state distribution through the nonlinear plant model including the adaptive controller. Section 4.4 shows how to employ GP regression for uncertainty quantification and GMM propagation. Section 4.5 focusses on the indicator functions, including potential imminent requirement violation and a confidence measure in the predictive performance of the monitoring algorithm.

4.3 Nonlinear Uncertainty Propagation using Gaussian Mixture Models

As shown in Example 4-2, even if the prior is normal, the posterior distribution of nonlinear function propagation is often ill modelled by a Gaussian density. This is due to the fact, that an approximation of the posterior as a Gaussian distribution inherently assumes a linear function propagation. Even though this might be approximately true if the nonlinear function is well approximated by a first order Taylor series over a large domain of the prior pdf, it doesn't hold in general. Hence, a long term prediction of a nonlinear system, which is linearized at a single point, can lead to significant errors.

To account for this problem, this thesis uses GMMs in order to model the state distribution. GMMs promise to yield a better estimate of the predictive distribution. This is due to the fact that a GMM is able to approximately represent a large class of distributions with a continuous pdf. Hence, it is also expected to be able to approximately model the predictive distribution arising from a nonlinear propagation. To achieve this, the known normally distributed prior at t_0 is approximated by a GMM. The single mixture components have a smaller standard deviation than the original prior. Hence, the validity region of a linearly approximating function can be smaller. Then, the system dynamics are linearized at the expected value of each mixture using a Taylor series expansion. Each mixture element is propagated through its respective linear surrogate models. Combining all propagated mixture components yields an approximation of the predictive distribution.

The outline of this section is as follows. Section 4.3.1 shows how to linearize the nonlinear closed loop dynamics of the adaptively controlled system. Section 4.3.2 shows how to propagate the mixture elements of a GMM through the linearized system.

4.3.1 Linearization of the adaptively controlled closed-loop system

This section constructs a linear surrogate model of the plant dynamics of the adaptively controlled closed-loop system using a Taylor series expansion. The employed calculation rules for matrix derivatives are summarized in Appendix A.7. Due to the proposed strategy, the uncertainty propagation of the nonlinear system dynamics and the adaptive controller is decoupled from the modelling uncertainty $\boldsymbol{\Delta}$. Furthermore, the plant dynamics as well as the parameter update dynamics can be analysed separately. Hence, first only the reduced right hand side of the closed-loop dynamics (3-13) is considered:

$$\boldsymbol{f}_x = \boldsymbol{A}_P\boldsymbol{x}_P + \boldsymbol{B}_P\boldsymbol{K}_e^T\boldsymbol{e} + \boldsymbol{B}_P\boldsymbol{K}_{rm,x}^T\boldsymbol{x}_{rm} + \boldsymbol{B}_P\boldsymbol{K}_{rm,r}^T\boldsymbol{r} + \boldsymbol{B}_P\boldsymbol{\Theta}^T\boldsymbol{\varphi}(\boldsymbol{x}_P). \tag{4-6}$$

Remember that the reference model is designer chosen. Since $\boldsymbol{r}$ is assumed to be constant (or following a known evolution) over the prediction horizon, the evolution of $\boldsymbol{x}_{rm}$ can be calculated exactly. Hence, both quantities are considered as constant inputs with regard to the Taylor series. A Taylor series expansion at $\boldsymbol{x}_P = \boldsymbol{x}_{P,l}$ and $\boldsymbol{\Theta} = \boldsymbol{\Theta}_l$ results in

$$\boldsymbol{f}_x = \boldsymbol{f}_x\left(\boldsymbol{x}_{P,l}, \boldsymbol{\Theta}_l\right) + \left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} \left(\boldsymbol{x}_P - \boldsymbol{x}_{P,l}\right) + \left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} \left(\boldsymbol{\Theta} - \boldsymbol{\Theta}_l\right) + O^2(\boldsymbol{x}_P, \boldsymbol{\Theta}). \tag{4-7}$$

Here, $O^2(\boldsymbol{x}_P, \boldsymbol{\Theta})$ denotes higher order terms. Furthermore, $\boldsymbol{f}_x(\boldsymbol{x}_{P,l}, \boldsymbol{\Theta}_l)$ is

$$\begin{gathered}\boldsymbol{f}_x(\boldsymbol{x}_{P,l}, \boldsymbol{\Theta}_l) = \boldsymbol{A}_P \boldsymbol{x}_{P,l} + \boldsymbol{B}_P \boldsymbol{K}_e^T(\boldsymbol{x}_{P,l} - \boldsymbol{x}_{rm}) + \boldsymbol{B}_P \boldsymbol{K}_{rm,x}^T \boldsymbol{x}_{rm} \\ + \boldsymbol{B}_P \boldsymbol{K}_{rm,r}^T \boldsymbol{r} + \boldsymbol{B}_P \boldsymbol{\Theta}_l^T \boldsymbol{\varphi}(\boldsymbol{x}_{P,l})\end{gathered} \tag{4-8}$$

The partial derivative of (4-6) with respect to the system states $\boldsymbol{x}_P$ is

$$\left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P = \boldsymbol{x}_{P,l} \\ \boldsymbol{\Theta} = \boldsymbol{\Theta}_l}} = \boldsymbol{A}_P + \boldsymbol{B}_P \boldsymbol{K}_e^T + \boldsymbol{B}_P \boldsymbol{\Theta}_l^T \left.\frac{\partial \boldsymbol{\varphi}(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P = \boldsymbol{x}_{P,l}} \tag{4-9}$$

Similarly, the partial derivative of (4-6) with respect to the adaptive parameters $\boldsymbol{\Theta}$ is

$$\left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{x}_P = \boldsymbol{x}_{P,l}} = \boldsymbol{B}_P \boldsymbol{\varphi}^T(\boldsymbol{x}_{P,l}) \tag{4-10}$$

Note, that the plant dynamics (3-13) are independent of the employed adaptive control technique. Hence the results above hold for both considered cases. Section 4.3.1.1 linearizes the update laws in the direct MRAC case, while section 4.3.1.2 construct a linear surrogate model for PMRAC.

4.3.1.1 Linearization in the Direct MRAC case

The right hand side of the parameter update law in the direct MRAC case (3-19) is

$$\boldsymbol{f}_\Theta = -\boldsymbol{\Gamma} \boldsymbol{\varphi}(\boldsymbol{x}_P) \boldsymbol{e}^T \boldsymbol{P} \boldsymbol{B}_P - \boldsymbol{\Gamma} k(\boldsymbol{\Theta}) \tag{4-11}$$

A Taylor series expansion at $\boldsymbol{x}_P = \boldsymbol{x}_{P,l}$ and $\boldsymbol{\Theta} = \boldsymbol{\Theta}_l$ results in

$$\boldsymbol{f}_\Theta = \boldsymbol{f}_\Theta(\boldsymbol{x}_{P,l}, \boldsymbol{\Theta}_l) + \left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P = \boldsymbol{x}_{P,l}} (\boldsymbol{x}_P - \boldsymbol{x}_{P,l}) + \left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{\Theta} = \boldsymbol{\Theta}_l} (\boldsymbol{\Theta} - \boldsymbol{\Theta}_l) + O^2(\boldsymbol{x}_P, \boldsymbol{\Theta}). \tag{4-12}$$

Here, $\boldsymbol{f}_\Theta(\boldsymbol{x}_{P,l}, \boldsymbol{\Theta}_l)$ is

$$\boldsymbol{f}_\Theta(\boldsymbol{x}_{P,l}, \boldsymbol{\Theta}_l) = -\boldsymbol{\Gamma} \boldsymbol{\varphi}(\boldsymbol{x}_{P,l})(\boldsymbol{x}_{P,l} - \boldsymbol{x}_{rm})^T \boldsymbol{P} \boldsymbol{B}_P - \boldsymbol{\Gamma} k(\boldsymbol{\Theta}_l) \tag{4-13}$$

The partial derivative of (4-12) with respect to the system states $\boldsymbol{x}_P$ is

$$\begin{gathered}\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P = \boldsymbol{x}_{P,l}} = -\boldsymbol{\Gamma} \left.\frac{\partial \boldsymbol{\varphi}(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P = \boldsymbol{x}_{P,l}} \boldsymbol{x}_{P,l}^T \boldsymbol{P} \boldsymbol{B}_P - \boldsymbol{\Gamma} \boldsymbol{\varphi}(\boldsymbol{x}_{P,l}) \boldsymbol{B}_P^T \boldsymbol{P} \\ + \boldsymbol{\Gamma} \left.\frac{\partial \boldsymbol{\varphi}(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P = \boldsymbol{x}_{P,l}} \boldsymbol{x}_{rm}^T \boldsymbol{P} \boldsymbol{B}_P\end{gathered} \tag{4-14}$$

Similarly, the partial derivative of (4-12) with respect to the adaptive parameters $\boldsymbol{\Theta}$ is

$$\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{\Theta} = \boldsymbol{\Theta}_l} = -\boldsymbol{\Gamma} \left.\frac{\partial k(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{\Theta} = \boldsymbol{\Theta}_l} \tag{4-15}$$

Note, that some update law modifications $k(\boldsymbol{\Theta})$ additionally depend on the system state $\boldsymbol{x}_P$. One such example is Narendras e –Modification ([154]). In this case, (4-14) needs to be expanded to include the partial derivative of $k(\boldsymbol{\Theta})$ with respect to $\boldsymbol{x}_P$. By using (4-8)-(4-10) as well as (4-13)-(4-15), the surrogate closed-loop system model for the direct MRAC case becomes

$$\begin{aligned}\begin{bmatrix}\dot{\boldsymbol{x}}_P\\ \mathrm{vec}(\boldsymbol{\Theta})\end{bmatrix} = \begin{bmatrix}\boldsymbol{f}_x(\boldsymbol{x}_{P,l},\boldsymbol{\Theta}_l)\\ \boldsymbol{f}_\Theta(\boldsymbol{x}_{P,l},\boldsymbol{\Theta}_l)\end{bmatrix} + \begin{bmatrix} \left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} & \left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} \\ \mathrm{M}\left(\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}}\right) & \mathrm{M}\left(\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}_l}\right)\end{bmatrix} \begin{bmatrix}\boldsymbol{x}_P-\boldsymbol{x}_{P,l}\\ \mathrm{vec}(\boldsymbol{\Theta}-\boldsymbol{\Theta}_l)\end{bmatrix} \\ +O^2(\boldsymbol{x}_P,\boldsymbol{\Theta}).\end{aligned} \tag{4-16}$$

Here, $\mathrm{vec}(\boldsymbol{M})$ denotes the operator that stacks the columns of the matrix $\boldsymbol{M}$ into a vector. Furthermore, $\mathrm{M}(\boldsymbol{M})$ denotes the operator that reshapes the matrix $\boldsymbol{M}$ such that the calculation result of (4-16) is preserved. By neglecting the higher order terms, the surrogate model becomes linear:

$$\begin{aligned}\dot{\boldsymbol{x}}_s = \begin{bmatrix}\dot{\boldsymbol{x}}_P\\ \mathrm{vec}(\boldsymbol{\Theta})\end{bmatrix} = \underbrace{\begin{bmatrix}\boldsymbol{f}_x(\boldsymbol{x}_{P,l},\boldsymbol{\Theta}_l)\\ \boldsymbol{f}_\Theta(\boldsymbol{x}_{P,l},\boldsymbol{\Theta}_l)\end{bmatrix}}_{\boldsymbol{f}(\boldsymbol{x}_{s,l})} + \underbrace{\begin{bmatrix} \left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} & \left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} \\ \mathrm{M}\left(\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}}\right) & \mathrm{M}\left(\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}_l}\right)\end{bmatrix}}_{\boldsymbol{A}_{lin}(\boldsymbol{x}_{s,l})} \underbrace{\begin{bmatrix}\boldsymbol{x}_P-\boldsymbol{x}_{P,l}\\ \mathrm{vec}(\boldsymbol{\Theta}-\boldsymbol{\Theta}_l)\end{bmatrix}}_{\boldsymbol{x}_s-\boldsymbol{x}_{s,l}} \\ = \boldsymbol{A}_{lin}(\boldsymbol{x}_{s,l})(\boldsymbol{x}_s-\boldsymbol{x}_{s,l}) + \boldsymbol{f}(\boldsymbol{x}_{s,l})\end{aligned} \tag{4-17}$$

4.3.1.2 Linearization of the PMRAC case

In the PMRAC case, not only the parameter update dynamics (3-38) but also the predictor dynamics (3-33) need to be considered. The right hand side of the parameter update law (3-38) is

$$\begin{aligned}\boldsymbol{f}_\Theta &= -\boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P)(\boldsymbol{e}^T\boldsymbol{P} - \boldsymbol{e}_P^T\boldsymbol{P}_P)\boldsymbol{B}_P - \boldsymbol{\Gamma}k(\boldsymbol{\Theta}) \\ &= -\boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P)\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P + \boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P)\boldsymbol{e}_P^T\boldsymbol{P}_P\boldsymbol{B}_P - \boldsymbol{\Gamma}k(\boldsymbol{\Theta})\end{aligned} \tag{4-18}$$

Expanding the tracking and prediction error further yields

$$\begin{aligned}\boldsymbol{f}_\Theta = -\boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P)\boldsymbol{x}_P^T\boldsymbol{P}\boldsymbol{B}_P + \boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P)\boldsymbol{x}_{rm}^T\boldsymbol{P}\boldsymbol{B}_P + \boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P)\hat{\boldsymbol{x}}_P^T\boldsymbol{P}_P\boldsymbol{B}_P \\ -\boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P)\boldsymbol{x}_P^T\boldsymbol{P}_P\boldsymbol{B}_P - \boldsymbol{\Gamma}k(\boldsymbol{\Theta}).\end{aligned} \tag{4-19}$$

Note, that (4-19) depends on the plant states $\boldsymbol{x}_P$, the adaptive parameters $\boldsymbol{\Theta}$ as well as on the predictor states $\hat{\boldsymbol{x}}_P$. Hence, a Taylor series expansion at $\boldsymbol{x}_P = \boldsymbol{x}_{P,l}$, $\boldsymbol{\Theta} = \boldsymbol{\Theta}_l$ and $\hat{\boldsymbol{x}}_P = \hat{\boldsymbol{x}}_{P,l}$ results in

$$\begin{aligned}\boldsymbol{f}_\Theta = \boldsymbol{f}_\Theta(\boldsymbol{x}_{P,l},\boldsymbol{\Theta}_l,\hat{\boldsymbol{x}}_{P,l}) + \left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \hat{\boldsymbol{x}}_P=\hat{\boldsymbol{x}}_{P,l}}}(\boldsymbol{x}_P-\boldsymbol{x}_{P,l}) + \left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}_l}(\boldsymbol{\Theta}-\boldsymbol{\Theta}_l) \\ \left.\frac{\partial \boldsymbol{f}_x}{\partial \hat{\boldsymbol{x}}_P}\right|_{\hat{\boldsymbol{x}}_P=\hat{\boldsymbol{x}}_{P,l}}(\hat{\boldsymbol{x}}_P-\hat{\boldsymbol{x}}_{P,l}) + O^2(\boldsymbol{x}_P,\boldsymbol{\Theta}).\end{aligned} \tag{4-20}$$

Here, $\boldsymbol{f}_{\Theta}(\boldsymbol{x}_{P,l},\boldsymbol{\Theta}_l,\hat{\boldsymbol{x}}_{P,l})$ is

$$\begin{aligned}\boldsymbol{f}_{\Theta}(\boldsymbol{x}_{P,l},\boldsymbol{\Theta}_l,\hat{\boldsymbol{x}}_{P,l}) = -\boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_{P,l})\boldsymbol{x}_{P,l}^T\boldsymbol{P}\boldsymbol{B}_P + \boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_{P,l})\boldsymbol{x}_{rm}^T\boldsymbol{P}\boldsymbol{B}_P + \boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_{P,l})\hat{\boldsymbol{x}}_{P,l}^T\boldsymbol{P}_P\boldsymbol{B}_P \\ -\boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_{P,l})\boldsymbol{x}_{P,l}^T\boldsymbol{P}_P\boldsymbol{B}_P - \boldsymbol{\Gamma}k(\boldsymbol{\Theta}_l)\end{aligned} \tag{4-21}$$

The partial derivative of (4-20) with respect to the system states $\boldsymbol{x}_P$ is

$$\begin{aligned}\left.\frac{\partial \boldsymbol{f}_{\Theta}}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \hat{\boldsymbol{x}}_P=\hat{\boldsymbol{x}}_{P,l}}} = -\boldsymbol{\Gamma}\left.\frac{\partial \boldsymbol{\varphi}(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} \boldsymbol{x}_{P,0}^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_{P,l})\boldsymbol{B}_P^T\boldsymbol{P} + \boldsymbol{\Gamma}\left.\frac{\partial \boldsymbol{\varphi}(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} \boldsymbol{x}_{rm}^T\boldsymbol{P}\boldsymbol{B}_P \\ +\boldsymbol{\Gamma}\left.\frac{\partial \boldsymbol{\varphi}(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} \hat{\boldsymbol{x}}_{P,l}^T\boldsymbol{P}_P\boldsymbol{B}_P - \boldsymbol{\Gamma}\left.\frac{\partial \boldsymbol{\varphi}(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} \boldsymbol{x}_{P,l}^T\boldsymbol{P}_P\boldsymbol{B}_P - \boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_{P,l})\boldsymbol{B}_P^T\boldsymbol{P}_P.\end{aligned} \tag{4-22}$$

The partial derivative of (4-20) with respect to the adaptive parameters $\boldsymbol{\Theta}$ is

$$\left.\frac{\partial \boldsymbol{f}_{\Theta}}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}_l} = -\boldsymbol{\Gamma}\left.\frac{\partial k(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}_l}. \tag{4-23}$$

Finally, the partial derivative of (4-20) with respect to the predictor states $\hat{\boldsymbol{x}}_P$ is

$$\left.\frac{\partial \boldsymbol{f}_{\Theta}}{\partial \hat{\boldsymbol{x}}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} = \boldsymbol{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_{P,l})\boldsymbol{B}_P^T\boldsymbol{P}_P \tag{4-24}$$

The right hand side of the predictor dynamics (3-33) is

$$\boldsymbol{f}_{\hat{x}} = \boldsymbol{A}_{prd}(\hat{\boldsymbol{x}}_P - \boldsymbol{x}_P) + \boldsymbol{A}_{rm}\boldsymbol{x}_P + \boldsymbol{B}_{rm}\boldsymbol{r} \tag{4-25}$$

Note that the predictor dynamics are already in a linear form. For consistency, the Taylor series expansion is applied nevertheless:

$$\boldsymbol{f}_{\hat{x}} = \boldsymbol{f}_{\hat{x}}(\boldsymbol{x}_{P,l},\hat{\boldsymbol{x}}_{P,l}) + \frac{\partial \boldsymbol{f}_{\hat{x}}}{\partial \boldsymbol{x}_P}(\boldsymbol{x}_P - \boldsymbol{x}_{P,l}) + \frac{\partial \boldsymbol{f}_{\hat{x}}}{\partial \hat{\boldsymbol{x}}_P}(\hat{\boldsymbol{x}}_P - \hat{\boldsymbol{x}}_{P,l}) \tag{4-26}$$

Note that $\boldsymbol{f}_{\hat{x}}$ only depends on the plant and predictor states. Furthermore, note that no higher order terms are present, since the predictor dynamics are already linear. Consequently, $\boldsymbol{f}_{\hat{x}}(\boldsymbol{x}_{P,l},\hat{\boldsymbol{x}}_{P,l})$ is

$$\boldsymbol{f}_{\hat{x}}(\boldsymbol{x}_{P,l},\hat{\boldsymbol{x}}_{P,l}) = \boldsymbol{A}_{prd}(\hat{\boldsymbol{x}}_{P,l} - \boldsymbol{x}_{P,l}) + \boldsymbol{A}_{rm}\boldsymbol{x}_{P,l} + \boldsymbol{B}_{rm}\boldsymbol{r} \tag{4-27}$$

The partial derivative of (4-25) with respect to the system states $\boldsymbol{x}_P$ is

$$\frac{\partial \boldsymbol{f}_{\hat{x}}}{\partial \boldsymbol{x}_P} = \boldsymbol{A}_{rm} - \boldsymbol{A}_{prd}. \tag{4-28}$$

Similarly, the partial derivative of (4-25) with respect to the predictor states $\hat{\boldsymbol{x}}_P$ is

$$\frac{\partial \boldsymbol{f}_{\hat{x}}}{\partial \hat{\boldsymbol{x}}_P} = \boldsymbol{A}_{prd}. \tag{4-29}$$

By using (4-8)-(4-10) as well as (4-21)-(4-24) and (4-27)-(4-29), the surrogate closed-loop system model for the P MRAC case becomes

$$\begin{bmatrix} \dot{\boldsymbol{x}}_P \\ \mathrm{vec}(\dot{\boldsymbol{\Theta}}) \\ \dot{\hat{\boldsymbol{x}}}_P \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{f}_x(\boldsymbol{x}_{P,l}, \boldsymbol{\Theta}_l, \hat{\boldsymbol{x}}_{P,l}) \\ \boldsymbol{f}_\Theta(\boldsymbol{x}_{P,l}, \boldsymbol{\Theta}_l, \hat{\boldsymbol{x}}_{P,l}) \\ \boldsymbol{f}_{\hat{x}}(\boldsymbol{x}_{P,l}, \boldsymbol{\Theta}_l, \hat{\boldsymbol{x}}_{P,l}) \end{bmatrix}}_{\boldsymbol{f}(\boldsymbol{x}_{s,l})} +$$

$$+ \underbrace{\begin{bmatrix} \left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l} \\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} & \left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} & \boldsymbol{0} \\ \mathrm{M}\left(\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l} \\ \hat{\boldsymbol{x}}_P=\hat{\boldsymbol{x}}_{P,l}}}\right) & \mathrm{M}\left(\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}_l}\right) & \mathrm{M}\left(\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \hat{\boldsymbol{x}}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}}\right) \\ \frac{\partial \boldsymbol{f}_{\hat{x}}}{\partial \boldsymbol{x}_P} & \boldsymbol{0} & \frac{\partial \boldsymbol{f}_{\hat{x}}}{\partial \hat{\boldsymbol{x}}_P} \end{bmatrix}}_{\boldsymbol{A}_{lin}(\boldsymbol{x}_{s,l})} \underbrace{\begin{bmatrix} \boldsymbol{x}_P - \boldsymbol{x}_{P,l} \\ \mathrm{vec}(\boldsymbol{\Theta} - \boldsymbol{\Theta}_l) \\ \hat{\boldsymbol{x}}_P - \hat{\boldsymbol{x}}_{P,l} \end{bmatrix}}_{\boldsymbol{x}_s - \boldsymbol{x}_{s,l}} + O^2(\boldsymbol{x}_P) \tag{4-30}$$

By neglecting the higher order terms, the surrogate model becomes linear such that it is representable similar to (4-17):

$$\dot{\boldsymbol{x}}_s = \boldsymbol{A}_{lin}(\boldsymbol{x}_{s,l})(\boldsymbol{x}_s - \boldsymbol{x}_{s,l}) + \boldsymbol{f}(\boldsymbol{x}_{s,l}). \tag{4-31}$$

4.3.2 Propagating a GMM through the linearized system dynamics

This section details the propagation of a rv with GMM distribution through the nonlinear system (3-13). The general derivation can be found i.a. in [206]. Consider the approximation of $p(\boldsymbol{x}_s)$ by a GMM in (4-5). While a continuous derivation is possible, deriving the GMM propagation for a discrete system directly gives rise to the equations used for implementation. Therefore, consider the discretized version of the linearized system dynamics (4-17) or (4-31), respectively:

$$\overline{\boldsymbol{x}}_{s,k+1} = \overline{\boldsymbol{A}}_{lin}(\overline{\boldsymbol{x}}_{s,k,l})(\overline{\boldsymbol{x}}_{s,k} - \overline{\boldsymbol{x}}_{s,k,l}) + \overline{\boldsymbol{f}}(\overline{\boldsymbol{x}}_{s,k,l}). \tag{4-32}$$

Discretized quantities are depicted by a bar. Predicting $p(\overline{\boldsymbol{x}}_{s,k+1})$ comes down to evaluating the Chapman-Kolmogorov equation in (4-2). However, instead of employing the real $p(\overline{\boldsymbol{x}}_{s,k+1})$, its approximation (4-5) is used. Furthermore, instead of using the nonlinear system dynamics, their linear approximation $\hat{p}(\overline{\boldsymbol{x}}_{s,k+1}|\overline{\boldsymbol{x}}_{s,k})$ in (4-32) is used. Therefore, also the result $\hat{p}(\overline{\boldsymbol{x}}_{s,k+1})$ represents an approximation of $p(\overline{\boldsymbol{x}}_{s,k+1})$:

$$\hat{p}(\overline{\boldsymbol{x}}_{s,k+1}) = \int \hat{p}(\overline{\boldsymbol{x}}_{s,k+1}|\overline{\boldsymbol{x}}_{s,k})\hat{p}(\overline{\boldsymbol{x}}_{s,k})d\overline{\boldsymbol{x}}_{s,k}. \tag{4-33}$$

Instead of solving (4-33) directly, note that propagating a rv with a GMM distribution through the linear system (4-32) amounts to an affine transformation of $\overline{\boldsymbol{x}}_{s,k}$. Hence, Theorem 2.2 can be applied to directly calculate the resulting distribution $\hat{p}(\overline{\boldsymbol{x}}_{s,k+1})$. Rewriting (4-32) yields

$$\overline{\boldsymbol{x}}_{s,k+1} = \overline{\boldsymbol{f}}(\overline{\boldsymbol{x}}_{s,k,l}) - \overline{\boldsymbol{A}}_{lin}(\overline{\boldsymbol{x}}_{s,k,l})\overline{\boldsymbol{x}}_{s,k,l} + \overline{\boldsymbol{A}}_{lin}(\overline{\boldsymbol{x}}_{s,k,l})\overline{\boldsymbol{x}}_{s,k}. \tag{4-34}$$

Applying Theorem 2.2 results in

$$\hat{p}\left(\bar{\boldsymbol{x}}_{s,k+1}\right) = \sum_{i=1}^{m} \alpha_i N \begin{pmatrix} \bar{\boldsymbol{A}}_{lin}\left(\bar{\boldsymbol{x}}_{s,k,l_i}\right)\boldsymbol{\mu}_{k,i} + \bar{\boldsymbol{f}}\left(\bar{\boldsymbol{x}}_{s,k,l_i}\right) - \bar{\boldsymbol{A}}_{lin}\left(\bar{\boldsymbol{x}}_{s,k,l_i}\right)\bar{\boldsymbol{x}}_{s,k,l_i}, \\ \bar{\boldsymbol{A}}_{lin}\left(\bar{\boldsymbol{x}}_{s,k,l_i}\right)\boldsymbol{\Sigma}_{k,i}\bar{\boldsymbol{A}}_{lin}^{T}\left(\bar{\boldsymbol{x}}_{s,k,l_i}\right) \end{pmatrix} \tag{4-35}$$

Here $\bar{\boldsymbol{x}}_{s,k,l_i}$ denotes the point respective to which the linearization is performed and corresponding to the i –th mixture component. Furthermore, $\boldsymbol{\mu}_{k,i}$ and $\boldsymbol{\Sigma}_{k,i}$ represent the mean and covariance of the i –th mixture component at the k –th timestep. Similarly to the EKF, the reference point is chosen to be the mean of each mixture component, such that $\bar{\boldsymbol{x}}_{s,k,l_i} = \boldsymbol{\mu}_{k,i}$. Hence, equation (4-35) becomes

$$\hat{p}\left(\boldsymbol{x}_{s,k+1}\right) = \sum_{i=1}^{m} \alpha_i N \left(\bar{\boldsymbol{f}}\left(\boldsymbol{\mu}_{k,i}\right), \bar{\boldsymbol{A}}_{lin}\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{k,i}\bar{\boldsymbol{A}}_{lin}^{T}\left(\boldsymbol{\mu}_{k,i}\right)\right) \tag{4-36}$$

Equation (4-36) contains the update equations for the mean and covariance. In particular, the update for the mean of the i –th mixture component is

$$\boldsymbol{\mu}_{k+1,i} = \bar{\boldsymbol{f}}\left(\boldsymbol{\mu}_{k,i}\right). \tag{4-37}$$

Similarly, the update law for the covariance of the i –th mixture component is

$$\boldsymbol{\Sigma}_{k+1,i} = \bar{\boldsymbol{A}}_{lin}\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{k,i}\bar{\boldsymbol{A}}_{lin}^{T}\left(\boldsymbol{\mu}_{k,i}\right). \tag{4-38}$$

Note, that (4-37) and (4-38) resemble the EKF update equations. In [206], Terejanu et al. also derive the continuous version of (4-37) and (4-38). Recursively applying (4-37) and (4-38) results in the predicted trajectory for the system states $\boldsymbol{x}_s$.

Note that only the mean and covariance are updated, the mixture weights α_i stay constant. This assumes that, throughout the prediction horizon, the covariance $\boldsymbol{\Sigma}_i$ of the i –th mixture component stays small enough in order for the linearization to be valid with respect to $\boldsymbol{\mu}_i$. In particular, for highly nonlinear functions this assumption is easily violated (see Example 4-4 for a comparison of functions). Terejanu et al. provide a way to also update the mixture weights themselves. Therefore, a multiparametric quadratic program with linear equality and inequality constraints has to be solved, which aims at minimizing the error between the true forecast distribution $p\left(\boldsymbol{x}_{s,k+1}\right)$ and its approximation $\hat{p}\left(\boldsymbol{x}_{s,k+1}\right)$. However, solving an optimization problem online demands high computational resources, which may not be available. Also changing the number of mixture components requires the solution of an optimization problem online. Again, the required computational resources might not be available. The presented monitoring algorithm instead follows a different path. Section 4.5.2 deals with errors occurring due to approximations during the prediction. Using the latter, a measure of confidence is derived, which is intended to relay information about the certainty the algorithm has in its own prediction capabilities. In the following, two examples show the capabilities of using GMMs for state propagation.

Example 4-3: **Propagation of a normally distributed random variable through a sine function by sampling and linear as well as GMM approximation**

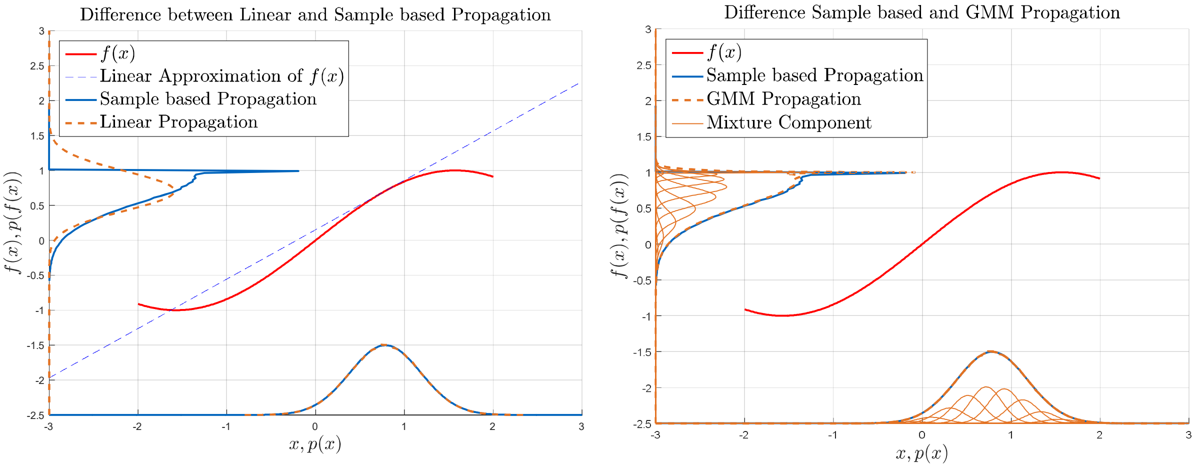


Figure 4-6: Comparison between linear, sampling based and GMM propagation

Consider the problem statement of Example 4-2. Instead of approximating the nonlinear function $f(x) = sin(x)$ by a single linear function only, the GMM propagation approach is employed. Therefore, the initial distribution $p(x) = N(0.25\pi, 0.4^2)$ is approximated by a GMM. The latter is chosen to consist of 40 mixture components with a standard deviation of $\sigma_i = 0.175$, which approximate $p(x)$ for $x \in [-4,4]$. The fitting is performed by solving (2-5). The update laws (4-37) and (4-38) are used in order to propagate the GMM through $f(x)$.

Figure 4-6 compares the propagation by linear and GMM propagation. In that, Figure 4-6 (left) shows the results of Example 4-2 for a single linear approximation. Figure 4-6 (right) shows the propagation by GMM approximation. It can be seen that the state distribution $\hat{p}(f(x))$ is nearly indistinguishable from the sample based propagation. In fact, the D_{KL} drops from 26.58 for Figure 4-6 (left) to 5.58 for Figure 4-6 (right). Primarily 8 mixture components contribute to the approximation of $\hat{p}(x)$ and $\hat{p}(f(x))$. This indicates, that already 8 mixtures are sufficient to represent $\hat{p}(f(x))$ well. This also indicates that the number of mixtures, as well as the region in which $\hat{p}(x)$ is approximated, could have been chosen smaller.

The occurring errors can be attributed to two reasons. First, even though the standard deviation of each mixture component is small, the linearization is not necessarily valid in a huge domain. This is in particular true for the maximum of the sign function, which is approximated by a horizontal line. Decreasing σ_i further and increasing the number of mixtures generally leads to a decrease of the error. However, this also increases the computational burden on the system. Secondly, each Gaussian mixture is inherently continuous. Hence, even if the mapping through the sine function is bounded by 1 and -1, respectively, the resulting GMM is not able to represent discontinuous functions.

Example 4-4: Propagation of a normally distributed random variable through a variety of nonlinear functions with varying parameters of the prior distribution

This example repeats the experiments in Example 4-2 and Example 4-3 for the nonlinear functions as well as varying parameters of the prior distribution. In particular, the nonlinear functions are $f_1(x) = 0.5x^2$, $f_2(x) = 0.02x^3, f_3(x) = \sin(2x)$ and $f_4(x) = \tan^{-1}(x)$. The mean of the prior distribution varies from $\mu \in [-1.5,1.5]$ and the standard deviation varies from $\sigma \in [0.05,1.8]$. Again the Kullback-Leibler divergence is used to relatively compare test cases. The left column of Figure 4-7 shows the linear propagation and the right column shows the GMM propagation, respectively. In order to be able to compare results, for one function the same Kullback-Leibler divergence scale is used. For more detail, an adjusted scale is depicted in Figure H- in Appendix H.2.

The first two rows of Figure 4-7 show the Kullback-Leibler divergence for the second and third order polynomial, respectively. It can be seen that a shift of the mean away from zero and an increase of the standard deviation lead to a decrease in D_{KL}. However, for the linear propagation the numerical value of D_{KL} still is considerably higher than in Example 4-2 with $D_{KL} = 26.58$. This indicates that despite the decrease in D_{KL}, the predictive distribution is not approximated well by a normal distribution. In particular, around $\mu = 0$, the D_{KL} assumes relatively large values. This indicates that the nonlinear function is badly approximated by a linear function. In fact, for $\mu = 0$, the linear approximation of $f_1(x)$ and $f_2(x)$ is a horizontal line. The prior distribution is therefore mapped upon a single deterministic value. By using the GMM propagation, D_{KL} is drastically reduced. Still, around $\mu = 0$, the D_{KL} remains large. Similar to Example 4-3, this is due to the facts that the linearization is not valid over each mixture component and that a GMM is not able to represent discontinuous functions.

The third row of Figure 4-7 shows the Kullback-Leibler divergence for the sine function as an extension to Example 4-2. The D_{KL} assumes its biggest values in areas close to its extrema. Again, the reason for this is that the linear approximation at the extrema of the nonlinear function assumes the shape of a horizontal line and is therefore only valid in a small area. Similar to the polynomial functions, the GMM propagation drastically reduces the D_{KL}.

Finally, row four of Figure 4-7 shows the Kullback-Leibler divergence for the arctangent function. It can be seen that a prior mean close to zero achieves low D_{KL} values, when compared to the other nonlinear functions. This is due to the fact that $\tan^{-1}(x)$ can be well approximated by a first order Taylor series around a zero mean. Still, the GMM propagation is able to reduce the D_{KL} further. Note that while $\tan^{-1}(x)$ does have suprema, it does not have extrema. Hence, within the analyzed region, the predictive will be continuous and representable by a GMM. Due to the small numerical values, the scale has been adjusted for the GMM propagation case.

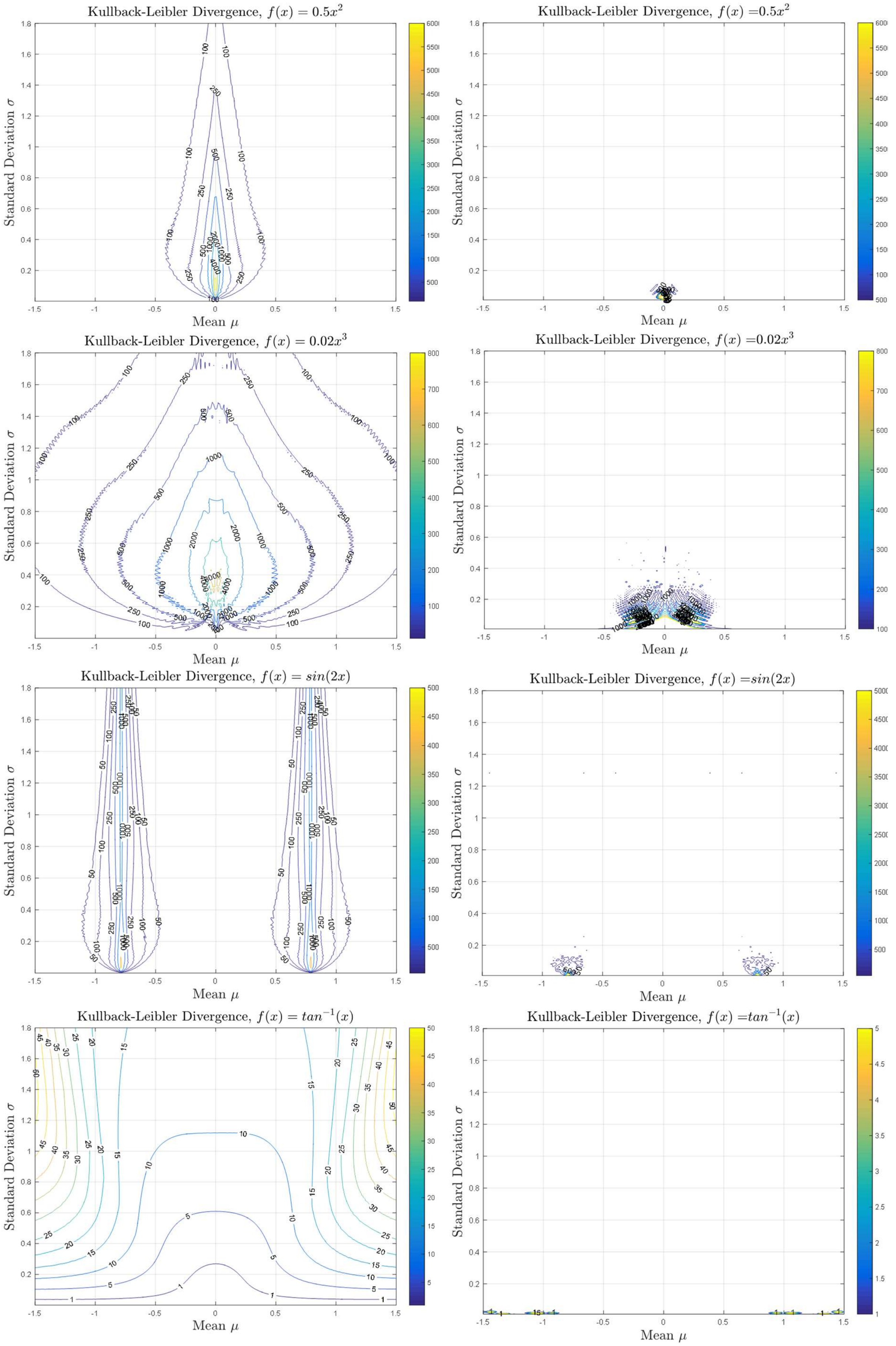


Figure 4-7: Comparison of linear (left) and GMM (right) propagation based on the Kullback-Leibler Divergence for varying parameters of the prior and different nonlinear functions

4.4 Online Uncertainty Propagation using Gaussian Processes

Chapter 4.3 assumed the plant dynamics to be known entirely. This section extends the approach to include modelling uncertainties $\boldsymbol{\Delta}(\boldsymbol{x}_{\mathrm{P}})$. Therefore, consider the linearized system dynamics of (4-17) or (4-31) subject to a modelling uncertainty $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$:

$$\bar{\boldsymbol{x}}_{s,k+1} = \bar{\boldsymbol{A}}_{lin}(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{B}}_{P}\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}). \tag{4-39}$$

GPs are used to model $\boldsymbol{\Delta}(\boldsymbol{x}_{\mathrm{P}})$. Thereby, this thesis considers two cases. In the first case, the structure of $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ is known and only parameter uncertainties are present. Hence, $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ can be modelled using a GP prior model. In the case that $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ is entirely unknown, GP regression infers a model for $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ based on online gathered data.

In contrast to section 2.2, this chapter admits the uncertainty $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) \in \mathbb{R}^{l\times 1}$ to be a vector. In general, $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ is represented by $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) = \boldsymbol{w}^T\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k})$. By placing a probability distribution on the parameters $\boldsymbol{w} \in \mathbb{R}^{k\times l}$, the uncertainty $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ forms a GP as of (2-26). The optimal parameters of the uncertainty $\boldsymbol{\Theta}^*$ as of Definitions 3.1 are deliberately substituted by $\boldsymbol{w}$. This leads to a clear distinction between the control objective, for which $\boldsymbol{\Theta}^*$ is a draw from $\boldsymbol{w}$, and the monitoring objective, for which the whole distribution of $\boldsymbol{w}$ is considered. In any case, let the parameters $\boldsymbol{w}$ be distributed according to

$$\mathrm{vec}(\boldsymbol{w}) \sim N(\mathrm{vec}(\boldsymbol{\mu}_w), \boldsymbol{\Sigma}_w). \tag{4-40}$$

First, since $\boldsymbol{w}$ is a matrix, it is convenient to reformulate $\boldsymbol{w}$ as a vector. Furthermore, the mean of $\boldsymbol{w}$ is represented by $\boldsymbol{\mu}_w$ instead of $\bar{\boldsymbol{w}}$, which in turn is the prevalent notation in the machine learning literature. Since a bar indicates a discretized value in this chapter, the mean of $\boldsymbol{w}$ could be easily confused otherwise. Finally, $\boldsymbol{\Sigma}_w \in \mathbb{R}^{kl\times kl}$ represents the covariance matrix. As before, assume the weights to be independent of each other. Hence, $\boldsymbol{\Sigma}_w$ is a diagonal matrix. Dividing $\boldsymbol{\Sigma}_w$ into l^2 submatrices $\boldsymbol{\Sigma}_{w,i,j} \in \mathbb{R}^{k\times k}$ yields

$$\boldsymbol{\Sigma}_w = \begin{bmatrix} \boldsymbol{\Sigma}_{w,1,1} & \cdots & \boldsymbol{\Sigma}_{w,1,l} \\ \vdots & \ddots & \vdots \\ \boldsymbol{\Sigma}_{w,l,1} & \cdots & \boldsymbol{\Sigma}_{w,l,l} \end{bmatrix}. \tag{4-41}$$

Since $\boldsymbol{\Sigma}_w$ is a diagonal matrix, $\boldsymbol{\Sigma}_{w,i,j} = \boldsymbol{0}, i \neq j$. The uncertainty is modelled by

$$\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) \sim GP\left(\boldsymbol{m}_{\Delta}(\bar{\boldsymbol{x}}_{s,k}), \boldsymbol{\Sigma}_{\Delta}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})\right). \tag{4-42}$$

To calculate the mean $\boldsymbol{m}_{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ and covariance function $\boldsymbol{\Sigma}_{\Delta}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})$ consider the columns of the parameter matrix $\boldsymbol{w} = [\boldsymbol{w}_1 \quad \cdots \quad \boldsymbol{w}_l]$. Following (2-22), the mean of the i-th element in the uncertainty vector $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ is

$$m_{\Delta,i}(\bar{\boldsymbol{x}}_{s,k}) = \mathbb{E}_{\Delta}[\Delta_i(\bar{\boldsymbol{x}}_{s,k})] = \mathbb{E}_{w}[\boldsymbol{w}_i^T\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k})] = \boldsymbol{\mu}_{w,i}^T\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}). \tag{4-43}$$

Combining all elements, the mean function $\boldsymbol{m}_\Delta(\bar{\boldsymbol{x}}_{s,k})$ is

$$\boldsymbol{m}_\Delta(\bar{\boldsymbol{x}}_{s,k}) = \begin{bmatrix} m_{\Delta,1}(\bar{\boldsymbol{x}}_{s,k}) \\ \vdots \\ m_{\Delta,l}(\bar{\boldsymbol{x}}_{s,k}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_{w,1}^T \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}) \\ \vdots \\ \boldsymbol{\mu}_{w,l}^T \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_{w,1}^T \\ \vdots \\ \boldsymbol{\mu}_{w,l}^T \end{bmatrix} \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}) = \boldsymbol{\mu}_w^T \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}). \tag{4-44}$$

In order to derive the covariance function $\boldsymbol{\Sigma}_\Delta(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})$, consider its l^2 entries $\Sigma_{\Delta,i,j}$:

$$\boldsymbol{\Sigma}_\Delta(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k}) = \begin{bmatrix} \Sigma_{\Delta,1,1} & \cdots & \Sigma_{\Delta,1,l} \\ \vdots & \ddots & \vdots \\ \Sigma_{\Delta,l,1} & \cdots & \Sigma_{\Delta,l,l} \end{bmatrix}. \tag{4-45}$$

Following (2-23), the i –th and j –th entry is

$$\begin{aligned} \Sigma_{\Delta,i,j} &= \mathbb{E}_\Delta\left[\left(\Delta_i(\bar{\boldsymbol{x}}_{s,k}) - \mathbb{E}_\Delta[\Delta_i(\bar{\boldsymbol{x}}_{s,k})]\right)^T \left(\Delta_j(\bar{\boldsymbol{x}}_{s,k}) - \mathbb{E}_\Delta[\Delta_j(\bar{\boldsymbol{x}}_{s,k})]\right)\right] \\ &= \mathbb{E}_\Delta\left[\left(\boldsymbol{w}_i^T \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}) - \boldsymbol{\mu}_{w,i}^T \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k})\right)^T \left(\boldsymbol{w}_j^T \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}) - \boldsymbol{\mu}_{w,j}^T \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k})\right)\right] \\ &= \boldsymbol{\phi}^T(\bar{\boldsymbol{x}}_{s,k}) \underbrace{\mathbb{E}_\Delta\left[\left(\boldsymbol{w}_i^T - \boldsymbol{\mu}_{w,i}^T\right)^T \left(\boldsymbol{w}_j^T - \boldsymbol{\mu}_{w,j}^T\right)\right]}_{\boldsymbol{\Sigma}_{w,i,j}} \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}) = \boldsymbol{\phi}^T(\bar{\boldsymbol{x}}_{s,k}) \boldsymbol{\Sigma}_{w,i,j} \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}) \end{aligned} \tag{4-46}$$

Here $\boldsymbol{\Sigma}_{w,i,j}$ corresponds to the ith and jth submatrix of $\boldsymbol{\Sigma}_w$. Since $\boldsymbol{\Sigma}_{w,i,j} = \boldsymbol{0}, i \neq j$, also $\boldsymbol{\Sigma}_{\Delta,i,j} = \boldsymbol{0}, i \neq j$. Hence, the covariance function becomes

$$\boldsymbol{\Sigma}_\Delta(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k}) = \begin{bmatrix} \boldsymbol{\phi}^T(\bar{\boldsymbol{x}}_{s,k}) \boldsymbol{\Sigma}_{w,1,1} \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}) & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \boldsymbol{\phi}^T(\bar{\boldsymbol{x}}_{s,k}) \boldsymbol{\Sigma}_{w,l,l} \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}) \end{bmatrix}. \tag{4-47}$$

The following sections show how to employ the GPs, which model the uncertainty $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$, in the prediction approach. The outline of this section is as follows. Chapter 4.4.1 shows how to propagate a GP if the regressor function is linear, i.e. the GP is modelled by linear kernel functions. Chapter 4.4.2 extends the result for nonlinear regressor functions. Finally, section 4.4.3 shows how to use GP regression to infer a model of $\boldsymbol{\Delta}(\boldsymbol{x}_\mathrm{P})$ from online gathered data and propagate the result.

4.4.1 Linear Gaussian Process models

This section represents the modelling uncertainty $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ by a linear combination of parameters $\boldsymbol{w}$ with the system state vector $\bar{\boldsymbol{x}}_{s,k}$ such that $\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}) = \bar{\boldsymbol{x}}_{s,k}$ and $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) = \boldsymbol{w}^T \bar{\boldsymbol{x}}_{s,k}$. In particular, following (4-44) the mean of $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) \sim GP\left(\boldsymbol{m}_\Delta(\bar{\boldsymbol{x}}_{s,k}), \boldsymbol{\Sigma}_\Delta(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})\right)$ is

$$\boldsymbol{m}_\Delta(\bar{\boldsymbol{x}}_{s,k}) = \boldsymbol{\mu}_w^T \bar{\boldsymbol{x}}_{s,k}. \tag{4-48}$$

Similarly, as of (4-47) the covariance is

$$\boldsymbol{\Sigma}_{\Delta}(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}) = \begin{bmatrix} \bar{\boldsymbol{x}}_{s,k}^T \boldsymbol{\Sigma}_{w,1,1} \bar{\boldsymbol{x}}_{s,k} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \bar{\boldsymbol{x}}_{s,k}^T \boldsymbol{\Sigma}_{w,l,l} \bar{\boldsymbol{x}}_{s,k} \end{bmatrix} \tag{4-49}$$

For a linear GP prior model, the dynamics of (4-39) become

$$\bar{\boldsymbol{x}}_{s,k+1} = \underbrace{\bar{\boldsymbol{A}}_{lin}(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{B}}_P \boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})}_{\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k})}. \tag{4-50}$$

Instead of only treating $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ as a GP, the whole right hand side of (4-50) can be included in a new variable $f_{lP}(\bar{\boldsymbol{x}}_{s,k})$, which in turn is expressed by the GP

$$\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k}) \sim GP\left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}), \boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k})\right). \tag{4-51}$$

Consequently, the mean of $\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k,l})$ is

$$\begin{aligned} \boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}) = \mathbb{E}_{\boldsymbol{f}_{lP}}[\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k})] &= \mathbb{E}_{\boldsymbol{f}_{lP}}[\bar{\boldsymbol{A}}_{lin}(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{B}}_P \boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})] \\ &= \bar{\boldsymbol{A}}_{lin}(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \bar{\boldsymbol{x}}_{s,k}. \end{aligned} \tag{4-52}$$

Similarly, the covariance is

$$\boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}) = \mathbb{E}_{\boldsymbol{f}_{lP}}\left[(\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k}) - \mathbb{E}_{\boldsymbol{f}_{lP}}[\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k})])(\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k}) - \mathbb{E}_{\boldsymbol{f}_{lP}}[\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k})])^T\right]. \tag{4-53}$$

The evaluation of one bracket of (4-53) yields

$$\begin{aligned} (\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k}) - \mathbb{E}_{\boldsymbol{f}_{lP}}[\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k})]) &= \bar{\boldsymbol{A}}_{lin}(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{B}}_P \boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) \\ &\quad -\bar{\boldsymbol{A}}_{lin}(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) - \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}_{s,k,l}) - \bar{\boldsymbol{B}}_P \boldsymbol{m}_{\Delta}(\bar{\boldsymbol{x}}_{s,k}) \\ &= \bar{\boldsymbol{B}}_P \boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) - \bar{\boldsymbol{B}}_P \boldsymbol{m}_{\Delta}(\bar{\boldsymbol{x}}_{s,k}) = \bar{\boldsymbol{B}}_P \left(\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) - \boldsymbol{m}_{\Delta}(\bar{\boldsymbol{x}}_{s,k})\right). \end{aligned} \tag{4-54}$$

Consequently, the covariance is

$$\begin{aligned} \boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}) &= \mathbb{E}_{\boldsymbol{f}_{lP}}\left[\bar{\boldsymbol{B}}_P \left(\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) - \boldsymbol{m}_{\Delta}(\bar{\boldsymbol{x}}_{s,k})\right)\left(\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) - \boldsymbol{m}_{\Delta}(\bar{\boldsymbol{x}}_{s,k})\right)^T \bar{\boldsymbol{B}}_P^T\right] \\ &= \bar{\boldsymbol{B}}_P \boldsymbol{\Sigma}_{\Delta}(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}) \bar{\boldsymbol{B}}_P^T. \end{aligned} \tag{4-55}$$

Propagating states through the system dynamics amounts to the evaluation of the Chapman-Kolmogorov equation in (4-2):

$$p(\bar{\boldsymbol{x}}_{s,k+1}) = \int p(\bar{\boldsymbol{x}}_{s,k+1}|\bar{\boldsymbol{x}}_{s,k}) p(\bar{\boldsymbol{x}}_{s,k}) d\bar{\boldsymbol{x}}_{s,k} \tag{4-56}$$

As in the previous case, $p(\bar{\boldsymbol{x}}_{s,k})$ is represented by a GMM $\hat{p}(\bar{\boldsymbol{x}}_{s,k+1})$ as in (4-5):

$$\hat{p}(\bar{\boldsymbol{x}}_{s,k+1}) = \int p(\bar{\boldsymbol{x}}_{s,k+1}|\bar{\boldsymbol{x}}_{s,k}) \hat{p}(\bar{\boldsymbol{x}}_{s,k}) d\bar{\boldsymbol{x}}_{s,k} \tag{4-57}$$

Here, the conditional distribution $p(\bar{\boldsymbol{x}}_{s,k+1}|\bar{\boldsymbol{x}}_{s,k})$ is represented by the GP $\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k})$ in (4-51), which models the system dynamics (4-50). For the conditional distribution $p(\bar{\boldsymbol{x}}_{s,k+1}|\bar{\boldsymbol{x}}_{s,k})$, $\bar{\boldsymbol{x}}_{s,k}$ is treated as though it was known. Hence, $p(\bar{\boldsymbol{x}}_{s,k+1}|\bar{\boldsymbol{x}}_{s,k})$ is the evaluation of $\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k})$ at the input $\bar{\boldsymbol{x}}_{s,k}$, which in turn is a normal distribution:

$$p(\bar{\boldsymbol{x}}_{s,k+1}|\bar{\boldsymbol{x}}_{s,k}) = N\left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}), \boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})\right) \tag{4-58}$$

By inserting (4-5) and (4-58), (4-57) becomes

$$\hat{p}(\bar{\boldsymbol{x}}_{s,k+1}) = \int N\left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}), \boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})\right) \sum_{i=1}^{m} \alpha_i N(\boldsymbol{\mu}_{k,i}, \boldsymbol{\Sigma}_{k,i})\, d\bar{\boldsymbol{x}}_{s,k} \tag{4-59}$$

Including the conditional in the sum results in

$$\hat{p}(\bar{\boldsymbol{x}}_{s,k+1}) = \int \sum_{i=1}^{m} \alpha_i N\left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k,i}), \boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k,i}, \bar{\boldsymbol{x}}'_{s,k,i})\right) N(\boldsymbol{\mu}_{k,i}, \boldsymbol{\Sigma}_{k,i})\, d\bar{\boldsymbol{x}}_{s,k}. \tag{4-60}$$

Finally, swapping the integral and the sum as well as factorizing the mixture weights α_i yields

$$\hat{p}(\bar{\boldsymbol{x}}_{s,k+1}) = \sum_{i=1}^{m} \alpha_i \int N\left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}), \boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})\right) N(\boldsymbol{\mu}_{k,i}, \boldsymbol{\Sigma}_{k,i}) d\bar{\boldsymbol{x}}_{s,k}. \tag{4-61}$$

Solving the integral is analytically intractable in most cases. Note that the approximate $\hat{p}(\bar{\boldsymbol{x}}_{s,k+1})$ still assumes the structure of a mixture of distributions, but not necessarily a GMM. To solve (4-61), note that (4-61) inherits the structure of (2-56), which represents solving a GP at an uncertain input. The authors in [171], [71], [53] showed how to analytically calculate the first and second central moment of the resulting distribution. Hence, here the product $N\left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}), \boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})\right) N(\boldsymbol{\mu}_{k,i}, \boldsymbol{\Sigma}_{k,i})$ is approximated by a Gaussian distribution $N(\boldsymbol{\mu}_{k+1,i}, \boldsymbol{\Sigma}_{k+1,i})$. This way $\hat{p}(\bar{\boldsymbol{x}}_{s,k+1})$ also inherits the basic structure of a GMM:

$$\hat{p}(\bar{\boldsymbol{x}}_{s,k+1}) = \sum_{i=1}^{m} \alpha_i N(\boldsymbol{\mu}_{k+1,i}, \boldsymbol{\Sigma}_{k+1,i}) \tag{4-62}$$

Equation (4-62) is achieved by noting that $N(\boldsymbol{\mu}_{k+1,i}, \boldsymbol{\Sigma}_{k+1,i})$ does not depend on $\bar{\boldsymbol{x}}_{s,k}$ and is moved in front of the integral in (4-61). In turn, the integral evaluates to one.

Solving (4-61) now amounts to the evaluation of the mean and covariance of each mixture component of $\hat{p}(\bar{\boldsymbol{x}}_{s,k+1})$. Since the conditional $p(\bar{\boldsymbol{x}}_{s,k+1}|\bar{\boldsymbol{x}}_{s,k})$ is well known, the approach of section 2.3.1, which employs the law of iterated expectation and total variance, is used (also see Appendix B.6). Remember that $\bar{\boldsymbol{x}}_{s,k+1}$ is given by $\boldsymbol{f}_{lP}$ as of (4-50). Hence, $p(\boldsymbol{f}_{lP}|\bar{\boldsymbol{x}}_{s,k}) = p(\bar{\boldsymbol{x}}_{s,k+1}|\bar{\boldsymbol{x}}_{s,k})$, implicitly.

Using the law of iterated expectations, the i –th mixture mean $\boldsymbol{\mu}_{k+1,i}$ is

$$\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}[\bar{\boldsymbol{x}}_{s,k+1}] = \boldsymbol{\mu}_{k+1,i} = \mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\mathbb{E}_{f_{lP}}[\boldsymbol{f}_{lP}|\bar{\boldsymbol{x}}_{s,k}]\right]. \tag{4-63}$$

Here, the subscript $\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}$ indicates that the expectation is taken with respect to the i –th mixture. Following section 2.3.1, the propagated expected value is

$$\boldsymbol{\mu}_{k+1,i} = \mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k})\right] = \bar{\boldsymbol{f}}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T\boldsymbol{\mu}_{k,i}. \tag{4-64}$$

The propagated covariance $\boldsymbol{\Sigma}_{k+1,i}$ is obtained by application of the law of total variance

$$\boldsymbol{\Sigma}_{k+1,i} = \mathbb{V}_{\boldsymbol{x}_{s,k|i}}\left[\mathbb{E}_{\boldsymbol{f}_{lP}}\left[\boldsymbol{f}_{lP}|\bar{\boldsymbol{x}}_{s,k}\right]\right] + \mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[\mathbb{V}_{f_{lP}}\left[\boldsymbol{f}_{lP}|\bar{\boldsymbol{x}}_{s,k}\right]\right] \tag{4-65}$$

The first term in (4-65) is

$$\begin{aligned}\mathbb{V}_{\boldsymbol{x}_{s,k|i}}\left[\mathbb{E}_{f_{lP}}\left[f_{lP}|\boldsymbol{x}_{s,k}\right]\right] &= \mathbb{V}_{\boldsymbol{x}_{s,k|i}}\left[\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k})\right]\\ = \mathbb{E}_{\boldsymbol{x}_{s,k|i}}&\left[\left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}) - \mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k})\right]\right)\left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}) - \mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k})\right]\right)^T\right].\end{aligned} \tag{4-66}$$

Inserting (4-52) into the first term of (4-66) as well as rearranging terms yields

$$\begin{aligned}\left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}) - \mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k})\right]\right) &= \bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i})(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{f}}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T\bar{\boldsymbol{x}}_{s,k}\\ &-\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i})(\boldsymbol{\mu}_{k,i} - \boldsymbol{\mu}_{k,i}) - \bar{\boldsymbol{f}}(\boldsymbol{\mu}_{k,i}) - \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T\boldsymbol{\mu}_{k,i}\\ &= \bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i})(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T\boldsymbol{\mu}_{k,i}\\ &= \bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i})(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i})\\ &= (\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T)(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i}),\end{aligned} \tag{4-67}$$

Note that (4-65) implicitly assumes that the linear model of (4-50) is evaluated at the mean of each mixture component such that $\bar{\boldsymbol{x}}_{s,k,l} = \boldsymbol{\mu}_{k,i}$. Inserting (4-65) into (4-66) yields

$$\begin{aligned}&\mathbb{V}_{\boldsymbol{x}_{s,k|i}}\left[\mathbb{E}_{f_{lP}}\left[f_{lP}|\boldsymbol{x}_{s,k}\right]\right]\\ &= \mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[\left((\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T)(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i})\right)\left((\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T)(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i})\right)^T\right]\\ &= (\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T)\underbrace{\mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i})(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i})^T\right]}_{\boldsymbol{\Sigma}_{k,i}}(\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T)^T\end{aligned} \tag{4-68}$$

Hence, the first term of (4-65) evaluates to

$$\mathbb{V}_{\boldsymbol{x}_{s,k|i}}\left[\mathbb{E}_{f_{lP}}\left[f_{lP}|\boldsymbol{x}_{s,k}\right]\right] = (\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T)\boldsymbol{\Sigma}_{k,i}(\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T)^T. \tag{4-69}$$

The second term of (4-65) is

$$\begin{aligned}\mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[\mathbb{V}_{f_{lP}}\left[f_{lP}|\boldsymbol{x}_{s,k}\right]\right] &= \mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[\boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})\right] = \mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[\bar{\boldsymbol{B}}_P\boldsymbol{\Sigma}_\Delta(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})\bar{\boldsymbol{B}}_P^T\right]\\ &= \bar{\boldsymbol{B}}_P\mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[\boldsymbol{\Sigma}_\Delta(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})\right]\bar{\boldsymbol{B}}_P^T.\end{aligned} \tag{4-70}$$

Using the expectation of quadratic forms (see section 2.3.1 or Appendix B.10) on (4-49) yields

$$\boldsymbol{\Sigma}_{\overline{\boldsymbol{x}}_{s,k|i|\Delta}} = \mathbb{E}_{\overline{\boldsymbol{x}}_{s,k|i}}\left[\boldsymbol{\Sigma}_{\Delta}\left(\overline{\boldsymbol{x}}_{s,k}, \overline{\boldsymbol{x}}'_{s,k}\right)\right] = \begin{bmatrix} \mathbb{E}_{\overline{\boldsymbol{x}}_{s,k|i}}\left[\overline{\boldsymbol{x}}_{s,k}^T \boldsymbol{\Sigma}_{w,1,1} \overline{\boldsymbol{x}}_{s,k}\right] & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathbb{E}_{\overline{\boldsymbol{x}}_{s,k|i}}\left[\overline{\boldsymbol{x}}_{s,k}^T \boldsymbol{\Sigma}_{w,l,l} \overline{\boldsymbol{x}}_{s,k}\right] \end{bmatrix}$$

$$= \begin{bmatrix} \mathrm{Tr}\left(\boldsymbol{\Sigma}_{w,1,1}\boldsymbol{\Sigma}_{k,i}\right) + \boldsymbol{\mu}_{k,i}^T \boldsymbol{\Sigma}_{w,1,1} \boldsymbol{\mu}_{k,i} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathrm{Tr}\left(\boldsymbol{\Sigma}_{w,l,l}\boldsymbol{\Sigma}_{k,i}\right) + \boldsymbol{\mu}_{k,i}^T \boldsymbol{\Sigma}_{w,l,l} \boldsymbol{\mu}_{k,i} \end{bmatrix}. \qquad \textbf{(4-71)}$$

By inserting (4-69) and (4-71) in (4-65),the propagated covariance of the i –th mixture element is

$$\boldsymbol{\Sigma}_{k+1,i} = \left(\overline{\boldsymbol{A}}_{lin}\left(\boldsymbol{\mu}_{k,i}\right) + \overline{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T\right)\boldsymbol{\Sigma}_{k,i}\left(\overline{\boldsymbol{A}}_{lin}\left(\boldsymbol{\mu}_{k,i}\right) + \overline{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T\right)^T + \overline{\boldsymbol{B}}_P \boldsymbol{\Sigma}_{\boldsymbol{x}_{s,k|i|\Delta}} \overline{\boldsymbol{B}}_P^T \qquad \textbf{(4-72)}$$

Hence, the total predictive GMM is

$$\hat{p}\left(\overline{\boldsymbol{x}}_{s,k+1}\right) = \sum_{i=1}^{m} \alpha_i N \left(\begin{matrix} \overline{\boldsymbol{f}}\left(\boldsymbol{\mu}_{k,i}\right) + \overline{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \boldsymbol{\mu}_{k,i}, \\ \left(\overline{\boldsymbol{A}}_{lin}\left(\boldsymbol{\mu}_{k,i}\right) + \overline{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T\right)\boldsymbol{\Sigma}_{k,i}\left(\overline{\boldsymbol{A}}_{lin}\left(\boldsymbol{\mu}_{k,i}\right) + \overline{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T\right)^T + \overline{\boldsymbol{B}}_P \boldsymbol{\Sigma}_{\boldsymbol{x}_{s,k|i|\Delta}} \overline{\boldsymbol{B}}_P^T \end{matrix} \right) \qquad \textbf{(4-73)}$$

The following example shows how GMMs can be used for the evaluation of a GP.

Example 4-5: **Evaluation of a GP prior model with linear kernels at a GMM distributed input**

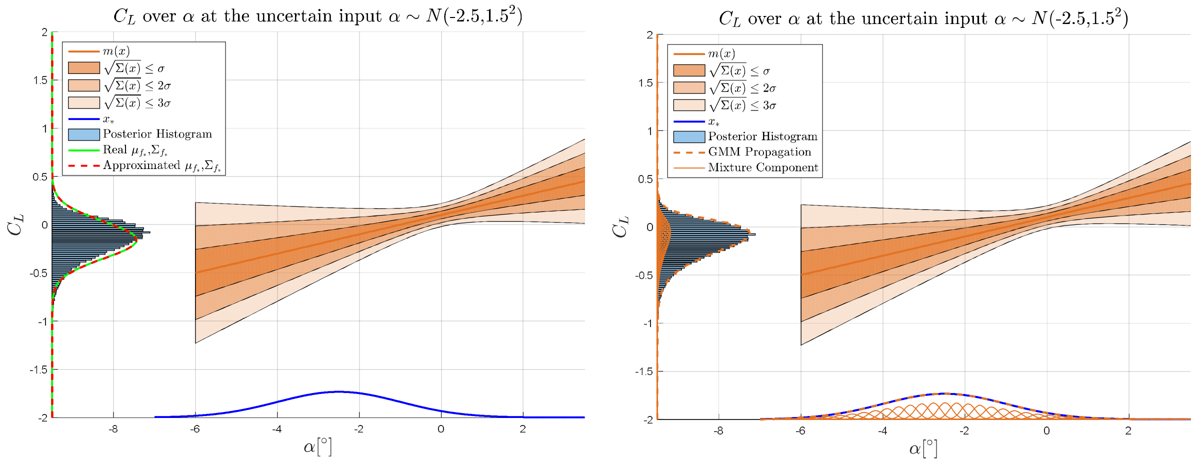


Figure 4-8: C_L over α modeled by a GP and evaluated at an a GMM distributed input

Consider Example 2-3, where C_L over α is modeled as a GP. Therefore, the input distribution $p(\alpha) = N(-2.5, 1.5^2)$ is approximated by a GMM. The latter is chosen to consist of 40 mixture components with a standard deviation of $\sigma_i = 0.175$, which approximate $p(x)$ for $x \in [-7{,}4]$. The fitting is performed by solving (2-5). The update laws (4-64) and (4-72) are used in order to propagate the GMM through the GP.

Figure 4-8 compares the approximation of the predictive distribution $\hat{p}(C_L)$ by a Gaussian and a GMM. In that, Figure 4-8 (left) shows the results of Figure 2-8 if the predictive distribution is approximated by a single Gaussian. Figure 4-8 (right) shows the propagation result if a GMM is used. It can be seen that the state distribution $\hat{p}(C_L)$ in the latter case is nearly indistinguishable from the sample based propagation, while it shows a significant error if the predictive is approximated by a single Gaussian. This can be attributed to the fact, that solving (4-57) implicitly requires marginalizing over the GP. In the first case, the predictive distribution is the result of averaging over all possible GP results given the input distribution. In the second case, the input space is significantly reduced as each mixture component is treated separately. As a result, the GP is averaged over a smaller domain, which increases the accuracy. Putting together all mixture components thus results in a better estimate.

4.4.2 Gaussian Process Models with nonlinear regressor functions

Often, the uncertainty is represented by a linear combination of weights and nonlinear regressor functions instead of linear ones. This section expands the results of section 4.4.1 in order to include nonlinear regressor functions. Therefore, consider the linearized plant dynamics in (4-39). The modelling uncertainty $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ now is $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) = \boldsymbol{w}^T\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k})$. As a result, the dynamics of (4-39) become

$$\bar{\boldsymbol{x}}_{s,k+1} = \bar{\boldsymbol{A}}_{lin}(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{B}}_P\boldsymbol{w}^T\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k}). \tag{4-74}$$

By placing a probability distribution on the parameters $\boldsymbol{w}$ with (4-40), $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ becomes a GP model as of (2-20) with mean (4-44) and covariance (4-47). Consider a Taylor series expansion of $\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})$ with respect to $\bar{\boldsymbol{x}}_{sl}$:

$$\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k}) = \boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k,l}) + \left.\frac{d\boldsymbol{\Delta}(\bar{\boldsymbol{x}}_{s,k})}{d\bar{\boldsymbol{x}}_{s,k}}\right|_{\bar{\boldsymbol{x}}_{s,k}=\bar{\boldsymbol{x}}_{s,k,l}} (\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) + O^2(\bar{\boldsymbol{x}}_{s,k}). \tag{4-75}$$

Omitting the higher order terms yields a first order approximation $\boldsymbol{\Delta}_l(\bar{\boldsymbol{x}}_{s,k})$ of the modelling uncertainty:

$$\boldsymbol{\Delta}_l(\bar{\boldsymbol{x}}_{s,k}) = \boldsymbol{w}^T\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k,l}) + \boldsymbol{w}^T \left.\frac{d\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k})}{d\bar{\boldsymbol{x}}_{s,k}}\right|_{\bar{\boldsymbol{x}}_{s,k}=\bar{\boldsymbol{x}}_{s,k,l}} (\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}). \tag{4-76}$$

For the ease of readability, let $\boldsymbol{\phi}_d(\bar{\boldsymbol{x}}_{s,k,l}) = \left.\frac{d\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k})}{d\bar{\boldsymbol{x}}_{s,k}}\right|_{\bar{\boldsymbol{x}}_{s,k}=\bar{\boldsymbol{x}}_{s,k,l}}$. The structure of the covariance matrix $\boldsymbol{\Sigma}_{\Delta_l}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})$ is given by (4-47). In particular, the i –th entry on the main diagonal is

$$\begin{aligned} &\Sigma_{\Delta_l,i,i}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k}) \\ &= \left(\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k,l}) + \boldsymbol{\phi}_d(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l})\right)^T \boldsymbol{\Sigma}_{w,i,i} \left(\boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k,l}) + \boldsymbol{\phi}_d(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l})\right). \end{aligned} \tag{4-77}$$

By using the first order Taylor series approximation in (4-76) the dynamics of (4-74) become

$$\bar{\boldsymbol{x}}_{s,k+1} = \underbrace{\begin{array}{c} \bar{\boldsymbol{A}}_{lin}(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}_{s,k,l}) \\ +\bar{\boldsymbol{B}}_P \boldsymbol{w}^T \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{B}}_P \boldsymbol{w}^T \boldsymbol{\phi}_d(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) \end{array}}_{\boldsymbol{f}_{lP}}. \tag{4-78}$$

Similar to section 4.4.1, the right hand side of (4-78) forms a new variable $f_{lP}(\bar{\boldsymbol{x}}_{s,k})$, which is expressed by the GP

$$\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k}) \sim GP\left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}), \boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k})\right). \tag{4-79}$$

The mean of $\boldsymbol{f}_{lP}(\bar{\boldsymbol{x}}_{s,k})$ is

$$\begin{gathered} \boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}) = \bar{\boldsymbol{A}}_{lin}(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l}) + \bar{\boldsymbol{f}}(\bar{\boldsymbol{x}}_{s,k,l}) \\ +\bar{\boldsymbol{B}}_P\left(\boldsymbol{\mu}_w^T \boldsymbol{\phi}(\bar{\boldsymbol{x}}_{s,k,l}) + \boldsymbol{\mu}_w^T \boldsymbol{\phi}_d(\bar{\boldsymbol{x}}_{s,k,l})(\bar{\boldsymbol{x}}_{s,k} - \bar{\boldsymbol{x}}_{s,k,l})\right). \end{gathered} \tag{4-80}$$

Similarly, following (4-55), the covariance is

$$\boldsymbol{\Sigma}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k}) = \bar{\boldsymbol{B}}_P \boldsymbol{\Sigma}_{\Delta_l}(\bar{\boldsymbol{x}}_{s,k}, \bar{\boldsymbol{x}}'_{s,k}) \bar{\boldsymbol{B}}_P^T. \tag{4-81}$$

In order to calculate $\hat{p}(\bar{\boldsymbol{x}}_{s,k+1})$, the next steps follow the approach in section 4.4.1. Thereby, similar to the GMM forecast in section 4.3, the first order Taylor series approximation in (4-76) is performed with respect to the mean of the i –th mixture component $\bar{\boldsymbol{x}}_{sl} = \boldsymbol{\mu}_{k,i}$. By (4-64), the propagated mean is

$$\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}[\bar{\boldsymbol{x}}_{s,k+1}] = \boldsymbol{\mu}_{k+1,i} = \mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\mathbb{E}_{f_{lP}}[\boldsymbol{f}_{lP}|\bar{\boldsymbol{x}}_{s,k}]\right] = \bar{\boldsymbol{f}}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \boldsymbol{\phi}(\boldsymbol{\mu}_{k,i}). \tag{4-82}$$

Applying the law of total variance as in (4-65), again results in the propagated covariance matrix. Thereby, the first term of (4-66), which in turn evaluates the first term of (4-65), is

$$\begin{gathered} \left(\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k}) - \mathbb{E}_{\boldsymbol{x}_{s,k|i}}[\boldsymbol{m}(\bar{\boldsymbol{x}}_{s,k})]\right) = \bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i})(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{f}}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \boldsymbol{\phi}(\boldsymbol{\mu}_{k,i}) \\ +\bar{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \boldsymbol{\phi}_d(\boldsymbol{\mu}_{k,i})(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i}) - \bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i})(\boldsymbol{\mu}_{k,i} - \boldsymbol{\mu}_{k,i}) - \bar{\boldsymbol{f}}(\boldsymbol{\mu}_{k,i}) - \bar{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \boldsymbol{\phi}(\boldsymbol{\mu}_{k,i}) \\ -\bar{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \boldsymbol{\phi}_d(\boldsymbol{\mu}_{k,i})(\boldsymbol{\mu}_{k,i} - \boldsymbol{\mu}_{k,i}) \\ = \bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i})(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \boldsymbol{\phi}_d(\boldsymbol{\mu}_{k,i})(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i}) \\ = \left(\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \boldsymbol{\phi}_d(\boldsymbol{\mu}_{k,i})\right)(\bar{\boldsymbol{x}}_{s,k} - \boldsymbol{\mu}_{k,i}). \end{gathered} \tag{4-83}$$

Following (4-68), the first term of (4-66) becomes

$$\begin{gathered} \mathbb{V}_{\boldsymbol{x}_{s,k|i}}\left[\mathbb{E}_{f_{lP}}[f_{lP}|\boldsymbol{x}_{s,k}]\right] \\ = \left(\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \boldsymbol{\phi}_d(\boldsymbol{\mu}_{k,i})\right) \boldsymbol{\Sigma}_{k,i} \left(\bar{\boldsymbol{A}}_{lin}(\boldsymbol{\mu}_{k,i}) + \bar{\boldsymbol{B}}_P \boldsymbol{\mu}_w^T \boldsymbol{\phi}_d(\boldsymbol{\mu}_{k,i})\right)^T. \end{gathered} \tag{4-84}$$

The second term of (4-66) is

$$\begin{aligned}\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\mathbb{V}_{f_{lP}}\left[f_{lP}|\boldsymbol{x}_{s,k}\right]\right] &= \mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\boldsymbol{\Sigma}\left(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}\right)\right] = \mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\bar{\boldsymbol{B}}_P\boldsymbol{\Sigma}_{\Delta_l}\left(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}\right)\bar{\boldsymbol{B}}_P^T\right] \\ &= \bar{\boldsymbol{B}}_P\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\boldsymbol{\Sigma}_{\Delta_l}\left(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}\right)\right]\bar{\boldsymbol{B}}_P^T\end{aligned} \tag{4-85}$$

Calculating $\mathbb{E}_{\boldsymbol{x}_{s,k|i}}\left[\boldsymbol{\Sigma}_{\Delta_l}\left(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}\right)\right]$ is shown in (4-71) and amounts to taking the expectation of each entry in $\boldsymbol{\Sigma}_{\Delta_l}\left(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}\right)$. Consider the i –th entry on the main diagonal of $\boldsymbol{\Sigma}_{\Delta_l}\left(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}\right)$ as given by (4-77). Taking its expectation yields

$$\begin{aligned}&\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\Sigma_{\Delta_l,i,i}\left(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}\right)\right] \\ &= \mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\left(\boldsymbol{\phi}\left(\boldsymbol{\mu}_{k,i}\right)+\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\left(\bar{\boldsymbol{x}}_{s,k}-\boldsymbol{\mu}_{k,i}\right)\right)^T\boldsymbol{\Sigma}_{w,i,i}\left(\boldsymbol{\phi}\left(\boldsymbol{\mu}_{k,i}\right)+\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\left(\bar{\boldsymbol{x}}_{s,k}-\boldsymbol{\mu}_{k,i}\right)\right)\right].\end{aligned} \tag{4-86}$$

For the ease of readability, let $\boldsymbol{\phi}_0\left(\boldsymbol{\mu}_{k,i}\right)=\boldsymbol{\phi}\left(\boldsymbol{\mu}_{k,i}\right)-\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\mu}_{k,i}$ and note that $\boldsymbol{\phi}_0$ is independent of $\bar{\boldsymbol{x}}_{s,k}$. Hence, (4-86) becomes

$$\begin{aligned}&\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\Sigma_{\Delta_l,i,i}\left(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}\right)\right] \\ &= \mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\left(\boldsymbol{\phi}_0\left(\boldsymbol{\mu}_{k,i}\right)+\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\bar{\boldsymbol{x}}_{s,k}\right)^T\boldsymbol{\Sigma}_{w,i,i}\left(\boldsymbol{\phi}_0\left(\boldsymbol{\mu}_{k,i}\right)+\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\bar{\boldsymbol{x}}_{s,k}\right)\right] \\ &= \mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\boldsymbol{\phi}_0^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_0\left(\boldsymbol{\mu}_{k,i}\right)\right]+\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\bar{\boldsymbol{x}}_{s,k}^T\boldsymbol{\phi}_d^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_0\left(\boldsymbol{\mu}_{k,i}\right)\right] \\ &+\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\boldsymbol{\phi}_0^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\bar{\boldsymbol{x}}_{s,k}\right]+\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\bar{\boldsymbol{x}}_{s,k}^T\boldsymbol{\phi}_d^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\bar{\boldsymbol{x}}_{s,k}\right].\end{aligned} \tag{4-87}$$

The first, second and third term in (4-87) linearly depend on $\bar{\boldsymbol{x}}_{s,k}$. Hence, evaluating those terms is straightforward by noting that $\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\bar{\boldsymbol{x}}_{s,k}\right]=\boldsymbol{\mu}_{k,i}$. The fourth term is solved by using the expectation of quadratic forms (see section 2.3.1 or Appendix B.10):

$$\begin{aligned}&\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\bar{\boldsymbol{x}}_{s,k}^T\boldsymbol{\phi}_d^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\bar{\boldsymbol{x}}_{s,k}\right] \\ &= \mathrm{Tr}\left(\boldsymbol{\phi}_d^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{k,i}\right)+\boldsymbol{\mu}_{k,i}^T\boldsymbol{\phi}_d^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\mu}_{k,i}.\end{aligned} \tag{4-88}$$

Hence, (4-87) becomes

$$\begin{aligned}\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\Sigma_{\Delta_l,i,i}\left(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}\right)\right] &= \boldsymbol{\phi}_0^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_0\left(\boldsymbol{\mu}_{k,i}\right)+\boldsymbol{\mu}_{k,i}^T\boldsymbol{\phi}_d^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_0\left(\boldsymbol{\mu}_{k,i}\right) \\ &+\boldsymbol{\phi}_0^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\mu}_{k,i}+\mathrm{Tr}\left(\boldsymbol{\phi}_d^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{k,i}\right) \\ &+\boldsymbol{\mu}_{k,i}^T\boldsymbol{\phi}_d^T\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\Sigma}_{w,i,i}\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\boldsymbol{\mu}_{k,i}.\end{aligned} \tag{4-89}$$

Similar to (4-71), let $\boldsymbol{\Sigma}_{\bar{\boldsymbol{x}}_{s,k|i|\Delta}}=\mathbb{E}_{\bar{\boldsymbol{x}}_{s,k|i}}\left[\boldsymbol{\Sigma}_{\Delta}\left(\bar{\boldsymbol{x}}_{s,k},\bar{\boldsymbol{x}}'_{s,k}\right)\right]$. By using (4-84) as well as (4-85) and (4-89) the propagated covariance is

$$\begin{aligned}\boldsymbol{\Sigma}_{k+1,i} &= \left(\bar{\boldsymbol{A}}_{lin}\left(\boldsymbol{\mu}_{k,i}\right)+\bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\right)\boldsymbol{\Sigma}_{k,i}\left(\bar{\boldsymbol{A}}_{lin}\left(\boldsymbol{\mu}_{k,i}\right)+\bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\right)^T \\ &+\bar{\boldsymbol{B}}_P\boldsymbol{\Sigma}_{\boldsymbol{x}_{s,k|i|\Delta}}\bar{\boldsymbol{B}}_P^T.\end{aligned} \tag{4-90}$$

With the mean (4-82) and the covariance (4-90), the total predictive GMM for nonlinear regressor functions is

$$\begin{aligned}&\hat{p}\left(\bar{\boldsymbol{x}}_{s,k+1}\right)\\&=\sum_{i=1}^{m}\alpha_i N\left(\begin{array}{c}\bar{\boldsymbol{f}}\left(\boldsymbol{\mu}_{k,i}\right)+\bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T\boldsymbol{\phi}\left(\boldsymbol{\mu}_{k,i}\right),\\ \left(\bar{\boldsymbol{A}}_{lin}\left(\boldsymbol{\mu}_{k,i}\right)+\bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\right)\boldsymbol{\Sigma}_{k,i}\left(\bar{\boldsymbol{A}}_{lin}\left(\boldsymbol{\mu}_{k,i}\right)+\bar{\boldsymbol{B}}_P\boldsymbol{\mu}_w^T\boldsymbol{\phi}_d\left(\boldsymbol{\mu}_{k,i}\right)\right)^T+\bar{\boldsymbol{B}}_P\boldsymbol{\Sigma}_{\boldsymbol{x}_{s,k|i|\Delta}}\bar{\boldsymbol{B}}_P^T\end{array}\right).\end{aligned} \tag{4-91}$$

4.4.3 Online GP regression for uncertainty quantification

Until know the uncertainty $\boldsymbol{\Delta}$ was assumed to be structurally known. This is the case in the presence of parameter uncertainties. However, often unmodelled or completely unknown dynamics affect the system. Neglecting the latter can lead to serious misjudgement in practice. One possibility to deal with this problem is to estimate bounds for the uncertainty and include them as an uncertainty in the forecast. Naturally, this approach leads to conservative estimates, which again might lead to physically irrelevant conclusions. Instead, the GP framework allows for the quantification of a model by using online gathered data. The key process behind this is GP regression as introduced in section 2.2.1.

Figure 4-9 shows the structure of the proposed online uncertainty quantification approach. As shown in section 2.2.1, GP regression uses pairs of input-output data points to generate a belief of an underlying function. Hence, for the data collection only pointwise realizations of the system signals are required. The index k indicates that a required quantity is evaluated at the specific time instant k,.

The uncertainty quantification approach consists of four major parts. The block “uncertainty calculation” requires information about the current system state $\boldsymbol{x}_{P,k}$, the control input $\boldsymbol{u}_k$ as well as the state derivative $\dot{\boldsymbol{x}}_{P,k}$. Note that using the If $\dot{\boldsymbol{x}}_{P,k}$ is not measurable, it has to be estimated from $\boldsymbol{x}_{P,k}$ (or the measurement output $\boldsymbol{y}_k$ respectively) using appropriate filtering techniques in the “Filter” block. Since computational resources are limited not every data point can be stored. The “Data Management” block decides whether a new data point is incorporated into the history stacks $\boldsymbol{X}_H, \boldsymbol{Y}_H$ or dropped. Finally, “Gaussian Process Regression” generates a belief on the uncertainty $\boldsymbol{\Delta}(\boldsymbol{x}_P, \boldsymbol{u})$ based on the stored data. The following sections detail the four parts of the uncertainty quantification approach.

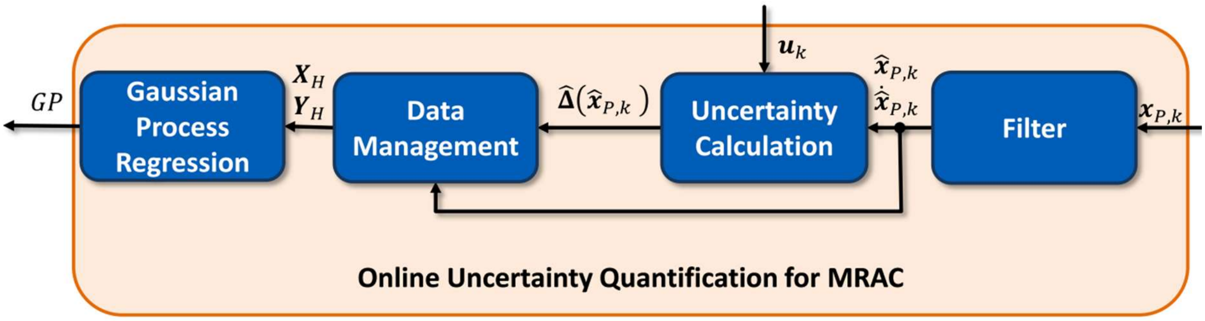

GP
Gaussian Process Regression
$\boldsymbol{X}_H$
$\boldsymbol{Y}_H$
Data Management
$\hat{\Delta}(\hat{\boldsymbol{x}}_{P,k})$
$\boldsymbol{u}_k$
Uncertainty Calculation
$\hat{\boldsymbol{x}}_{P,k}$
$\hat{\dot{\boldsymbol{x}}}_{P,k}$
Filter
$\boldsymbol{x}_{P,k}$
Online Uncertainty Quantification for MRAC

Figure 4-9: Uncertainty Quantification for the application to MRAC

4.4.3.1 Derivative Estimation through fixed-point smoothing

The uncertainty calculation requires information about the first time derivative of the system state $\dot{\boldsymbol{x}}_P$ of the closed loop dynamics in (3-13). Dependent on the control problem as well as the installed hardware, sometimes the state derivatives can be measured directly. In velocity control problems, accelerometers measure the accelerations of the system. On the other hand, new developments in the field of laser gyroscopes enable the measurement of rotational rates.

In most cases the state derivative cannot be measured directly and it has to be estimated. Several techniques exist that allow the estimation of state derivatives. These filters can be classified into techniques, which instantly yield a solution and methods that require time in order to arrive at an estimate $\dot{\hat{\boldsymbol{x}}}_P$.

A member of the former class is the Nonlinear Adaptive Variable Structure Derivative Estimator by Xu et al. ([108]). The authors employ multiple basic functions such as a switching gain, a saturation block and a deadzone modification in order to arrive at a suitable structure of a model, which estimates the state derivative. The parameters of the underlying functions are adapted online. While the state derivative is available directly, the approach has shown to be susceptible to noise.

A representative of the later class is *Optimal Fixed Point Smoothing* (OFPS), which is based on Bayesian filtering and described by i.a. Gelb ([70]). The OFPS consists of two parts. Up to the current time t_k, a forward Kalman Filter estimates $\dot{\hat{\boldsymbol{x}}}_P(t_k)$. A backward Kalman Filter operates on the time window $t_k \dots t_k + \Delta\mathrm{T}$ in order to arrive at a point estimate of $\dot{\hat{\boldsymbol{x}}}_P(t_k)$. As a result, the estimates of the state derivative are delayed by $\Delta\mathrm{T}$. In the past, authors successfully employed the OFPS in various applications including flight tests ([43], [38], [79]). Further techniques with potential to arrive at a high quality estimate of the state derivative are GP filtering by Deisenroth et al. ([54]) and the results of Solak et al. ([198]).

The uncertainty quantification algorithm does not require the data points to be available instantly. This allows for the application of time consuming techniques, which tend to achieve improved estimation performance when compared to methods that yield a solution instantly. Therefore, this thesis employs the OFPS. The necessary equations required to build and implement a discrete time OFPS are summarized in Appendix E.

4.4.3.2 Uncertainty Calculation

The modelling uncertainty at time k is calculated by solving (3-13) to $\widehat{\boldsymbol{\Delta}}(\hat{\boldsymbol{x}}_P)$:

$$\widehat{\boldsymbol{\Delta}}(\hat{\boldsymbol{x}}_{P,k}) = \boldsymbol{B}_P^{+}\left(\boldsymbol{A}_P\hat{\boldsymbol{x}}_{P,k} + \boldsymbol{B}_P\boldsymbol{K}_e^T\hat{\boldsymbol{e}}_k + \boldsymbol{B}_P\boldsymbol{K}_{rm,x}^T\boldsymbol{x}_{rm,k} + \boldsymbol{B}_P\boldsymbol{K}_{rm,r}^T\boldsymbol{r}_k + \boldsymbol{B}_P\boldsymbol{u}_{ad,k} - \dot{\hat{\boldsymbol{x}}}_{P,k}\right). \qquad \textbf{(4-92)}$$

Here, $\boldsymbol{B}_P^{+} = (\boldsymbol{B}_P^T\boldsymbol{B}_P)^{-1}\boldsymbol{B}_P^T$ denotes the Moore-Penrose pseudo-inverse of $\boldsymbol{B}_P$ (see e.g. [15]). Note, that the utilization of the OFPS for the state derivative estimation $\dot{\hat{\boldsymbol{x}}}_{P,k}$ induces a time delay, due to the backward iterated KF. In the actual application, other signals need to be delayed by the same amount of time. Furthermore, note that for the uncertainty calculation also the smoothed estimates $\hat{\boldsymbol{x}}_{P,k}$ of $\boldsymbol{x}_{P,k}$ can be used, as the latter are not necessarily available due to the output model.

4.4.3.3 Data Management for GP Regression

GP regression relies on a set of input-output data in order to form a belief on the uncertainty $\boldsymbol{\Delta}(\boldsymbol{x}_P)$. As introduced in section 2.2.1, the data is stored in the history stacks $\boldsymbol{X}_H$ and $\boldsymbol{Y}_H$. Here, the smoothed states $\hat{\boldsymbol{x}}_{P,k}$ form the input data $\boldsymbol{X}_H$ and the estimated uncertainty $\widehat{\boldsymbol{\Delta}}(\hat{\boldsymbol{x}}_{P,k})$

of (4-92) forms the output data $\boldsymbol{Y}_H$. Since computational resources are limited, only a finite number of p_H data points can be stored. As a consequence, the question arises, which data points to pick. In the following let $p_c \leq p_H$ denote the number of data points, which are currently stored in the history stacks $(\boldsymbol{X}_H, \boldsymbol{Y}_H)$.

In the beginning, the history stacks are empty with $p_c < p_H$. Let $\boldsymbol{x}^*$ denote the candidate (input) data point, which is tested for storage in $\boldsymbol{X}_H$. A natural way is to store each incoming data point $\boldsymbol{x}^*$ until the history stack $\boldsymbol{X}_H$ is full. This approach is often undesirable, as redundant information is accumulated. In contrast, a more suitable method is to only pick points, which contain yet unknown information when compared to the already existing history stacks. To achieve this, it is checked how well the candidate data point $\boldsymbol{x}^*$ is already represented by the existing elements in $\boldsymbol{X}_H$. Following this argument, Csato introduced the kernel linear independence test ([48]). The derivation in [48] is based on the representability of the GP within the Kernel-Hilbert space given the current history stack. The underlying concept is to evaluate the basis spanned by the kernel functions within the feature space. A candidate data point is only added to the history stack if it sufficiently increases the basis and thus the representability of the GP. The derivation in [48] results in the indicator

$$\gamma = \boldsymbol{k}_{\boldsymbol{x}^*\boldsymbol{x}^{*\prime}} - \boldsymbol{k}_{\boldsymbol{x}^*\boldsymbol{X}_H}\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{X}_H}\boldsymbol{k}_{\boldsymbol{X}_H\boldsymbol{x}^*}. \tag{4-93}$$

The candidate data point $\boldsymbol{x}^*$ is only added to the history stack $\boldsymbol{X}_H$ if the linear independence test exceeds a predefined limit, i.e. $\gamma > \epsilon_{dp}$ for some ϵ_{dp}. Consequently, if a candidate (input) data point $\boldsymbol{x}^*$ is added to $\boldsymbol{X}_H$, its respective uncertainty estimate $\widehat{\boldsymbol{\Delta}}(\boldsymbol{x}^*)$ is added to $\boldsymbol{Y}_H$.

Once the history stack $\boldsymbol{X}_H$ is full, i.e. $p_c = p_H$, the question arises how to deal with newly arriving data points. Several approaches exist. The simplest approach is to employ a static history stack. Thereby, once the history stack $\boldsymbol{X}_H$ is full, all consecutive data points are discarded. This approach is easy to implement and only requires low computational ressources. However, this technique only leads to local uncertainty approximation, as data points outside of the region of $\boldsymbol{X}_H$ are not considered. Furthermore, a change of the uncertainty over time cannot be captured.

The second method is to employ a cyclic history stack [38]. Thereby, once a new data point arrives, which fulfils the kernel linear independence test in (4-93), the oldest stored data point is deleted and the new one is added. This approach is easy to implement while still exhibiting low computational costs. However, it might delete information, which is still critical to the GP.

A third option is to employ the Sparse Online Gaussian Process Algorithm ([48], [38], [37], [194]). The approach only considers candidate data points, which already pass the kernel linear independence test in (4-93). First the candidate point is appended to the existing $\boldsymbol{X}_H$. The next step constructs p_H new GPs by deleting a different single data point. The Kullback-Leibler-divergence between the original (based on $\boldsymbol{X}_H$) and each newly formed GP is calculated. The algorithm then selects the GP, which exhibits the largest calculated Kullback-Leibler-divergence. The underlying idea behind this approach is to find the representation to which the candidate data point contributes the most. The Sparse Online Gaussian Process Algorithm automatically chooses the GP with the largest change to the former one. This way it keeps data, which is still important, which might have already been deleted by the cyclic approach. However, it exhibits higher computational costs than the static or cyclic history stack.

The mentioned list of techniques, which is by no means exhaustive, probably represents the most common approaches for data management in GP regression. Further methods can be found in i.e. [172].

4.4.3.4 GP Regression applied to MRAC

The underlying assumption in GP regression is that the uncertainty is modelled by a GP such that:

$$\boldsymbol{\Delta}(\widehat{\boldsymbol{x}}_P) \sim GP(\boldsymbol{m}(\widehat{\boldsymbol{x}}_P), \boldsymbol{\Sigma}(\widehat{\boldsymbol{x}}_P, \widehat{\boldsymbol{x}}'_P)). \tag{4-94}$$

The final step in GP regression utilizes the history stack pair $(\boldsymbol{X}_H, \boldsymbol{Y}_H)$ in order to infer a belief $\widehat{\boldsymbol{\Delta}}(\widehat{\boldsymbol{x}}_P)$ on the uncertainty $\boldsymbol{\Delta}(\widehat{\boldsymbol{x}}_P)$:

$$\widehat{\boldsymbol{\Delta}}(\widehat{\boldsymbol{x}}_P) \sim GP(\widehat{\boldsymbol{m}}(\widehat{\boldsymbol{x}}_P), \widehat{\boldsymbol{\Sigma}}(\widehat{\boldsymbol{x}}_P, \widehat{\boldsymbol{x}}'_P)). \tag{4-95}$$

The solution for the regression problem is given in (2-53) and (2-54). The following example is intended to create an understanding of how the presented online GP regression of section 4.4.3 generates a belief on the uncertainty $\boldsymbol{\Delta}$ during controller runtime.

Example 4-6: **GP Regression example applied to MRAC**

Consider the simplified roll dynamics of a fixed-wing aircraft:

$$\begin{bmatrix} \dot{\Phi} \\ \dot{p}_B \end{bmatrix} = \boldsymbol{A}_P \begin{bmatrix} \Phi \\ p_B \end{bmatrix} + \boldsymbol{B}_P\big(u - \Delta(\Phi)\big). \tag{4-96}$$

Here, p_B denotes the roll rate, Φ denotes the roll angle and u represents the system input, which is given by the aileron deflection in reality. The input matrix $\boldsymbol{B}_P$ represents the roll moment generated by an aileron deflection. Assume a generic uncertainty of the form

$$\Delta(\Phi) = \Theta_0^* + \Theta_1^*\Phi + \Theta_2^*\Phi^3 + \Theta_3^*\left(\frac{1}{1+e^{-\Phi}} + 0.1\cos\left(\frac{1}{6}\Phi\right)\right). \tag{4-97}$$

The input Φ to the uncertainty is assumed to be given in degrees. The system is subject to measurement noise $\boldsymbol{v}$, where $\boldsymbol{v} \sim N\left(\begin{bmatrix} 0° \\ 0\frac{°}{s} \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$. Consider a reference model of the form (3-7) as well as the control law in (3-11) and the direct MRAC update laws with σ-modification of (3-19).

The simulation runs for a total of $40s$ with a time step of $0.001s$. The reference command is comprised of consecutive step commands with an amplitude of $0°, 30°$ and $-30°$, each lasting $5s$. The initial condition of the reference model is $[\Phi_{rm}(t_0) \quad p_{B,rm}(t_0)] = \left[0° \quad 0\frac{°}{s}\right]$. The initial condition of the plant states is $[\Phi(t_0) \quad p_B(t_0)] = \left[28.65° \quad 0\frac{°}{s}\right]$.

The uncertainty quantification approach of section 4.4.3 is applied in order to form a belief on $\Delta(\Phi)$ using online gathered data. Therefore, an OFPS is chosen to estimate the state derivative $\hat{\boldsymbol{x}}_{P,k}$. The history stack is limited to $p_H = 100$ data points. A static history stack management is chosen. The squared exponential kernel function is chosen as a basis for the regression. Table H-2 in Appendix H.3 lists all relevant parameters in order to recreate the simulation.

Figure 4-10 shows the evolution of the states (left column) as well as the uncertainty approximation (right column) at the four time instants $0s, 7.5s, 12.5s$ and $40s$. In the beginning, the history stacks are empty. Hence, all information about the uncertainty is compressed in the prior $p(f)$. Since no prior information is incorporated, the GP has a zero mean. Hence, it is seen to badly approximate the uncertainty $\Delta(\Phi)$.

After $5s$ the first step occurs and the data management algorithm starts to store data points based on the kernel linear independence test in (4-93). The second row in Figure 4-10 depicts the state evolution as well as the current approximation of the uncertainty after $7.5s$. It can be seen that the algorithm stored data points evenly distributed within the already visited domain of the state space. This behavior can be attributed to the kernel linear independence test in (4-93), which only stores a candidate point if it sufficiently increases the basis of the GP. It can also be seen that the mean of the posterior GP approximates the uncertainty $\Delta(\Phi)$ with only minor deviation in areas, which contain a sufficient density of data points. Similarly, its standard deviation decreases in the same areas. This can be attributed to the regression and the approximation capability of the GP. Note that areas of the state space, which do not contain data, still resemble the prior GP.

After $10s$, the references signal commands a negative step with an amplitude of $-30°$. The third row in Figure 4-10 depicts the simulation state after $12.5s$. It can be seen that the system tracks the reference model with only minor deviations. In this phase, the algorithm also gathers data points for negative roll angles. Consequently, the mean of the posterior GP approximates the uncertainty with only minor deviations and its standard deviation decreases significantly.

Finally, the last row in Figure 4-10 depicts the situation after $40s$. It can be seen that the system tracked 7 consecutive step commands. When compared to the situation after $12.5s$, only a small number of data points was added to the history stack. This again can be attributed to the kernel linear independence test, which only stores points if they significantly increase the basis of the GP. Since the reference command sequence repeats itself after $20s$, no fundamentally new information is encountered afterwards. In contrast, if the amplitude of the reference command was increased, the algorithm would store data points in the new state space regions.

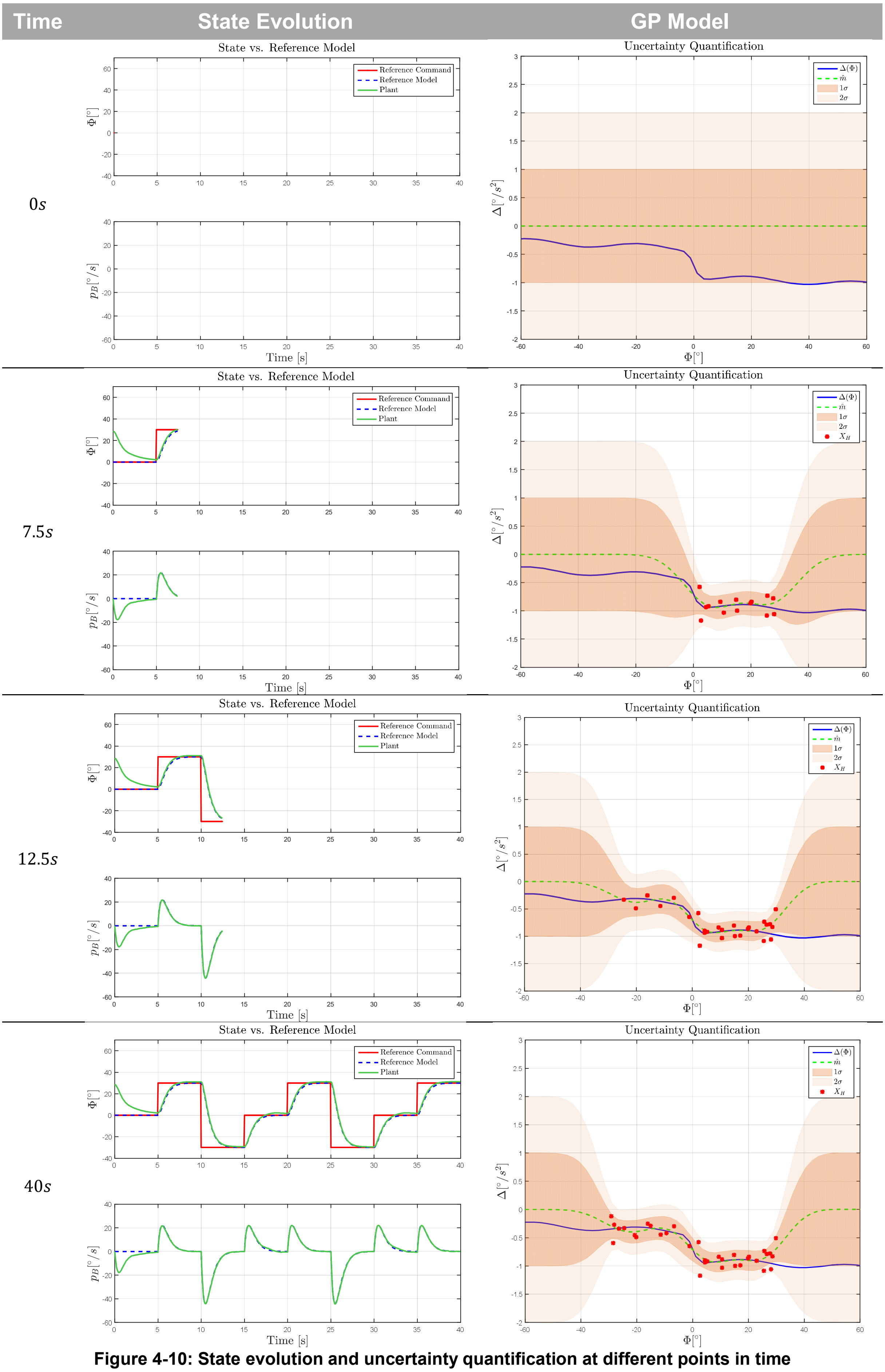


Figure 4-10: State evolution and uncertainty quantification at different points in time

4.5 Indicator Function, Confidence Measure and Monitoring Output

Sections 4.3 and 4.4 showed how to propagate a GMM through a nonlinear system and a GP. Recursive application of this process yields a stochastic forecast of the system states over the prediction horizon. The following chapter relies on the result of the forecast in order to assess the occurrence of a requirement violation and the confidence the monitoring algorithm has in its prediction. Therefore, section 4.5.1 develops an indicator function for requirement violation within the prediction horizon. Secondly, section 4.5.2 introduces a confidence measure, which assesses the propagated approximation error. Finally, section 4.5.3 shows how to integrate both results to calculate a single monitoring output, which reflects both, the result of the prediction as well as the monitor's confidence in it.

4.5.1 Indicator Function for Requirement Violation

Let $\hat{p}\left(\bar{\boldsymbol{x}}_{s,k+\Delta k}\right) = \sum_{i=1}^{m} \alpha_i N\left(\boldsymbol{\mu}_{k+\Delta k,i}, \boldsymbol{\Sigma}_{k+\Delta k,i}\right)$ denote the approximated state distribution of $\bar{\boldsymbol{x}}_{s,k+\Delta k}$ after Δk forecast steps. Assume that $\bar{\boldsymbol{x}}_{req}$ denotes a state requirement on $\bar{\boldsymbol{x}}_s$. The question arises under which conditions the monitoring algorithm reports an imminent requirement violation. These conditions in turn are highly dependent on the application. For example, it might be desirable to only flag requirement violations for a certain system state if they occur over more than one prediction time step. In contrast, for a different state it might be desirable to report an imminent requirement violation the second it occurs in the prediction. Consequently, consider the evaluation of the monitoring output to be given by an indicator function f_{mon}:

$$f_{mon} := \begin{cases} 1 & \text{if } h_{ind} > \delta_{ind} \\ 0 & \text{if } h_{ind} \leq \delta_{ind} \end{cases} \tag{4-98}$$

The monitoring function is comprised of a test function h_{ind} and a threshold δ_{ind}. The underlying idea is to evaluate the number of requirement violations occurring during the prediction horizon using the test function h_{ind}. If the latter exceeds a predefined threshold δ_{ind}, the monitor raises a warning. Consider the following test function candidate:

$$h_{ind} = \sum_{\Delta k=\alpha_{ind}}^{\beta_{ind}} g_{ind}\left(\hat{p}\left(\bar{\boldsymbol{x}}_{s,k+\Delta k}\right), \bar{\boldsymbol{x}}_{req}\right) \tag{4-99}$$

Here, $g_{ind}(\cdot)$ represents an indicator function, which determines whether a requirement violation occurred at the single time step $k + \Delta k$, where $\Delta k \in [\alpha_{ind}, \beta_{ind}]$. In some situations it might be desirable to consider only a subset of the prediction horizon. Here, α_{ind} and β_{ind} represent the first and last time step defining this subset within the prediction horizon. As an indicator function $g_{ind}(\cdot)$, this thesis checks whether the probability $P_\zeta\left(\bar{\boldsymbol{x}}_{s,k+\Delta k} > \bar{\boldsymbol{x}}_{req}\right)$ of exceeding the state requirement $\bar{\boldsymbol{x}}_{req}$ exceeds a predefined limit δ_ζ:

$$g_{ind}\left(\hat{p}\left(\bar{\boldsymbol{x}}_{s,k+\Delta k}\right), \bar{\boldsymbol{x}}_{req}\right) := \begin{cases} 1 & \text{if } P_\zeta\left(\bar{\boldsymbol{x}}_{s,k+\Delta k} > \bar{\boldsymbol{x}}_{req}\right) \geq \delta_\zeta \\ 0 & \text{if } P_\zeta\left(\bar{\boldsymbol{x}}_{s,k+\Delta k} > \bar{\boldsymbol{x}}_{req}\right) < \delta_\zeta \end{cases} \tag{4-100}$$

In the presented case, the probability of a requirement violation is formulated with respect to an absolute and constant state requirement $\bar{\boldsymbol{x}}_{req}$. This case can easily be extended to more complex structures such as algebraic functions $f\left(\bar{\boldsymbol{x}}_{req}, \Delta k\right)$. E.g. Bierling in [17] uses relative state requirements, which tighten over time. In any case, the probability $P_\zeta\left(\overline{\boldsymbol{x}}_{s,k+\Delta k} > \overline{\boldsymbol{x}}_{req}\right)$ is calculated by evaluating the cdf of $\overline{\boldsymbol{x}}_{s,k+\Delta k}$:

$$P_\zeta\left(\overline{\boldsymbol{x}}_{s,k+i} > \overline{\boldsymbol{x}}_{req}\right) = 1 - P_\zeta\left(\overline{\boldsymbol{x}}_{s,k+\Delta k} \leq \overline{\boldsymbol{x}}_{req}\right) = 1 - \int_{-\infty}^{\overline{\boldsymbol{x}}_{req}} \hat{p}\left(\overline{\boldsymbol{x}}_{s,k+i}\right) d\overline{\boldsymbol{x}}_{s,k+i} \tag{4-101}$$

Inserting the equation for the GMM in (4-5) as well as swapping the sum and the integral yields

$$\begin{aligned} P_\zeta\left(\overline{\boldsymbol{x}}_{s,k+i} > \overline{\boldsymbol{x}}_{req}\right) &= 1 - \int_{-\infty}^{\overline{\boldsymbol{x}}_{req}} \sum_{i=1}^{m} \alpha_i N\left(\boldsymbol{\mu}_{k+\Delta k,i}, \boldsymbol{\Sigma}_{k+\Delta k,i}\right) d\overline{\boldsymbol{x}}_{s,k+i} \\ &= 1 - \sum_{i=1}^{m} \alpha_i \int_{-\infty}^{\overline{\boldsymbol{x}}_{req}} N\left(\boldsymbol{\mu}_{k+\Delta k,i}, \boldsymbol{\Sigma}_{k+\Delta k,i}\right) d\overline{\boldsymbol{x}}_{s,k+i}. \end{aligned} \tag{4-102}$$

Evaluating the probability $P_\zeta\left(\overline{\boldsymbol{x}}_{s,k+i} > \overline{\boldsymbol{x}}_{req}\right)$ of a requirement violation reduces to the problem of evaluating the weighted sum of the cdfs of the Gaussian mixtures. Using equations (4-98)-(4-102), different monitoring strategies are imaginable. Therefore, let k_{end} denote the length of the prediction horizon. Table 4-1 lists four parameter combinations and the four resulting monitoring strategies. The threshold δ_ζ is problem specific and therefore not included in the scenarios of Table 4-1. The mentioned scenarios represent the most important parameter sets in the view of the author. The depicted list is not meant to be exhaustive and is easily extendable. Note, that it also might be desirable to solely supply the probability in (4-102) instead of a binary indicator.

Parameter Set	Strategy Description
$\alpha_{ind} = 0, \beta_{ind} = k_{end}$ $\delta_{ind} = 0$	The monitor indicates a requirement violation if the requirement limit is exceeded at least once during the complete prediction horizon.
$\alpha_{ind} = 0, \beta_{ind} = k_{end}$ $\delta_{ind} > 0$	The monitor indicates a requirement violation if the requirement limit is exceeded at last $\delta_{ind} + 1$ times during the complete prediction horizon.
$\alpha_{ind} = 0, \beta_{ind} < k_{end}$ $\delta_{ind} = 0$	The monitor indicates a requirement violation if the requirement is exceeded at least once during the time frame $\Delta k \in [0, \beta_{ind}]$. As a consequence, the predicted state distribution is allowed to violate the requirements in the horizon $\beta_{ind} < \Delta k \leq k_{end}$
$\alpha_{ind} > 0, \beta_{ind} = k_{end}$ $\delta_{ind} = 0$	The monitor indicates a requirement violation if the requirement is exceeded at least once during the time frame $\Delta k \in [\alpha_{ind}, k_{end}]$. As a consequence, the predicted state distribution is allowed to violate the requirements in the horizon $0 \leq \Delta k < \alpha_{ind}$

Table 4-1: Strategies for the parameter selection of the indicator function

4.5.2 Confidence in the Monitor Prediction Quality

The prediction and therefore the result of the monitoring algorithm depend on a series of models and approximations of the real system. Consequently, the forecast is subject to errors, which in turn are propagated through the dynamical model. The sole reliance on the indication of the monitor neglects the approximation errors. Colloquially speaking, the monitor is given full trust. The indicator function in (4-98) intrinsically forms a binary classifier. As a result of the propagated errors, the false positive and false negative values might increase, thus provoking a reaction of the operator or control system, which is not desirable.

For the proposed monitoring technique, three main sources of propagation errors exist. First, the initial state distribution is approximated by a GMM in (4-5). While the approximation improves with the number of mixture components, an approximation error remains. Secondly, as shown in section 4.3, the GMM is propagated through a surrogate model, consisting of piecewise linear functions. The latter originate from a first order Taylor series expansion at the mean of each mixture element. Still, the linearization may not be valid over most of the support of the respective mixture, hence introducing errors in the propagation. Lastly, in section 4.4.3, the unknown uncertainty is assumed to be modelled by a GP. In reality this assumption might not hold.

While the first two error sources can be assessed at each prediction step, the assumption on the representability of the uncertainty by a GP requires extensive prior analysis and exceeds the scope of this thesis. Therefore, it only concentrates on the propagation error resulting from the approximation of the state distribution by a GMM as well as its propagation through the nonlinear dynamics.

The basis of the prediction step is the Chapman-Kolmogorov equation for the real state pdf $p(\boldsymbol{x}_{s,k+1})$ in (4-2). Expanding it by adding and subtracting the estimated predicted state distribution $\hat{p}(\boldsymbol{x}_{s,k+1})$ and using (4-33) yields

$$\begin{aligned} p(\boldsymbol{x}_{s,k+1}) &= \int p(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})p(\boldsymbol{x}_{s,k})d\boldsymbol{x}_{s,k} \\ &+ \int \hat{p}(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})\hat{p}(\boldsymbol{x}_{s,k})d\boldsymbol{x}_{s,k} - \int \hat{p}(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})\hat{p}(\boldsymbol{x}_{s,k})d\boldsymbol{x}_{s,k}. \end{aligned} \tag{4-103}$$

Rearranging terms yields

$$\begin{aligned} p(\boldsymbol{x}_{s,k+1}) &= \int \hat{p}(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})\hat{p}(\boldsymbol{x}_{s,k})d\boldsymbol{x}_{s,k} \\ &+ \int p(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})p(\boldsymbol{x}_{s,k}) - \hat{p}(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})\hat{p}(\boldsymbol{x}_{s,k})d\boldsymbol{x}_{s,k}. \end{aligned} \tag{4-104}$$

The approximation $\hat{p}(\boldsymbol{x}_{s,k})$ of the initial state distribution $p(\boldsymbol{x}_{s,k})$ results in an approximation error $\eta_{GMM}(\boldsymbol{x}_{s,k})$ such that

$$p(\boldsymbol{x}_{s,k}) = \hat{p}(\boldsymbol{x}_{s,k}) + \eta_{GMM}(\boldsymbol{x}_{s,k}). \tag{4-105}$$

By using (4-105), (4-104) becomes

$$
\begin{aligned}
p(\boldsymbol{x}_{s,k+1}) &= \int \hat{p}(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})\hat{p}(\boldsymbol{x}_{s,k})d\boldsymbol{x}_{s,k} \\
&+ \int p(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})[\hat{p}(\boldsymbol{x}_{s,k}) + \eta_{GMM}(\boldsymbol{x}_{s,k})] \; - \hat{p}(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})\hat{p}(\boldsymbol{x}_{s,k})d\boldsymbol{x}_{s,k} \\
&= \int \hat{p}(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})\hat{p}(\boldsymbol{x}_{s,k})d\boldsymbol{x}_{s,k} \\
&+ \underbrace{\int [p(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k}) - \hat{p}(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})]\hat{p}(\boldsymbol{x}_{s,k})d\boldsymbol{x}_{s,k}}_{\Delta p_1(x_{s,k})} + \underbrace{\int p(\boldsymbol{x}_{s,k+1}|\boldsymbol{x}_{s,k})\eta_{GMM}(\boldsymbol{x}_{s,k})\, d\boldsymbol{x}_{s,k}}_{\Delta p_2(x_{s,k})}
\end{aligned}
\qquad \textbf{(4-106)}
$$

The terms $\Delta p_1(\boldsymbol{x}_{s,k})$ and $\Delta p_2(\boldsymbol{x}_{s,k})$ represent the propagated error resulting from approximation inaccuracy. In particular, $\Delta p_1(\boldsymbol{x}_{s,k})$ represents the error resulting from the first-order Taylor series expansion of the nonlinear dynamics. In turn, $\Delta p_2(\boldsymbol{x}_{s,k})$ represents the error resulting from the GMM approximation of the initial state distribution $p(\boldsymbol{x}_{s,k})$. If both, $\Delta p_1(\boldsymbol{x}_{s,k})$ and $\Delta p_2(\boldsymbol{x}_{s,k})$, vanish, then the real distribution forecast is obtained.

One way to decrease the propagation error and increase the quality of the monitoring algorithm is to enhance the precision of its modules. In fact, Terejanu et al. argues in [206] that as the limit of the mixtures' covariance approaches zero, the error terms $\Delta p_1(\boldsymbol{x}_{s,k}), \Delta p_2(\boldsymbol{x}_{s,k})$ tend to zero. Following the neural network universal approximation theorem, this requires an increase in the number of mixture elements. Therefore, possible solutions to increase the approximation property include the increase of the number of mixture weights or allowing for an update of the mixture weights α_i at each prediction step. However, an increase in precision usually increases the complexity and the demand in computational resources. Instead, this thesis takes a different route by introducing a confidence measure for the prediction. The underlying idea is to assess the quality of the prediction steps and include the results in the indication given by the monitor. Colloquially, the goal is an assessment of how far the monitoring output can be trusted.

To achieve this, this thesis proposes the use of the *Overlapping Coefficient* (OVL; [99], see Appendix B.9). In essence, the OVL is a mathematical operation, which compares two distributions and returns the amount of shared area. Figure 4-11 shows an exemplary application of the OVL. Figure 4-11 (left) shows the shared area of two normal distributions with equal standard deviation but shifted mean. The OVL and therefore the shared area amounts to 62% in this example. The OVL can be applied to arbitrary distribution pairs. Figure 4-11 (right) shows the shared area and the OVL of a normal distribution with respect to $p(\tan^{-1}(x))$, given that x follows the standard normal distribution.

The underlying idea of the application of the OVL is to assess the shared area between the real forecast $p(\boldsymbol{x}_{s,k+1})$ and the approximated one $\hat{p}(\boldsymbol{x}_{s,k+1})$. Colloquially speaking, the OVL can then be interpreted as an estimate about how much of the distribution is propagated correctly. The higher the score of the OVL, the more confidence can be put into the result of the forecast. Often, the Kullback-Leibler Divergence is used in order to compare the similarity between two distributions. However, the latter is unbounded and hard to interpret. Due to these properties, this thesis uses the OVL instead.

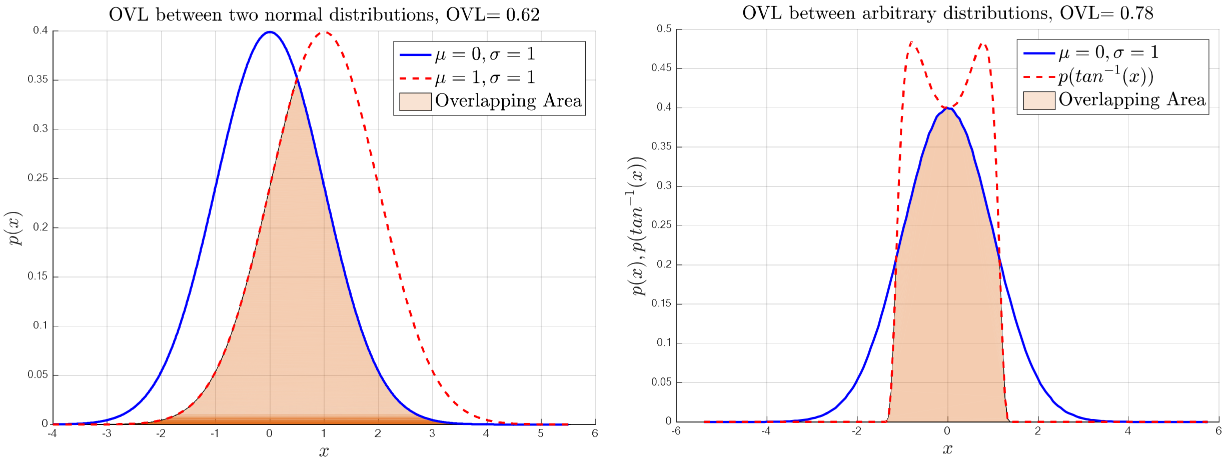


Figure 4-11: Overlapping Coefficient between two normal distributions (left) and two arbitrary distributions (right)

In the case of the presented GMM, the OVL is calculated for each mixture element:

$$\mathrm{OVL}\left(p(\boldsymbol{x}_{s,k+1}),\hat{p}(\boldsymbol{x}_{s,k+1})\right)=\sum_{i=1}^{m}\alpha_i\mathrm{OVL}\left(p_i(\boldsymbol{x}_{s,k+1}),N(\boldsymbol{\mu}_{i,k+1},\boldsymbol{\Sigma}_{i,k+1})\right). \qquad \textbf{(4-107)}$$

Here, $p_i(\boldsymbol{x}_{s,k})$ denotes the probability distribution for propagating a normally distributed random variable, which inherits the statistics of the i-th mixture element, through the nonlinear dynamics. It therefore, resembles the true posterior, which is formed by propagating the i-th mixture element through the nonlinear dynamics.

Solving (4-107) requires knowledge about the true posterior distributions $p_i(\boldsymbol{x}_{s,k})$. Since $p_i(\boldsymbol{x}_{s,k})$ is almost surely non-normal, calculating the respective distribution exactly is often not possible. Instead, sampling based approaches offer a way to estimate $p_i(\boldsymbol{x}_{s,k})$. The underlying idea is to generate a large number of samples from the prior distribution, propagate them through the nonlinear dynamics and analyze the result. While this approach yields good results in practice, it is computationally expensive. However, note that the nonlinear system dynamics are assumed to be time invariant. This indicates that the propagation of a normal distribution through the nonlinear dynamics solely relies on the first and second moment of the prior. Hence, this thesis pre-calculates the OVL for the expected range of the first and second moment and stores the result in a lookup-table. The online algorithm reduces to a query in the lookup table for each mixture element. Figure H-2 in Appendix H.4 applies the OVL to the functions of Example 4-4 and therefore exemplarily demonstrates how such a lookup table can be constructed.

During the forecast, the OVL not only depends on the current prediction step, but also on all past prediction steps. Hence, the OVL for the $\Delta\mathrm{k}$ –th prediction step becomes:

$$\mathrm{OVL}\left(p(\boldsymbol{x}_{s,k+\Delta\mathrm{k}}),\hat{p}(\boldsymbol{x}_{s,k+\Delta\mathrm{k}})\right)=\prod_{i=0}^{\Delta k}\mathrm{OVL}\left(p(\boldsymbol{x}_{s,k+\mathrm{i}}),\hat{p}(\boldsymbol{x}_{s,k+i})\right). \qquad \textbf{(4-108)}$$

Note, that (4-108) also depends on the difference between the initial distribution $p(\boldsymbol{x}_{s,k})$ and its initial approximation $\hat{p}(\boldsymbol{x}_{s,k})$ by a GMM. Hence, (4-108) inherently deals with both, errors due to the approximation of the nonlinear dynamics and the approximation error of the initial state distribution by a GMM.

4.5.3 Monitoring Output

Section 4.5.1 introduced an indicator function for potential imminent requirement violation based on the prediction of the state trajectories. Section 4.5.2 introduced a measure, which assesses the monitor's confidence into its indication. Alone, each quantity only presents a part of the big picture. If for example, the confidence measure is neglected, the output can raise a warning even though the quality of the forecast is degraded and doesn't represent the reality sufficiently well. In turn, the confidence measure only relays information about the prediction quality, not about the probability of a state requirement violation.

A logical conclusion is to supply the probability of a state requirement violation in (4-102) instead of a binary value (4-98), as well as the confidence of (4-108). Together they yield a bigger picture and allow the operator to assess the current situation. While this might be true for extreme cases, such as a high probability in the indicator and a high confidence (or vice versa), the assessment becomes more complicated if both quantities reside in medium ranges. One such example is that the indicator function indicates a 40% chance of requirement violation, while the confidence is at 60%. Deriving suitable actions from this circumstance is not straight forward.

Instead of assessing the indicator probability of (4-102) and the confidence separately, a different approach includes the combination of both quantities into a single modified probability. Therefore, let the monitoring output be given by the modified probability $P_{mod}\left(\overline{\boldsymbol{x}}_{s,k+i} \leq \overline{\boldsymbol{x}}_{req}\right)$ of abiding all state requirements:

$$P_{mod}\left(\overline{\boldsymbol{x}}_{s,k+i} \leq \overline{\boldsymbol{x}}_{req}\right) = \left(1 - P_{\zeta}\left(\overline{\boldsymbol{x}}_{s,k+\Delta k} > \overline{\boldsymbol{x}}_{req}\right)\right) \mathrm{OVL}\left(p\left(\overline{\boldsymbol{x}}_{s,k+\Delta \mathrm{k}}\right), \hat{p}\left(\overline{\boldsymbol{x}}_{s,k+\Delta \mathrm{k}}\right)\right) \tag{4-109}$$

Note that the indicator function supplies the probability of a requirement violation, i.e. $P\left(\overline{\boldsymbol{x}}_{s,k+i} > \overline{\boldsymbol{x}}_{req}\right)$. Since it simplifies the calculation, here the probability of state requirement abidance is used. They are connected by $P_{\zeta}\left(\overline{\boldsymbol{x}}_{s,k+\Delta k} \leq \overline{\boldsymbol{x}}_{req}\right) = 1 - P_{\zeta}\left(\overline{\boldsymbol{x}}_{s,k+\Delta k} > \overline{\boldsymbol{x}}_{req}\right)$. In the above mentioned example, $P_{mod}\left(\overline{\boldsymbol{x}}_{s,k+i} \leq \overline{\boldsymbol{x}}_{req}\right)$ would assess the requirement abidance with 36%, which is easier to interpret than the single quantities. $P_{mod}\left(\overline{\boldsymbol{x}}_{s,k+i} \leq \overline{\boldsymbol{x}}_{req}\right)$ can then be integrated into the indication strategy of (4-98) to again yield a binary output.

The proposed modified probability for requirement violation is more conservative than the individual quantities. In fact, if the prediction is completely certain that no requirement violation will occur, but the confidence is low, then also the modified probability will be low and vice versa. While this might deteriorate the achieved prediction performance, in the author's opinion, it is desirable to act cautious if either quantity is low.

4.6 Implementation Aspects

The following section is intended to highlight the, in the view of the author, most important implementation aspects. Thereby, a special focus is on approaches that decrease the computational complexity of the algorithm. Note that the list is not intended to be exhaustive. As a basis, Table 4-2 depicts a sample application of the monitoring algorithm. The latter assumes direct MRAC with online GP regression.

```
while new measurements are available do:
    Given a new measurement $\boldsymbol{x}_{s,k}$, calculate $\hat{\boldsymbol{x}}_{s,k}, \dot{\hat{\boldsymbol{x}}}_{s,k}$ by section 4.4.3.1
    Calculate $\gamma$ by (4-93)
    if $\gamma > \epsilon_{dp}$ then
        Calculate $\hat{\boldsymbol{\Delta}}(\hat{\boldsymbol{x}}_{s,k})$ by (4-92)
        if $p_c < p_H$ then
            store $\left(\hat{\boldsymbol{x}}_{s,k}, \hat{\boldsymbol{\Delta}}(\hat{\boldsymbol{x}}_{P,k})\right)$ in $(\boldsymbol{X}_H, \boldsymbol{Y}_H)$
        elseif $p_c == p_H$
            use data management in section 4.4.3.3 to store or discard $\left(\hat{\boldsymbol{x}}_{P,k}, \hat{\boldsymbol{\Delta}}(\hat{\boldsymbol{x}}_{P,k})\right)$
    Calculate $\hat{p}(\boldsymbol{x}_{s,k})$ as of (4-5) by (2-5)
    for $\Delta k = 1$ to $k_{end}$ do:
        Calculate the linearized system dynamics by (4-17)
        Calculate $\hat{p}(\boldsymbol{x}_{s,k+\Delta k})$ by (4-91)
        Calculate algorithm confidence by (4-108)
        Calculate $P_{mon}(\bar{\boldsymbol{x}}_{s,k+i} \leq \bar{\boldsymbol{x}}_{req})$ by
        Calculate $P_{mod}(\bar{x}_{s,k+i} \leq \bar{x}_{req})$ by (4-109)
        Use $P_{mod}(\bar{x}_{s,k+i} > \bar{x}_{req}) = 1 - P_{mod}(\bar{x}_{s,k+i} \leq \bar{x}_{req})$ to calculate $f_{mon}$ by (4-98)
        If $f_{mon} == 1$ then
            Issue warning to the operator
            Issue integrity value and requirement violation probability to the operator
```

Table 4-2: The proposed monitoring algorithm exemplarily depicted for direct MRAC

Approximating the initial state distribution $\hat{p}(\boldsymbol{x}_{s,k})$ by a GMM requires the solution of the optimization problem in (2-5). Optimization is computationally expensive and hence should be avoided during the runtime of the algorithm. In general, the characteristics of the employed sensor do not change. Hence, the solution to (2-5) is calculated prior to algorithm deployment and only the mean is shifted based on the current measurement. However, if the need of changing the characteristics occur, the optimization problem does not necessarily be computed again. Rather, the GMM can be scaled by a measure the new pdf scales compared to the original one. However, the user has to ensure that the quality of the representation is still sufficient. In particular, upscaling leads to poor representability and requires a new soluation to the optimization problem.

Intuitively, a user would select the frequency of the monitor equal to the one of controller. This originates from the reasoning that choosing smaller time steps for the monitoring algorithm tends to yield better results. However, this increases the number of iterations the algorithm stays in the prediction loop. Hence, choosing a larger time step reduces computational expense, while only marginally increasing the prediction error. This in turn can be attributed to the effect that the time step has a linearizing effect on the system dynamics.

The GP regression step of (2-53) and (2-54) requires the inversion of the matrix $\left[\boldsymbol{k}_{\boldsymbol{X}_H \boldsymbol{X}_H} + \sigma_n^2 \boldsymbol{I}\right]^{-1}$. While an inversion operation tends to be computationally expensive, a variety of techniques exist, which reduce this cost by leveraging special matrix properties. Note, that the matrix $\boldsymbol{k}_{\boldsymbol{X}_H \boldsymbol{X}_H} + \sigma_n^2 \boldsymbol{I}$ is positive definite. Hence, this thesis uses the Cholesky factorization (see Appendix 0), which decomposes the respective matrix into a lower and an upper triangular matrix. The original inverse is computed by multiplying the inverse of the lower (or upper) decomposition with its inversed and transposed version. Alternatively, instead of inverting a matrix, in [48] Csató introduces a gradient based approach for GP regression.

The integral of the indicator function in (4-102) is not easily calculated. Instead, similarly to the OVL in section 4.5.2, the solution for each mixture can be pre-calculated for the expected mean and covariance domain and stored in a look-up table. Evaluating the latter online is computationally more efficient than calculating the integral directly.

In order to save computational time, the forecast can be halted once the OVL becomes lower than a predefined threshold. The underlying argument is that the OVL is monotonically decreasing and hence the prediction will be highly uncertain once it falls below a certain threshold.

To reduce the required computational power, the prediction does not have to be executed in each controller step. Instead, the prediction horizon can be calculated within a number of controller steps. The computational burden is then divided upon each controller step.

Most flight control computers rely on a CPU as the core processing unit. Recently, GPUs gained extensive interest in industry, as they are capable of processing large amounts of parallel information. Amongst others, they are utilized for the training of deep neural networks or Bitcoin mining. The presented monitoring approach relies on the forecast of multiple Gaussian mixtures. The mixtures could be treated independently and therefore processed on parallelizing hardware, thus effectively reducing computational time.

4.7 Conclusion

Chapter 4 proposed a monitoring approach, which predicts the trajectories of the system and controller states. The underlying goal is to move the assessment of the nonlinear control approach from an a-priori analysis to an online evaluation. This is required, as existing tools and techniques for the analysis of nonlinear controllers are highly conservative as well as not applicable to the current certification framework. To evaluate the controller online, the monitor aims at achieving an online assessment of potential imminent requirement violation. This is achieved by employing a GMM and propagating its mixtures through a surrogate model of the nonlinear dynamics, which in turn is represented by a first order Taylor-series approximation at each mixture mean (see section 4.3). Unknown or neglected dynamics are represented by a GP model, which is either available from prior testing or generated online using GP regression (see section 4.4). At each prediction step, the probability of a requirement violation is assessed. If it exceeds a certain limit, the monitor issues a warning. Since the approach relies on a series of approximation, a confidence measure is introduced, which is intended to track the propagated approximation error during the prediction. Both, the probability of a requirement violation as well as the monitor's confidence can be combined in a single indicator, which encompasses the complete information about the forecast (see section 4.5.3).

Several authors already use GMMs and GPs to propagate uncertainties through nonlinear systems. E.g. in [206], Terejanu showed how to propagate GMMs through nonlinear systems and applied it to filtering problems. Similarly, using GP models for long term prediction has been applied amongst others in Hardy ([85]). In particular, Hardy used GMMs and propagated them through GPs to deal with non-Gaussian distributions. The proposed monitoring approach differs as it extends either approach by leveraging known prior models apart from GP models to incorporate as much knowledge as possible. The novelty of the proposed approach therefore primarily lies within the combination of known techniques and their application to nonlinear control problems.

The presented monitoring approach exhibits two main advantages. First, it overcomes the limiting nature of an offline Lyapunov analysis, which intrinsically aims at finding the worst case scenario and uses the latter as its solution. In contrast, the presented monitoring approach predicts system trajectories dependent on the current system state. Hence, the monitor tries to find a local solution to the prediction and is not interested in the worst case scenario. Furthermore, the latter is acceptable as long as its probability of appearance is low. As a result the monitoring approach is less conservative and admits physically reasonable solutions. Secondly, the monitor is able to handle uncertainties as long as they can be modelled by GPs and inferred from online gathered data. As a result, the system dynamics are allowed to change between operations as their change can be assessed online. Furthermore, the monitor is theoretically able to handle system faults and damage occurring during operation.

However, the proposed monitor also exhibits some disadvantages. First, the algorithm is computationally expensive as it requires the recursive application of the prediction step. Furthermore, even though the confidence measure is intended to track the propagated approximation error, no mathematical guarantees are known to the author. Lastly, the monitoring algorithm does not indicate actions. That is, the monitor will flag an imminent requirement violation, but it will not offer advise on how to counter the violation.

The proposed monitoring algorithm presents an alternative way for the future certification of nonlinear flight control algorithms by moving from the controller assessment from an offline analysis to online verification. In the same time the mentioned disadvantages offer room for further development.

5 Application in Numerical Simulation

This section applies the presented monitoring technique to two simulation examples. The first example features a direct MRAC approach in order to control the roll dynamics of an aircraft subject to wingrock motion. The second example features a PMRAC approach for the short period of a F-16 fighter aircraft. In this section, all plots showing the evolution of states are down-sampled from the simulation frequency in a ratio of $15{:}1$.

5.1 Direct Model Reference Adaptive Control for wing-rock dynamics

This section adopts the simulation example of [41]. Modern highly swept-back, or delta wing, fighter aircraft are susceptible to lightly damped oscillations in the roll angle, known as “wing-rock”. The latter primarily occurs at conditions commonly encountered during landing ([179]). The ability of controlling the aircraft precisely in the presence of wing rock is critical for landing. Let ϕ denote the roll attitude of an aircraft, p_B denote the roll rate and ζ_C the aileron command. Then a model for the roll dynamics of an aircraft subject to wingrock dynamics is ([142]):

$$\begin{aligned} \dot{\phi} &= p_B \\ \dot{p}_B &= L_\zeta \zeta_C - \Delta(\phi, p_B). \end{aligned} \tag{5-1}$$

Here, the uncertainty $\Delta(\phi, p_B)$ describes the wing rock motion:

$$\Delta(\phi, p_B) = \Theta_0^* + \Theta_1^* \phi + \Theta_2^* p_B + \Theta_3^* |\phi| p_B + \Theta_4^* |p_B| p_B + \Theta_5^* \phi^3. \tag{5-2}$$

Note, that the dynamics in (5-1) lack a roll damping term. The latter is included in the uncertainty formulation of (5-2) for the purpose of this simulation example. Note that $\boldsymbol{B}_P = [0 \quad L_\zeta]^T$. The parameters of (5-2) are adopted from Singh et al. ([196]). The control architecture of section 3.2 with the control law (3-11) such that $\zeta_C = \zeta_{rm} + \zeta_{bl} + \zeta_{ad}$ is employed. Therefore consider the following reference model:

$$\begin{bmatrix} \dot{\phi}_{rm} \\ \dot{p}_{rm} \end{bmatrix} = \boldsymbol{A}_{rm} \begin{bmatrix} \phi_{rm} \\ p_{rm} \end{bmatrix} + \boldsymbol{B}_{rm} r. \tag{5-3}$$

The adaptive controller uses the structure of (3-15), where the basis function $\boldsymbol{\varphi}$ is given by the basis of the uncertainty in (5-2):

$$\boldsymbol{\varphi} = [1 \quad \phi \quad p_B \quad |\phi| p_B \quad |p_B| p_B \quad \phi^3]^T. \tag{5-4}$$

The update law of (3-19) uses a σ-modification with $k(\boldsymbol{\Theta}) = -\sigma\boldsymbol{\Theta}$, such that

$$\dot{\boldsymbol{\Theta}} = -\boldsymbol{\Gamma}\boldsymbol{\varphi}\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}\sigma\boldsymbol{\Theta}. \tag{5-5}$$

The simulation runs for a total of $40s$ with a time step of $0.01s$. The exogenous reference input commands an attitude of 57° between seconds 15 and 17 and -57° between seconds 25 to 27, respectively. The plant states are initialized at $\phi(t_0) = 68^\circ$ and $p_B(t_0) = -57^\circ/s$. The reference model states as well as the adaptive weights are initialized at zero. Table 5-1 summarizes the relevant simulation parameters for the plant and the controller as well as the respective initial conditions for a concise overview.

Simulation Parameter	Numerical Value
General Simulation	
t_0	$0s$
dt	$0.01s$
t_{end}	$40s$
Plant	
$\boldsymbol{A}_P$	$\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$
$\boldsymbol{B}_P = \begin{bmatrix} 0 \\ L_\zeta \end{bmatrix}$	$\begin{bmatrix} 0 \\ 3 \end{bmatrix}$
$[\Theta_0^* \quad \Theta_1^* \quad \Theta_2^* \quad \Theta_3^* \quad \Theta_4^* \quad \Theta_5^*]$	$[0.8 \quad 0.2314 \quad 0.6918 \quad -0.6245 \quad 0.0095 \quad 0.0214]$
$[\phi(t_0) \quad p_B(t_0)]$	$\left[68^\circ \quad -57\frac{^\circ}{s}\right]$
Baseline Controller	
$\boldsymbol{A}_{rm}$	$\begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix}$
$\boldsymbol{B}_{rm}$	$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$
$[\phi_{rm}(t_0) \quad p_{rm}(t_0)]$	$\left[0^\circ \quad 0\frac{^\circ}{s}\right]$
$\boldsymbol{K}_e$	$[0.5 \quad 0.4]^T$
Adaptive Controller	
$\boldsymbol{\Gamma}$	$2\,\boldsymbol{I}_{6x6}$
$\boldsymbol{Q} \mid \boldsymbol{P}$	$\boldsymbol{I}_{2x2} \mid \begin{bmatrix} 1.4417 & 0.333 \\ 0.333 & 0.6944 \end{bmatrix}$
σ	0.1
$\boldsymbol{\Theta}(t_0)$	$[0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T$

Table 5-1: List of parameters used in the MRAC simulation example

This chapter is divided into four parts. Section 5.1.1 presents the results of numerical simulation of the plant and the controller. Section 5.1.2 introduces the GMM based forecasting. Section 5.1.3 completes the proposed monitoring approach by applying the GP regression based uncertainty quantification.

5.1.1 Numerical Simulation of Direct MRAC for wing-rock dynamics

This section presents the results of the numerical simulation of the plant and the adaptive controller. Figure 5-1 shows the tracking performance of the closed loop system with adaptive controller. In the absence of the latter, the plant is unstable. In contrast, augmenting the baseline controller with MRAC leads to a tracking performance with only minor tracking error. This can be attributed to the fact that the adaptive element approximates the uncertainty $\Delta(\phi, p_B)$ during the runtime of the simulation and therefore decreases its effect on the plant and error dynamics.

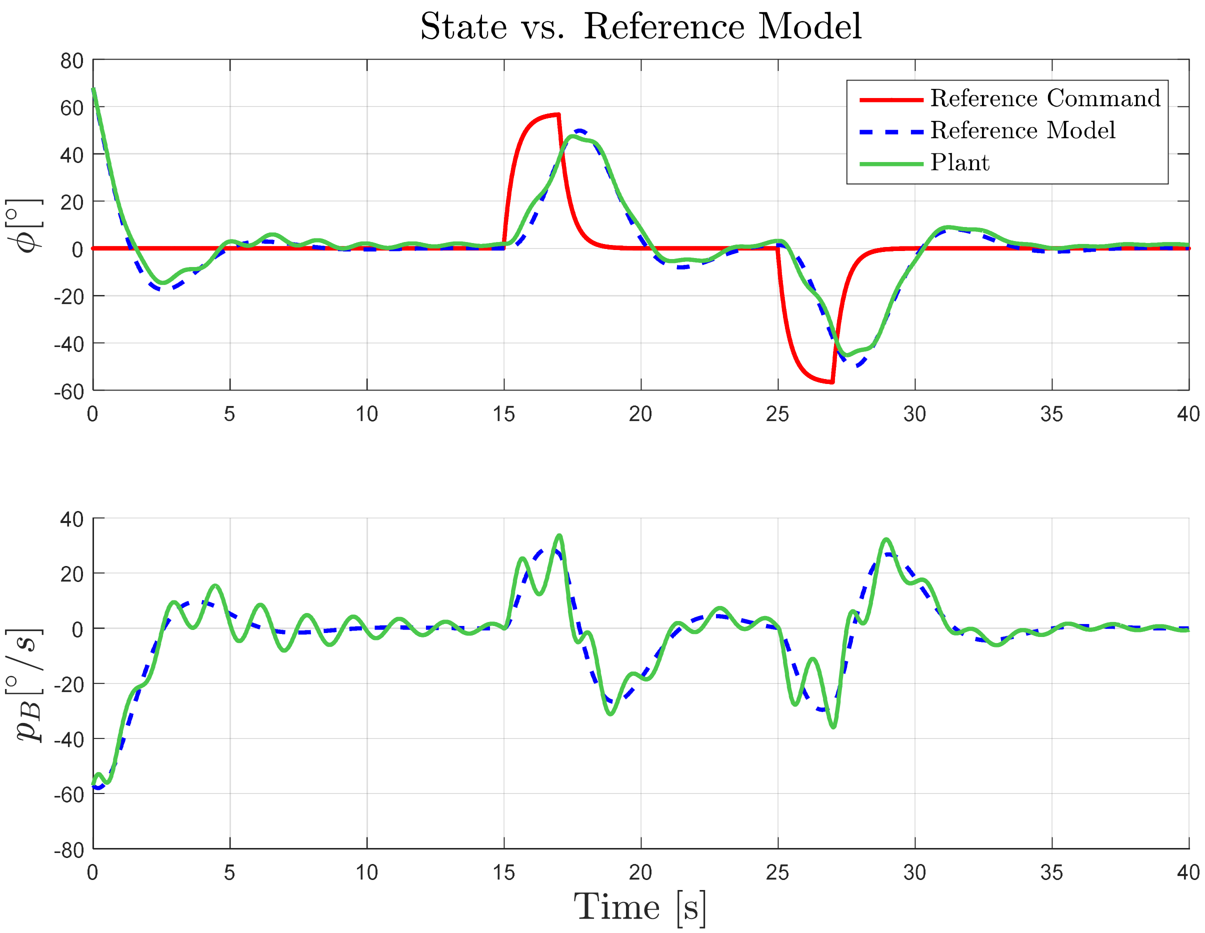


Figure 5-1: Tracking capability of the adaptive controller in the presence of wing-rock motion

Figure 5-2 shows the evolution of the adaptive parameter. It can be seen that the weights do not converge to their optimal values. Parameter convergence is not necessarily guaranteed by direct MRAC. In fact, in order to achieve parameter convergence, different augmentations (such as concurrent learning by Chowdhary [42], [41]) need to be used or Persistency of Excitation of the regressor function $\boldsymbol{\varphi}$ has to be ensured ([23], [155]). While parameter convergence would allow a uniform cancellation of the uncertainty in time, it is not necessary to achieve good tracking performance. In fact, disturbances or unmodelled dynamics would prevent parameter convergence even if such augmentations were used. E.g. the experimental application of concurrent learning in [41] suggests that the parameters converge to a region around their optimal values. As a consequence, in real systems the adaptive parameters will never completely converge to their optimal parameters and therefore elicit the application of a state monitor.

In the next step, the initial conditions of the plant states are independently, normally distributed with $\phi(t_0)\sim N(68°,(2.85°)^2)$ and $p_B(t_0)\sim N\left(-57\frac{°}{s},\left(2.85\frac{°}{s}\right)^2\right)$. The states of the reference model as well as the adaptive weights are still initialized at zero. The simulation is repeated 2500 times.

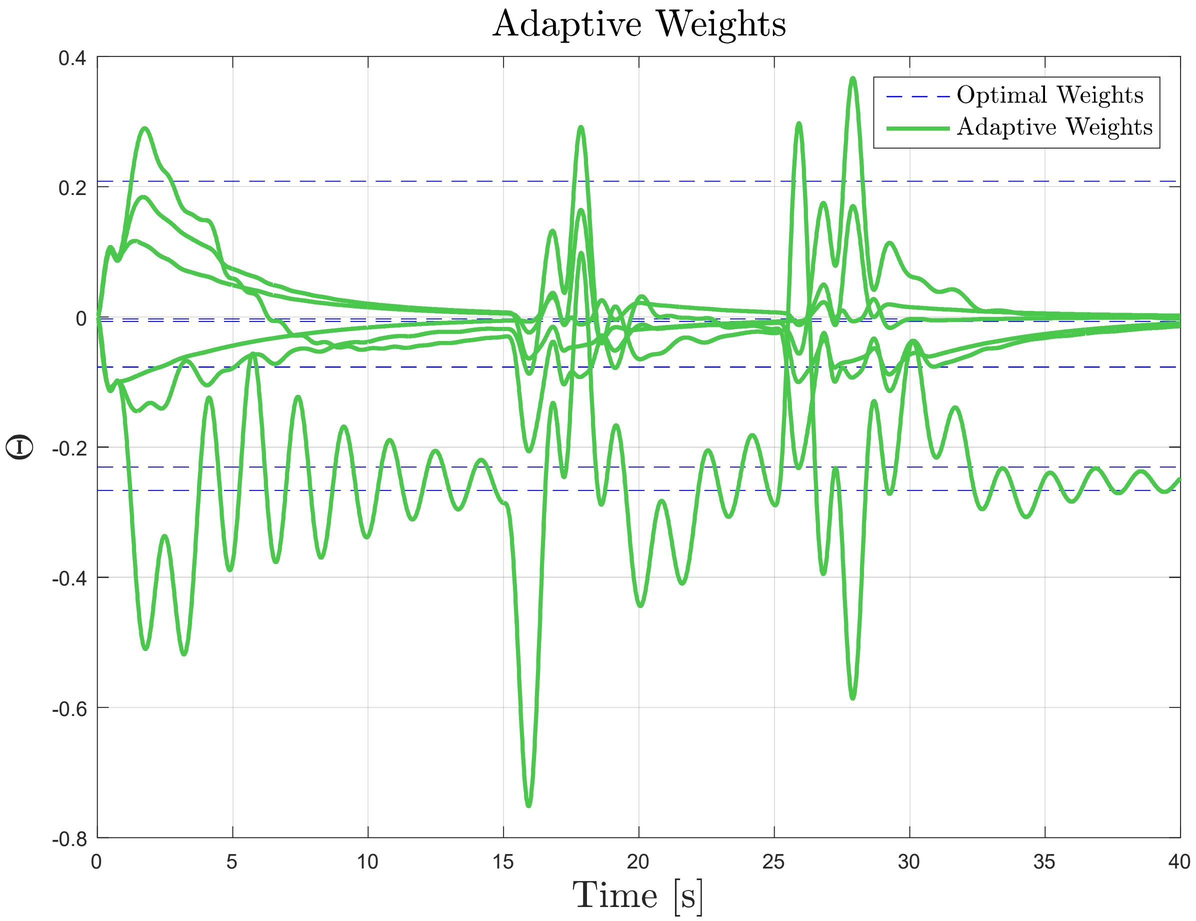


Figure 5-2: Evolution of the adaptive parameters in the presence of wing-rock motion

Figure 5-3 shows the reference command, the state of the reference model as well as the evolution of the state trajectories over the simulation horizon. It can be seen that despite distributed initial conditions, the plant state trajectories stay bounded. This suggests that the adaptive controller is able to stabilize the system and track the reference model under various conditions. Moreover, the distributed state trajectories are seen to converge to their mean trajectory. After 15 seconds the distributed trajectories are nearly indistinguishable from the mean. This can be attributed to the fact that the adaptive controller is able to cancel the uncertainty pointwise in time. This in turn allows the baseline controller to reduce the error between the state trajectories and the reference model. Furthermore, the σ −modification term in the update law decreases the distribution of the adaptive parameter trajectories as it regulates them towards the origin. See Figure H-3 in Appendix H.5 for the evolution of the adaptive parameters. Finally, within the first 15 seconds, the mean state trajectory is seen to oscillate between the upper and lower boundary of the trajectory distribution. This effect is particularly apparent in the evolution of the roll rate. An asymmetrical distribution around the mean suggests that the states are non-normally distributed over the considered simulation horizon.

Figure 5-4 and Figure 5-5 respectively show the roll angle and roll rate trajectory distribution as a histogram at selected time instants, namely after $1s$, $2.8s$, $5.0s$ and at $10.0s$. Furthermore, the respective approximating normal distribution is depicted. The later originates from taking the mean and variance of the state trajectory distribution at the mentioned time instants. It indicates the result if the data is assumed to be normally distributed.

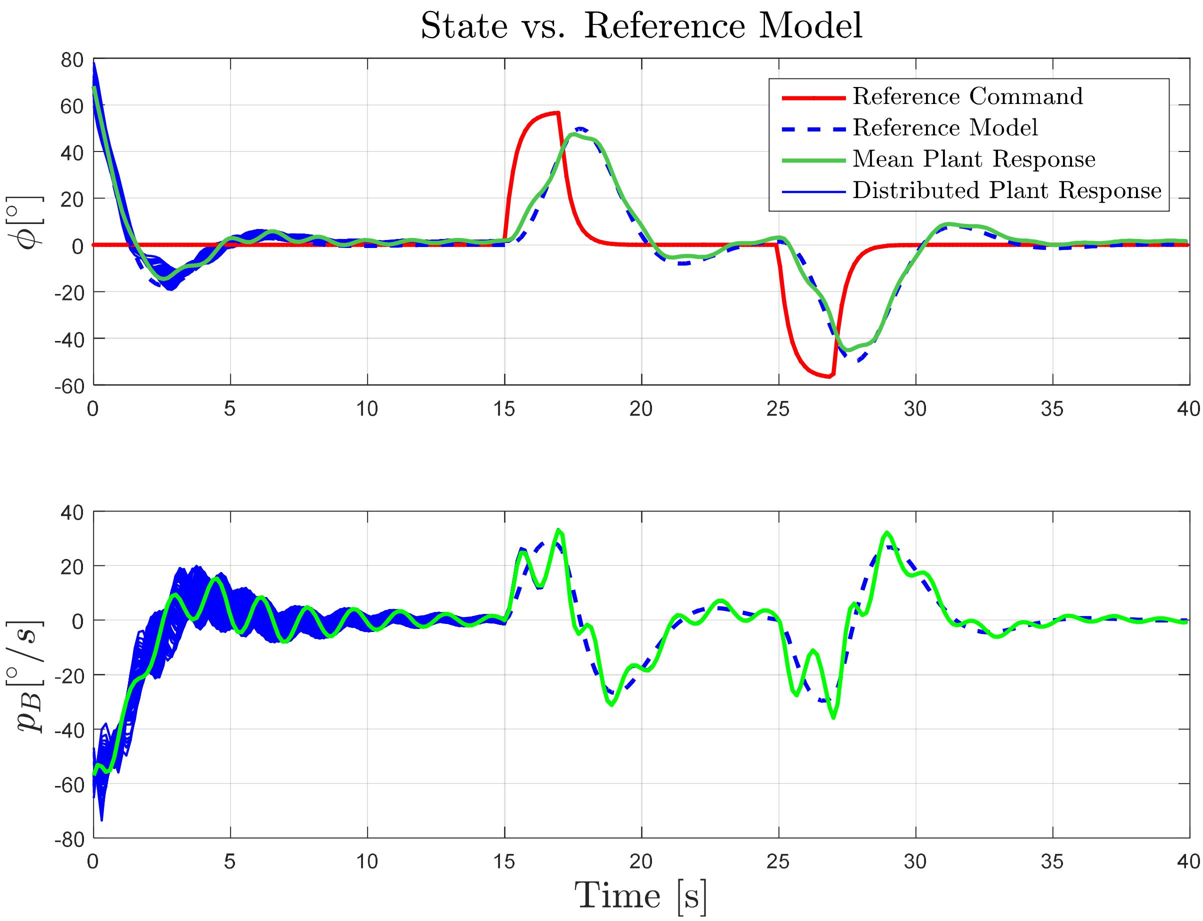


Figure 5-3: Tracking performance of the adaptive controller under wing-rock motion for normally distributed initial conditions

It can be seen that even after $1s$ into the simulation, the roll angle distribution differs from a normal distribution as it starts to become skewed. However, assuming it to be normally distributed results in only minor errors. In contrast, the roll rate is seen to be heavily skewed towards the right as well as truncated at approximately $-37\ \frac{\circ}{s}$. This can be attributed to the nonlinear dynamics of the system. In fact, the nonlinear uncertainty directly affects the roll rate dynamics, while the roll angle dynamics are linear. Due to the linear dynamics, the deterioration from a Gaussian requires time and the approximation of the roll angle distribution by a Gaussian only leads to minor errors.

From $1s$ to $2.8s$ the roll angle distribution becomes skewed to the right, while the roll rate keeps its skewness to the right. In both cases, the state distribution is no longer well approximated by a Gaussian. This indicates that significant errors could be made by neglecting the nonlinear nature of the system during the prediction. After $5s$ into the simulation the roll angle and roll rate distribution become multimodal. Furthermore, the roll rate distribution changes its skewness from the right to the left side. In both cases the support decreases by approximately 50%. This indicates that the single trajectories are converging to the mean trajectory. Finally, after $10s$ both state distributions are multimodal. Furthermore, the support decreases by additional 50%.

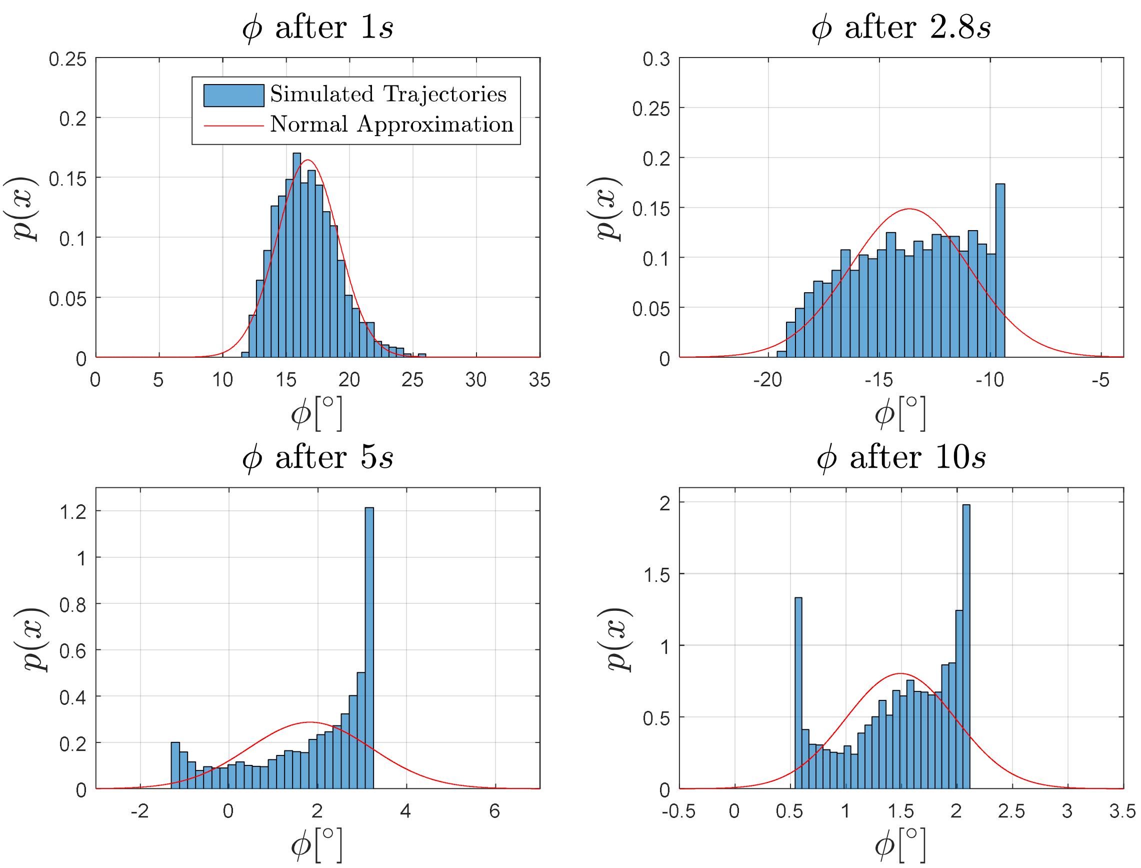


Figure 5-4: Distribution of the roll angle ϕ in the wing-rock example after 1s, 2.8s, 5.0s and 10.0s

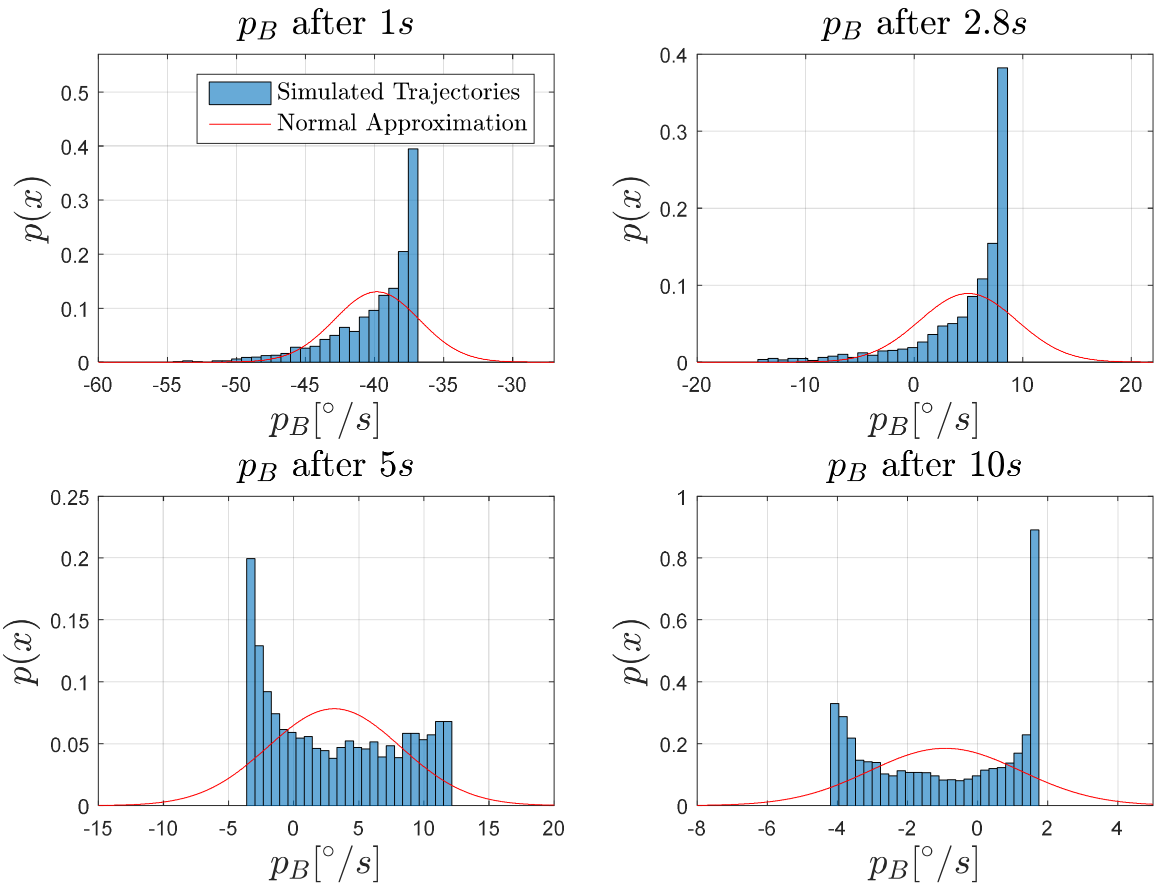


Figure 5-5: Distribution of the roll rate $p_B$in the wing-rock example after 1s, 2.8s, 5.0s and 10.0s

5.1.2 Numerical Simulation with known nonlinearity

This section applies the runtime monitoring approach of section 4 to the simulated system. In particular this section focusses on the propagation of the GMM through the nonlinear dynamics as proposed in section 4.3. Therefore, the nonlinerity is assumed to be exactly known. This assumption will be lifted in section 5.1.3.

In order to propagate the GMM, first the linearized system dynamics need to be derived. For consistency, consider the right hand side of the dynamics in (5-1) and (5-2) to be given in the form of (4-6) and (4-11):

$$\begin{gathered} f_{\phi} = p_B \\ f_{p_B} = L_{\zeta}\boldsymbol{K}_{\boldsymbol{e}}^T\boldsymbol{e} + L_{\zeta}\boldsymbol{K}_{rm,x}^T[\phi_{rm} \quad p_{rm}]^T + L_{\zeta}\boldsymbol{K}_{rm,r}^T r + L_{\zeta}\zeta_{ad} - \Delta(\phi, p_B). \\ \boldsymbol{f}_{\boldsymbol{\Theta}} = -\boldsymbol{\Gamma}\boldsymbol{\varphi}\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}\sigma\boldsymbol{\Theta} \end{gathered} \tag{5-6}$$

Following the definitions and derivation in section 4.3, evaluating $\boldsymbol{f}(\phi_l, p_{B,l}, \phi_{rm}, p_{rm}, \boldsymbol{\Theta}_l)$ at the linearization point $\phi = \phi_l$, $p_B = p_{B,l}$ and $\boldsymbol{\Theta}$=$\boldsymbol{\Theta}_l$ gives

$$\boldsymbol{f}(\phi_l, p_{B,l}, \phi_{rm}, p_{rm}, \boldsymbol{\Theta}_l) = \begin{bmatrix} \boldsymbol{f}_{\phi,l} \\ \boldsymbol{f}_{p_B,l} \\ \boldsymbol{f}_{\boldsymbol{\Theta},l} \end{bmatrix} = \begin{bmatrix} p_{B,l} \\ L_{\zeta}\zeta_C - \Delta(\phi_l, p_{B,l}) \\ -\boldsymbol{\Gamma}\boldsymbol{\varphi}_l\boldsymbol{e}_l^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}\sigma\boldsymbol{\Theta}_l \end{bmatrix}. \tag{5-7}$$

In order to construct the matrix $\boldsymbol{A}_{lin}$, the partial derivatives of f_{ϕ}, f_{p_B} and $\boldsymbol{f}_{\boldsymbol{\Theta}}$ with respect to the system states f_{ϕ} and f_{p_B} and the controller states $\boldsymbol{\Theta}$ need to be derived. The partial derivatives of f_{ϕ} and f_{p_B} with respect to the system states ϕ and p_B are

$$\left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l} \\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} = \left.\begin{bmatrix} \dfrac{\partial f_{\phi}}{\partial \phi} & \dfrac{\partial f_{\phi}}{\partial p_B} \\ \dfrac{\partial f_{p_B}}{\partial \phi} & \dfrac{\partial f_{p_B}}{\partial p_B} \end{bmatrix}\right|_{\substack{\phi=\phi_l \\ p_B=p_{B,l} \\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{0}_{1x2} \\ L_{\zeta}\boldsymbol{K}_{\boldsymbol{e}}^T + L_{\zeta}\boldsymbol{\Theta}_l^T \left.\dfrac{\partial \boldsymbol{\varphi}(\phi, p_B)}{\partial(\phi, p_B)}\right|_{\substack{\phi=\phi_l \\ p_B=p_{B,l}}} \end{bmatrix}. \tag{5-8}$$

Here, $\boldsymbol{0}_{1x2}$ denotes a row vector with $\boldsymbol{0}_{1x2} \in \mathbb{R}^{1\times 2}$. The partial derivative $\frac{\partial \boldsymbol{\varphi}(\phi, p_B)}{\partial(\phi, p_B)}$ of (5-4) is

$$\left.\begin{bmatrix} \dfrac{\partial \boldsymbol{\varphi}}{\partial \phi} & \dfrac{\partial \boldsymbol{\varphi}}{\partial p_B} \end{bmatrix}\right|_{\substack{\phi=\phi_l \\ p_B=p_{B,l}}} = \begin{bmatrix} 0 & 1 & 0 & \dfrac{\phi_l}{|\phi_l|}p_{B,l} & 0 & 3\phi_l^2 \\ 0 & 0 & 1 & |\phi_l| & 2|p_{B,l}| & 0 \end{bmatrix}^T \tag{5-9}$$

The partial derivatives of f_{ϕ} and f_{p_B} with respect to $\boldsymbol{\Theta}$ are

$$\left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} = \left.\begin{bmatrix} \dfrac{\partial f_{\phi}}{\partial \boldsymbol{\Theta}} \\ \dfrac{\partial f_{p_B}}{\partial \boldsymbol{\Theta}} \end{bmatrix}\right|_{\substack{\phi=\phi_l \\ p_B=p_{B,l}}} = \begin{bmatrix} \boldsymbol{0}_{1x6} \\ L_{\zeta}\boldsymbol{\varphi}^T(\phi_l, p_{B,l}) \end{bmatrix}. \tag{5-10}$$

Similarly, the partial derivative of $\boldsymbol{f}_\Theta$ with respect to the system states ϕ and p_B is

$$\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial(\phi, p_B)}\right|_{\substack{\phi=\phi_l \\ p_B=p_{B,l}}} = -\boldsymbol{\Gamma}\left.\frac{\partial \boldsymbol{\varphi}(\phi, p_B)}{\partial(\phi, p_B)}\right|_{\substack{\phi=\phi_l \\ p_B=p_{B,l}}} \left(\begin{bmatrix}\phi_l & p_{B,l}\end{bmatrix}\boldsymbol{P}\boldsymbol{B}_P\right) - \boldsymbol{\Gamma}\boldsymbol{\varphi}\left(\phi_l, p_{B,l}\right)\boldsymbol{B}_P^T\boldsymbol{P}$$
$$+\boldsymbol{\Gamma}\left.\frac{\partial \boldsymbol{\varphi}(\phi, p_B)}{\partial(\phi, p_B)}\right|_{\substack{\phi=\phi_l \\ p_B=p_{B,l}}} \left(\begin{bmatrix}\phi_{rm} & p_{rm}\end{bmatrix}\boldsymbol{P}\boldsymbol{B}_P\right) \tag{5-11}$$

Finally, the partial derivative of $\boldsymbol{f}_\Theta$ with respect to the adaptive weights $\boldsymbol{\Theta}$ is

$$\left.\frac{\partial \boldsymbol{f}_\Theta}{\partial \boldsymbol{\Theta}}\right|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}_l} = -\boldsymbol{\Gamma}\sigma. \tag{5-12}$$

By constructing $\boldsymbol{A}_{lin}$ as shown in (4-17), the full linearized system dynamics are

$$\begin{bmatrix}\dot{\phi} \\ \dot{p}_B \\ \dot{\boldsymbol{\Theta}}\end{bmatrix} = \boldsymbol{f}\left(\phi_l, p_{B,l}, \phi_{rm}, p_{rm}, \boldsymbol{\Theta}_l\right) + \boldsymbol{A}_{lin}\left(\phi_l, p_{B,l}, \phi_{rm}, p_{rm}, \boldsymbol{\Theta}_l\right)\begin{bmatrix}\phi - \phi_l \\ p_B - p_{B,l} \\ \boldsymbol{\Theta} - \boldsymbol{\Theta}_l\end{bmatrix}. \tag{5-13}$$

The GMM used in this simulation example consists of 99 mixture elements. The initial standard deviation of each mixture element is selected to be $\sigma_{i,0} = 0.03$. The mixture elements are equidistantly distributed in the range of 4σ of the initial state distribution $\phi(t_0) \sim N\left(68°, \left(2.85\frac{°}{s}\right)^2\right)$ and $p_B(t_0) \sim N\left(-57\frac{°}{s}, \left(2.85\frac{°}{s}\right)^2\right)$. By solving the quadratic program in (2-5), the GMM is fitted to the initial state distribution of the simulation.

The prediction horizon is set to $t_{pred} = 39.99s$ with a time step of $\Delta t_{pred} = 0.01s$. This results in $k_{end} = 3999$ prediction steps. Note, that the initial time is not included in the prediction horizon. Hence, predicting for $39.99s$ results in a prediction to the end of the simulation. The GMM is propagated through the linearized dynamics in (5-13) according to the update laws in (4-37) and (4-38). Since the uncertainty is assumed to be known, it is evaluated at each mixture element at each prediction step. Table 5-2 summarizes the parameters used for the GMM based forecast.

Simulation Parameter	Numerical Value
Prediction	
t_{pred}	$39.99s$
dt_{pred}	$0.01s$
GMM	
m (Number of mixture elements)	99
$\sigma_{i,0}$	0.03
Initial distribution of the mixture elements μ_i	Equidistant within 4σ of the initial state distribution

Table 5-2: List of parameters used for the GMM forecast in the wing-rock example

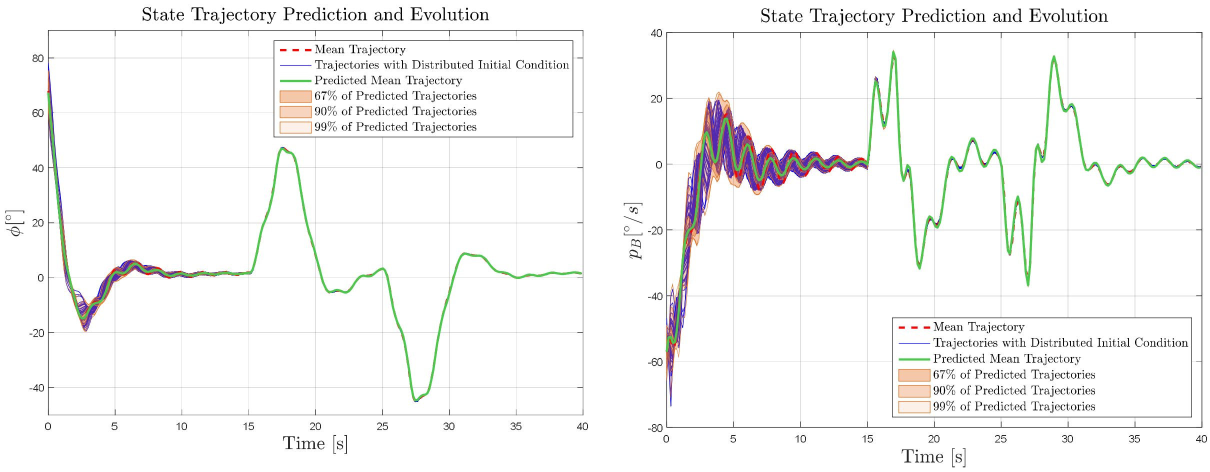


Figure 5-6: Evolution of the predicted system states ϕ and p_B over $40s$ in the wing-rock example

Figure 5-6 shows the predicted mean trajectory as well as the area in which 67%, 90% and 99% of all state trajectories are predicted to reside in at each time instant. Furthermore, the plot depicts the mean trajectory as well as the evolution of the state trajectories for distributed initial conditions. The predicted area significantly shrinks after approximately $15s$. This indicates that the state trajectories converge to their mean, which in turn aligns with the simulation results in section 5.1.1. The underlying reason for this is that the adaptive weights converge to their mean trajectory (see Figure H-4 in Appendix H.5). The adaptive controller is able to cancel the uncertainty pointwise in time, resulting in linear error dynamics. In the absence of further disturbances, the propagation of a Gaussian rv (or mixture element) through stable linear dynamics results in a decrease of the variance. In turn, the area in which the states are predicted to reside in shrinks significantly. Furthermore, after $15s$ both mean trajectories predicted by the GMM are nearly indistinguishable from the simulated ones. This indicates that the proposed GMM forecast algorithm is in fact able to predict the evolution of the states over the simulation horizon.

Figure 5-7 and Figure 5-8 show a highlighted version of the roll angle and roll rate evolution and prediction for the first $15s$ of the simulation. The predicted mean roll angle matches the simulated one with only minor errors. Similarly, the monitoring algorithm is able to predict the mean roll rate with only little error. Between seconds 5 and 10 the prediction is seen to match the frequency of the occurring oscillation, but assuming reduced amplitude. A discrepancy of about $2.5°/s$ occurs. The latter can be explained with the predicted adaptive parameters.

Figure 5-9 shows the evolution as well as the prediction of the adaptive parameters within the first $15s$. It can be seen that after approximately $3s$ significant errors between the prediction as well as the simulation for Θ_3, Θ_4 and Θ_5 occur. The mentioned adaptive parameters correspond to the terms in the regressor vector (5-4), which are dependent on the roll rate p_B. These errors occur in a time frame, where the states are close to zero. In this region, the two uncertainty terms $|\phi|p_B$ and $|p_B|p_B$ assume a plateau. Any mixture element propagated through the uncertainty in this region becomes very sharp, i.e. the posterior variance approaches zero. Since a GMM only approximates continuous functions, it is unable to exactly represent truncated probability distributions. As a result, approximation errors occur, which are then propagated in consecutive state prediction steps, also influencing the seemingly unaffected linear uncertainty term p_B. The wrongly predicted adaptive weights in turn lead to higher damping in the roll rate dynamics, effectively reducing the amplitude in the response. For the sake of completeness, the prediction of the adaptive parameters over the complete simulation horizon is depicted by Figure H-4 in Appendix H.5.

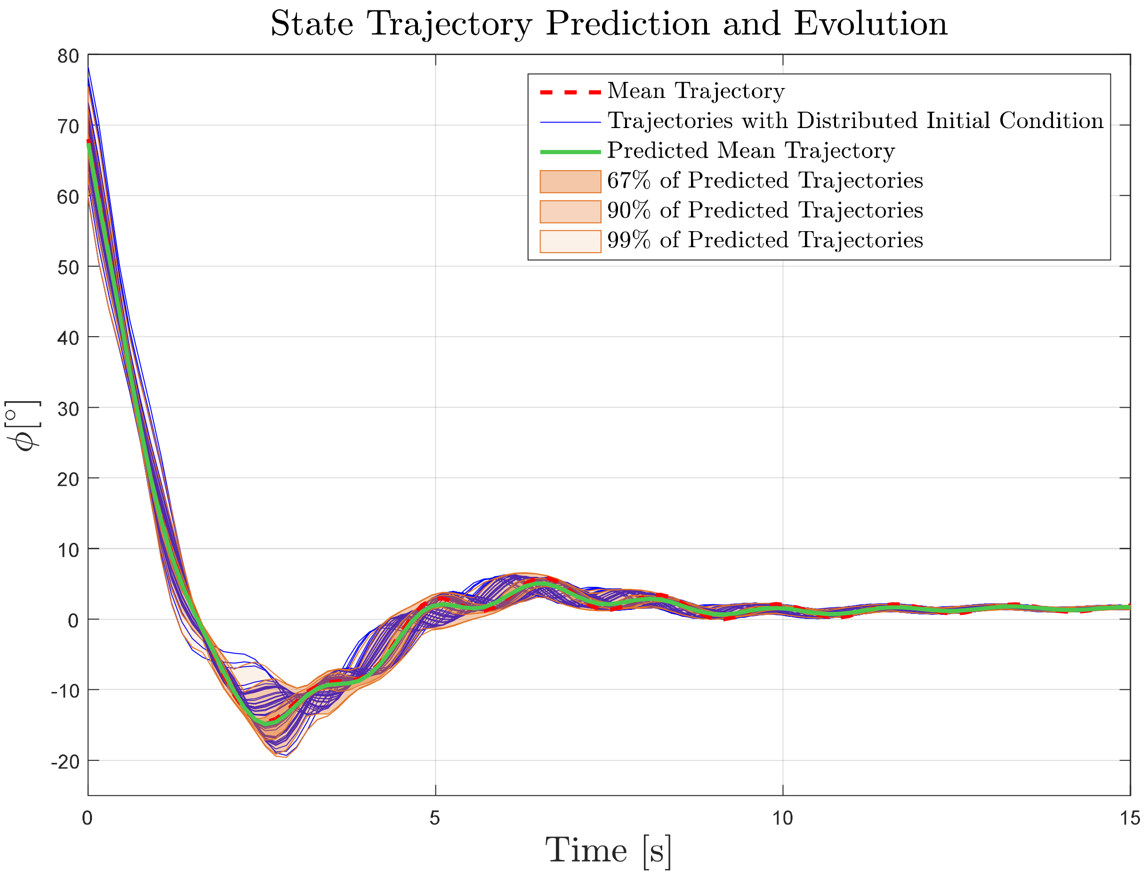


Figure 5-7: Predicted evolution of the roll angle ϕ over 15 seconds leveraging a GMM

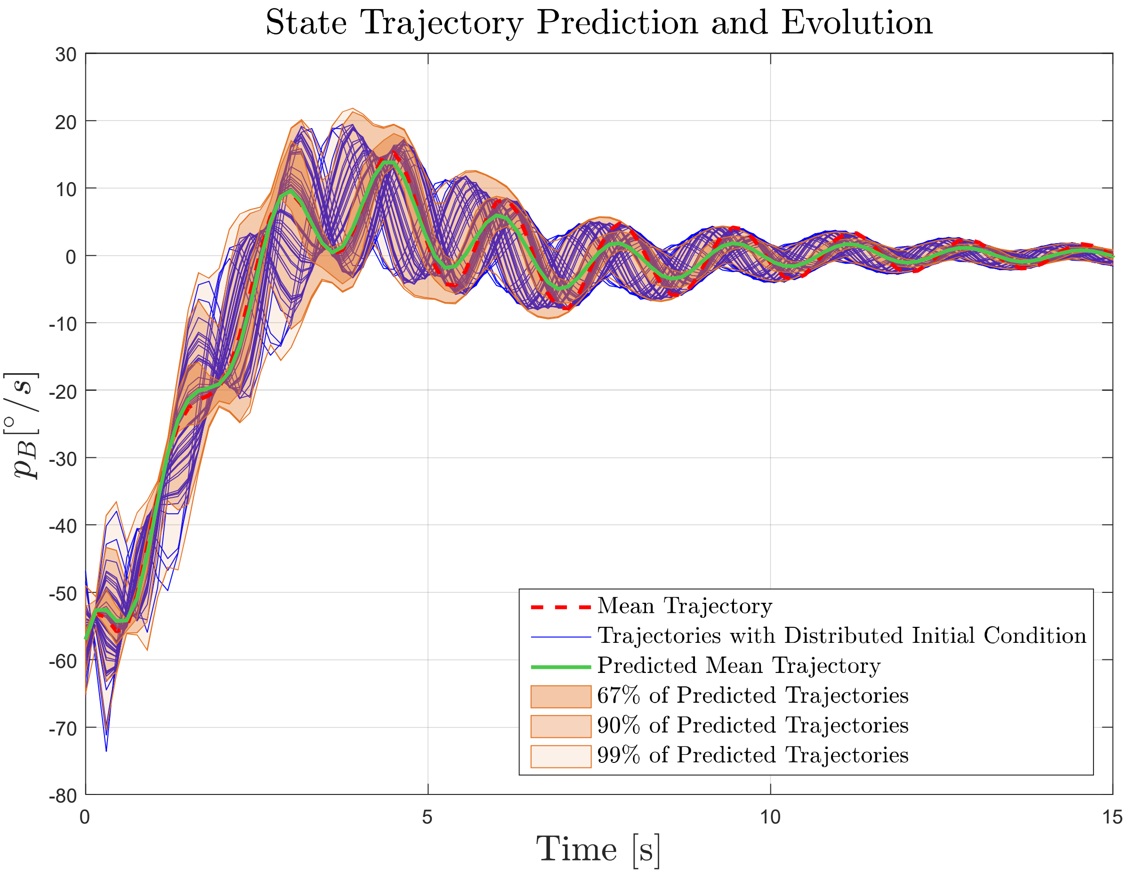


Figure 5-8: Predicted evolution of the roll rate $p_B$over 15 seconds leveraging a GMM

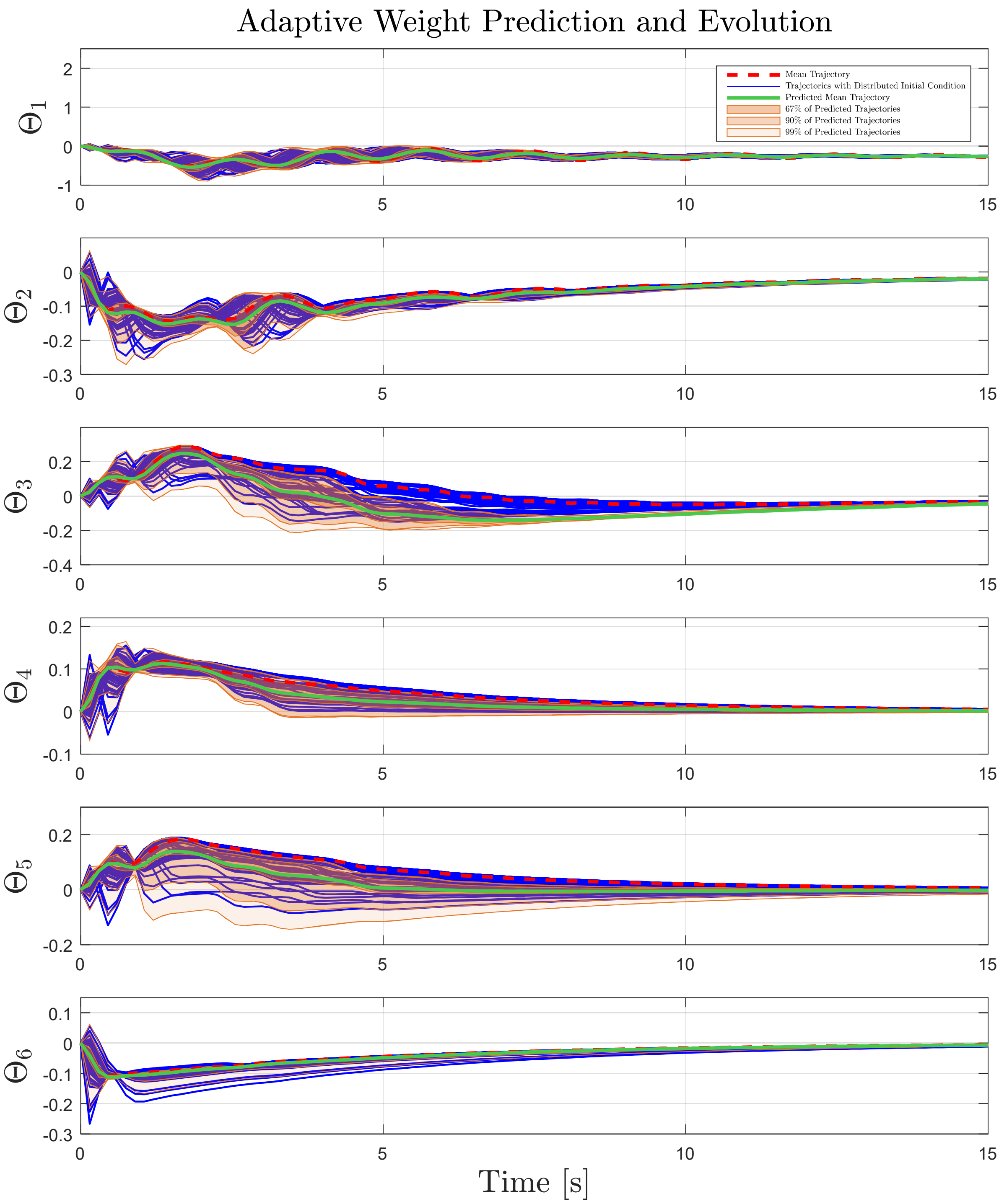


Figure 5-9: Predicted evolution of the adaptive weights over 15 seconds of the wing-rock simulation leveraging a GMM

The predicted area in Figure 5-7 and Figure 5-8 is seen to encompass nearly every simulated state trajectory at each time instant. Furthermore, the 67%, 90% and 99% channels are seen to be asymmetrical around the mean. This indicates that the GMM represents a non-Gaussian probability distribution. These observations lead to the hypothesis that the GMM is able to predict the state pdf at each time instant despite the nonlinear dynamics. In the following, the GMM based state prediction is compared to a forecast, which propagates a normal distribution only. In particular, first the prediction of the mean is analysed, then followed by the full state distribution. Thereby, the aforementioned hypothesis of the GMM approximation capability is tested and analyzed.

Comparison of the predicted mean trajectory between GMM and normal distribution

Figure 5-10 and Figure 5-11 show the state pdf prediction using a normal distribution (which is equivalent to using a single mixture element) instead of a GMM. It can be seen that the predicted mean roll angle and roll rate match the simulated ones with only minor errors. This leads to the hypothesis that the assumption on a normally distributed state trajectory is sufficient to predict its mean with satisfactory performance. In order to compare the results numerically, the following three metrics are employed. The average mean error E_{AM} describes the average deviation of the predicted mean from the real one and is defined as

$$E_{AM} = \frac{1}{k_{end}} \sum_{k=1}^{k_{end}} \left| e_{pred,k} \right|. \qquad \textbf{(5-14)}$$

Here, $e_{pred,k}$ defines the error between the predicted mean and the real one at the time instant k. The truncated $\mathcal{L}_2$ norm is defined as

$$E_{\mathcal{L}_2} = \frac{1}{k_{end}} \sqrt{\sum_{k=1}^{k_{end}} \left| e_{pred,k} \right|^2}. \qquad \textbf{(5-15)}$$

Finally, the third metric $E_{\mathcal{L}_\infty}$ measures the maximum absolute mean error occurring during the simulation:

$$E_{\mathcal{L}_\infty} = \max_{k \in (1 \dots k_{end})} \left| e_{pred,k} \right|. \qquad \textbf{(5-16)}$$

The metrics are evaluated for both, the prediction with the GMM as well as for the forecast with a normal distribution only. Furthermore, the two plant states are evaluated independently. Table 5-3 summarizes the results for the application of the metrics to the $15s$ and $40s$ simulation. It can be seen that the application of the GMM compared to a single normal distribution reduces each error metric by approximately 30% for each plant state. Note that the effect decreases for longer simulation time. This again can be attributed to the convergence of the adaptive parameters and that the nonlinear effect on the error dynamics decreases. Even though the GMM shows supreme performance, the absolute error values are still acceptable in most applications. This leads to the conclusion that using a normal distribution as the basis for the pdf forecast would be sufficient if only the mean trajectory is of interest. However, in order to assess the risk or probability of a requirement violation, the full distribution needs to be taken into account.

	Roll Angle ϕ			**Roll Rate p_B**		
	$E_{AM}[°]$	$E_{\mathcal{L}_2}[°]$	$E_{\mathcal{L}_\infty}[°]$	$E_{AM}[°/s]$	$E_{\mathcal{L}_2}[°/s]$	$E_{\mathcal{L}_\infty}[°/s]$
GMM $(15s)$	0.21	0.0086	1.19	0.72	0.0289	3.16
Normal $(15s)$	0.32	0.0123	1.66	1.05	0.0405	4.62
GMM $(40s)$	0.23	0.0052	1.20	0.63	0.0140	3.16
Normal $(40s)$	0.27	0.0061	1.66	0.76	0.0177	4.62

Table 5-3: Average mean error, $\mathcal{L}_2$ −like norm and maximum mean error for the prediction of the mean trajectory in the wing-rock simulation example comparing a GMM and a normal distribution forecast

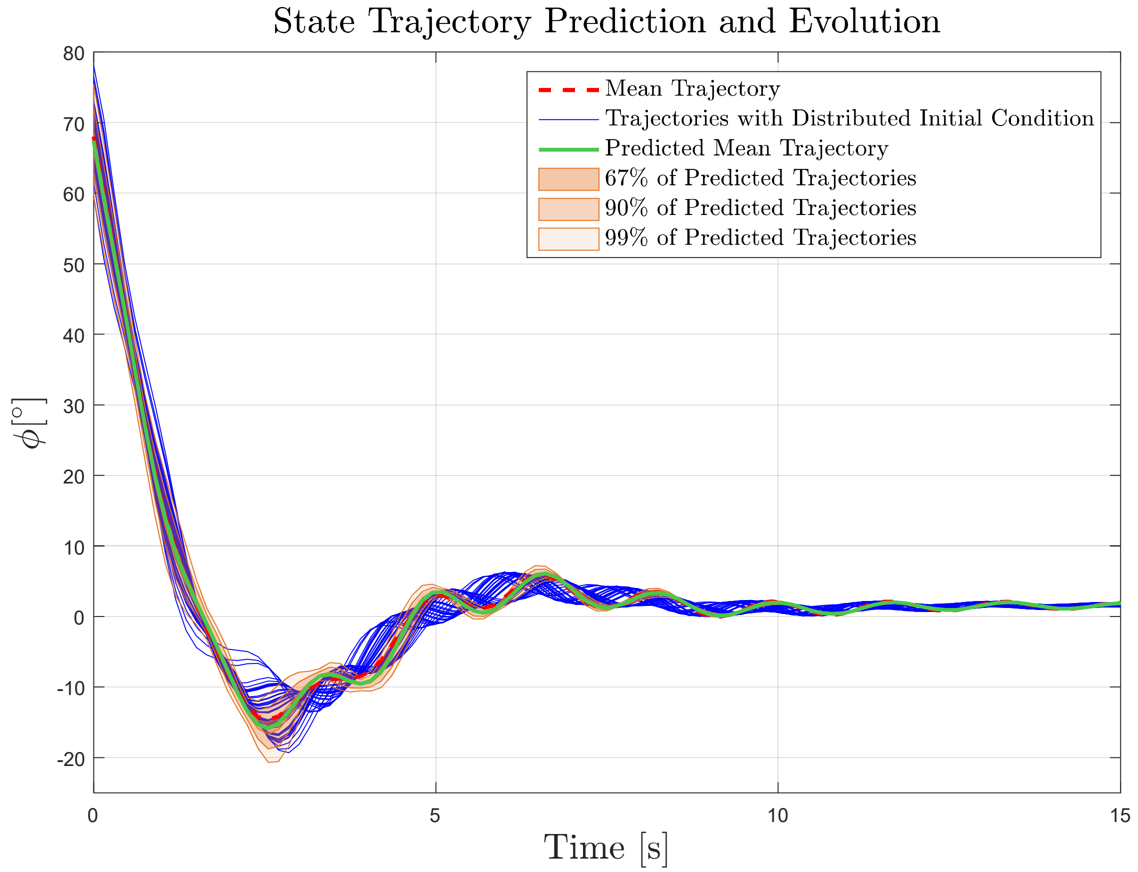


Figure 5-10: Predicted evolution of the roll angle ϕ over 15 seconds leveraging a normal distribution

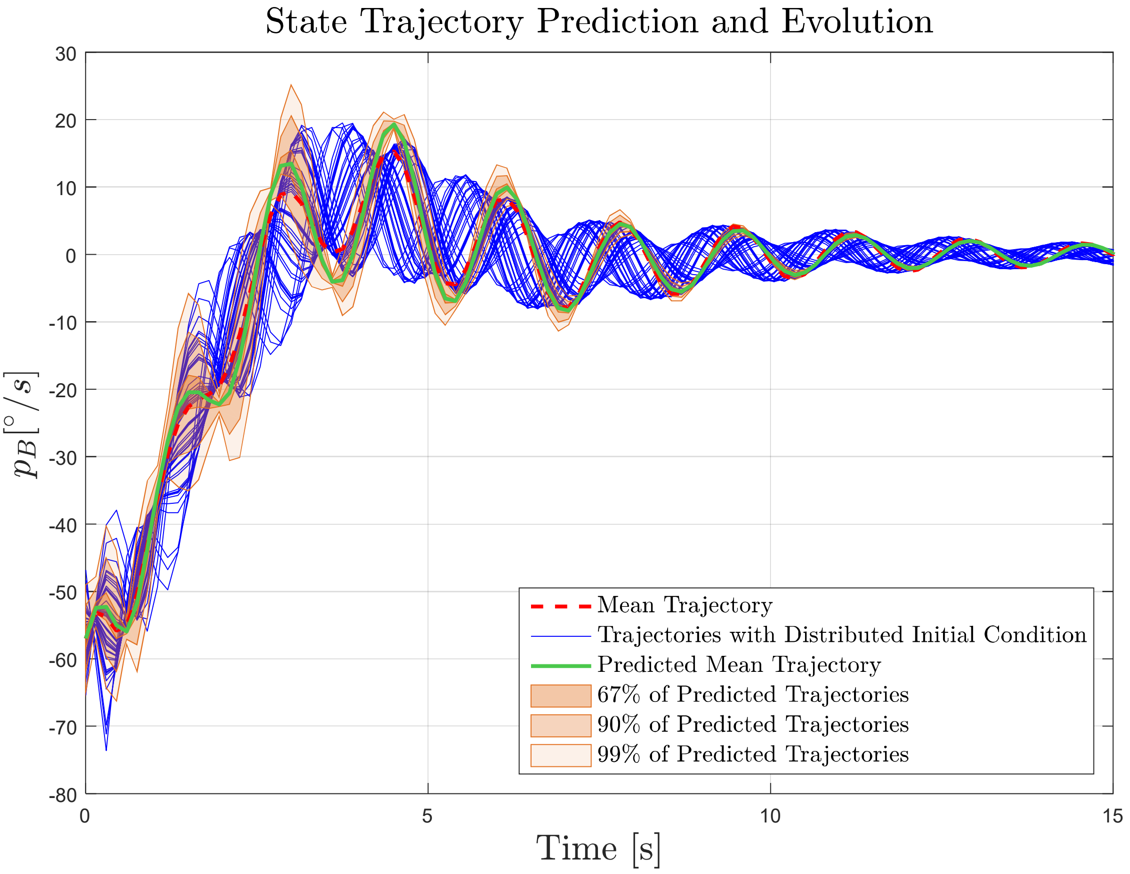


Figure 5-11: Predicted evolution of the roll rate $p_B$over 15 seconds leveraging a normal distribution

Comparison of the full pdf approximation

The predicted channels in Figure 5-7 and Figure 5-8 are seen to encompass nearly every simulated state trajectory at each time instant. In comparison, the predicted channels in Figure 5-10 and Figure 5-11 are seen to encompass the simulated state trajectories only in rare time instants. During the remaining time, large errors between the predicted and actual state trajectory occur. To better analyse both arguments the predicted pdf is analysed at various time instants.

Figure 5-12 and Figure 5-13 respectively show the approximation of the plant state pdf for the roll angle and roll rate at different prediction steps, namely after $0s, 1s$,$2.8s$ and $5s$. The pdf of the simulated state trajectories is represented by a histogram. Additionaly, the predicted pdf using a GMM and a Gaussian is depicted. Note that the latter differs from the approximating normal in Figure 5-4 and Figure 5-5. In the former case, the histogram was approximated by a normal distribution at each time instant. Here, the initial state distribution is assumed to be normal and then propagated through the nonlinear dynamics.

It can be seen that the normal distribution as well as the GMM approximate the initial state pdfs with no visible error. After $1s$ into the simulation the roll angle distribution is seen to become slightly skewed to the left. The GMM is seen to accurately approximate the pdf. In contrast, the normal distribution is unable to recreate the skewness. However, only minor errors occur in the latter case. In contrast, the roll rate pdf is seen to be highly skewed to the right and truncated at approximately $-37\ \frac{\circ}{s}$. The GMM approximation is seen to recreate the skewness with a shift of approximately $2\ \frac{\circ}{s}$. Note that the GMM is unable to completely recreate the truncation, since it can only represent continuous pdfs. On the other hand, the normal distribution is seen to be unable to recreate both the skewness and the truncation.

From $1s$ to $2.8s$ the roll angle distribution becomes skewed to the right. The GMM is able to recreate the general shape of the distribution, including a sharp peak at around $-10°$. However, the predicted peak is shifted by approximately $1°$. This can be attributed to the propagation of errors, in particular from previous errors in the roll rate estimation. At this time step, the roll angle pdf is badly approximated by a Gaussian. In particular, the difference between the mean and the developed peak is approximately $5°$, which can lead to serious misjudgement. The roll rate pdf is seen to keep its skewness to the right. While the GMM can represent the general shape of the pdf well, it keeps its shift of approximately $2\ \frac{\circ}{s}$. This can be attributed to propagated errors. Note that the overlap between the Gaussian and the state pdf is minimal.

Finally, after $5s$ into the simulation the roll angle and roll rate distribution become multimodal. Furthermore, the roll rate distribution changes its skewness from the right to the left side. The GMM is able to represent the two modes in both cases. The GMM approximation of the roll angle peaks is shifted by approximately $0.5°$ respectively. On the other hand, the Gaussian approximation neglects the left mode and centers around the right one, thus again introducing significant errors into a potential requirement violation assessment. The GMM approximation of the roll rate inherits its shift of approximately $2\ \frac{\circ}{s}$, but retains the general shape of the roll rate pdf. Note that the Gaussian approximation is sharply centered around $1.5\frac{\circ}{s}$. Assessing the requirement violation using the latter leads to serious faults and constitutes the use of GMMs instead of Gaussian approximations.

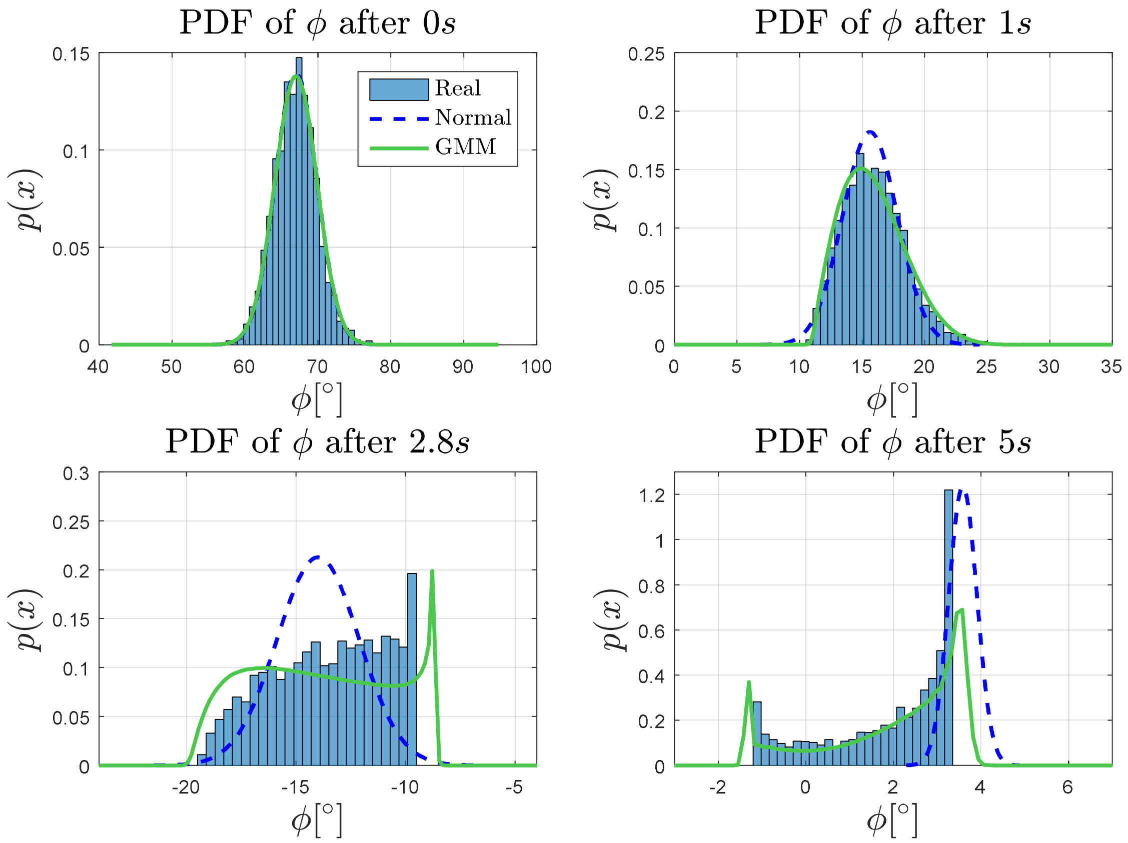


Figure 5-12: Predicted pdf of the roll angle ϕ in the wing-rock example after 0s, 1s, 2.8s and 5.0s

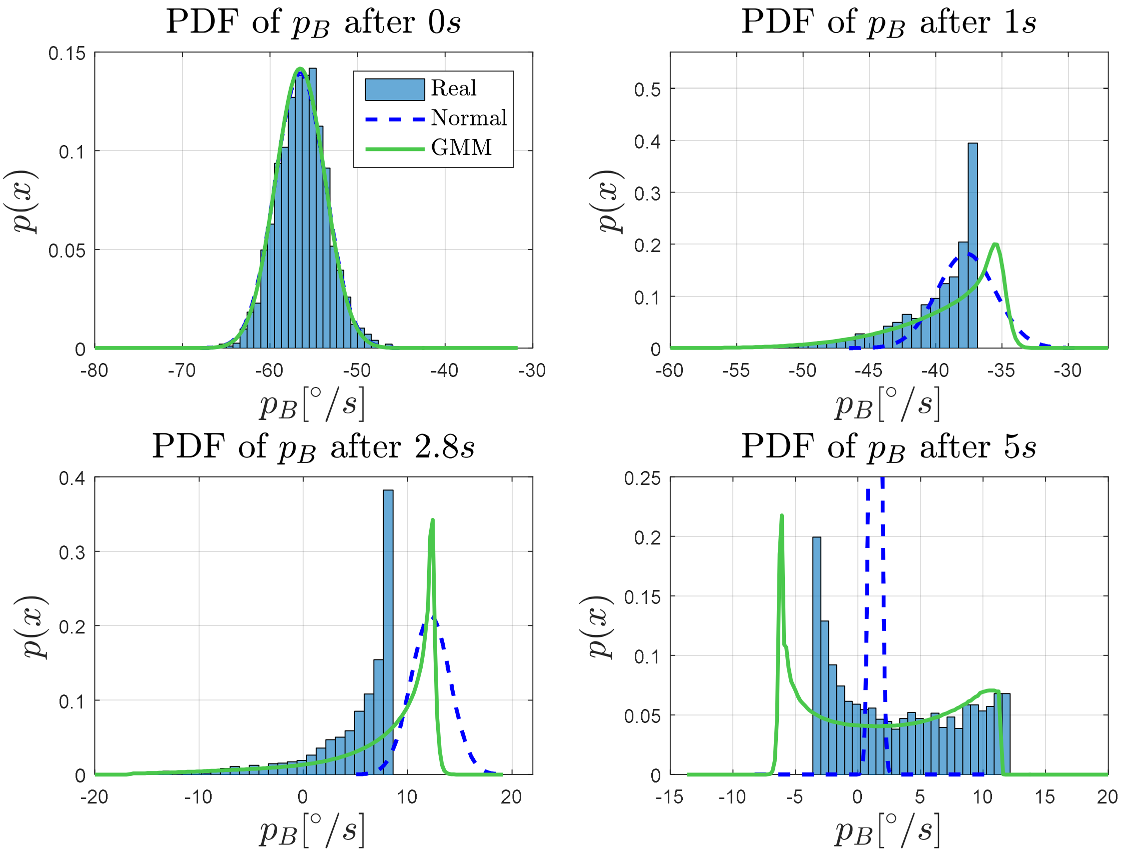


Figure 5-13: Predicted pdf of the roll rate $p_B$in the wing-rock example after 0s, 1s, 2.8s and 5.0s

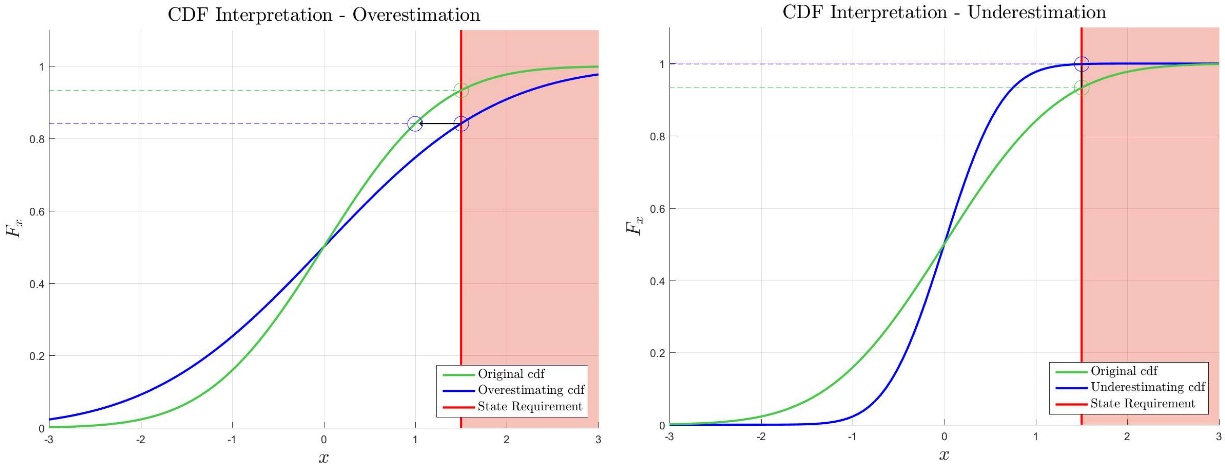


Figure 5-14: Interpretation of over- and underestimating distributions

The proposed monitoring approach calculates the probability of an imminent requirement violation. In this context, calculating the cdf instead of the pdf allows for better interpretability. In this, the concepts over- and underestimation play a vital role. Figure 5-14 shows an interpretation of both concepts. The underlying goal of this example is to determine with which probability a state requirement violation occurs. The latter is arbitrarily selected to $x_{req} = 1.5$. Figure 5-14 (left) shows the cdf of a random variable x as well as a cdf, which is overestimating the distribution of x. The cdf of x is seen to cross the state requirement at a value of $F_x = 0.93$. Hence, the probability of a requirement violation is $P(\|x > x_{req}\|) = 0.07$. In contrast, the overestimating cdf is seen to cross the state requirement at a value of $F_x = 0.84$. In consequence, the probability of a requirement violation is $P(\|x > x_{req}\|) = 0.16$ in this case. Assuming that the overestimating cdf corresponds to the state distribution predicted by the forecast, the monitor would issue a warning earlier than necessary, as the probability of a requirement violation is perceived to be higher than in reality. In fact, the gap between the original and the overestimating cdf for the same probability $P(\|x > x_{req}\|) = 0.16$ corresponds to $\Delta x = 0.5$.

In contrast, Figure 5-14 (right) shows the underestimation of the cdf of x. Again the probability of a requirement violation is $P(\|x > x_{req}\|) = 0.07$. In contrast, the underestimating cdf is seen to cross the state requirement at a value of $F_x = 0.9987$. Hence, the probability of a requirement violation is $P(\|x > x_{req}\|) = 0.0013$ in this case. Again, assuming that the underestimating cdf corresponds to the state distribution predicted by the forecast, the monitor would issue warnings later, as the probability of a requirement violation is perceived to be lower than in reality. In summary, an overestimating forecast tends to issue more warnings and thus increase the false positive rate, while an underestimating forecast misses warnings and increases the false negative rate. In the opinion of the author, overestimation of the monitoring forecast is preferred to underestimation, as the system should not miss a single probable requirement violation.

Figure 5-15 and Figure 5-16 respectively show the approximation of the plant state cdf for the roll angle and roll rate at different prediction steps, namely after $0s, 1s$,$2.8s$ and $5s$. The plots depict the cdf of the simulated trajectories, the predicted cdf using a GMM and the predicted cdf using a Gaussian. Furthermore, the circles indicate the borders of the channels encompassing the central 67%, 90% and 99% of the state trajectories. In this, the circles form pairs. In particular, the third and fourth circle define the 67% channel, the second and fifth circle describe the 90% channel and the first and sixth circle describe the 99% channel.

Several conditions can be derived from the discrepancy between the approximating GMM (or normal) and the distribution of the simulated trajectories. If all circles of the approximating GMM (or normal) match with the real ones, the predicted and real channels match. If the three left circles corresponding to the GMM (or normal) are farther to the left than the real ones and if the three right circles corresponding to the GMM (or normal) are farther to the right than the real ones, then the GMM (or normal) is overestimating the trajectory distribution. As a result, the monitoring algorithm will indicate an imminent requirement violation even though there is still some buffer. On the other hand, if the three left circles corresponding to the GMM (or normal) are farther to the right than the real ones and if the three right circles corresponding to the GMM (or normal) are farther to the left than the real ones, then the GMM (or normal) is underestimating the trajectory distribution. As a result, the monitoring algorithm will not indicate a requirement violation even though it is imminent.

In Figure 5-15 and Figure 5-16 it can be seen that the initial cdfs of the GMM, the normal and the simulated trajectories are nearly indistinguishable. This indicates that the GMM approximates the initial normal state distribution well. Note that the outer right circle of the initial trajectory distribution is farther to the right than its corresponding approximations. This can be attributed to the fact that the cdf is numerically derived from simulating the closed-loop system 2500 times. A single draw at the limits of the initial distributions can already cause a significant shift.

After $1s$ the cdf of the roll angle ϕ is still seen to retain the shape of a Gaussian cdf. This corresponds to the respective pdf in Figure 5-13. The circles of the GMM approximation nearly completely overlap with the simulated trajectories, while the circles corresponding to the normal distribution deviate from the latter. This indicates that the approximation of the channels starts to deviate from the simulated trajectories. This becomes even more apparent when taking the cdf of the roll rate p_B into account. It can be seen the cdf of the simulated trajectories strongly deviates from a Gaussian cdf. Still, the GMM is able to recreate its general shape. Note that all circles of the approximating GMM lie farther from the mean than the real ones. This indicates that the predicted channels are larger than necessary. In contrast, the Gaussian approximation fails to approximate the cdf and the predicted circles are significantly different to the real ones.

From $1s$ to $5s$ the cdf of the simulated trajectories keeps changing its underlying shape for both plant states. In any case, the GMM is able to recreate its general shape with minor errors.

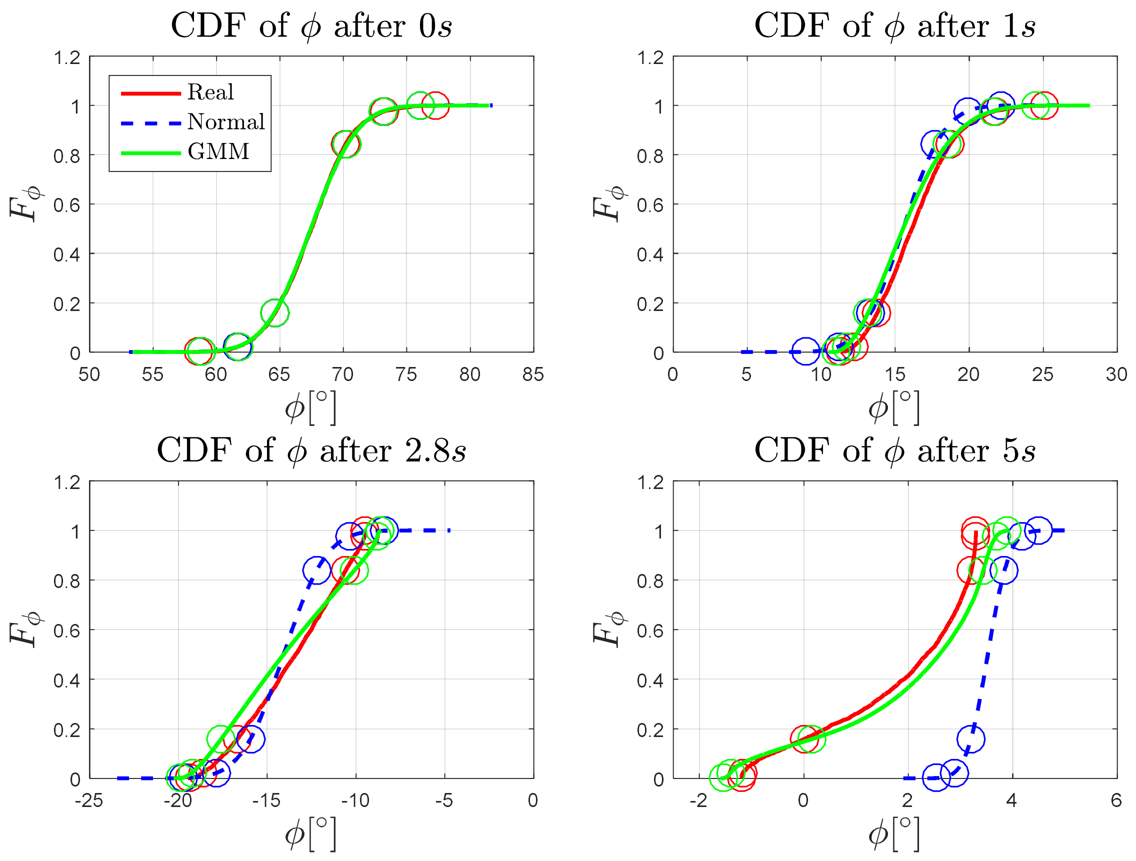


Figure 5-15: Predicted cdf of the roll angle ϕ in the wing-rock example after 0s, 1s, 2.8s and 5.0s

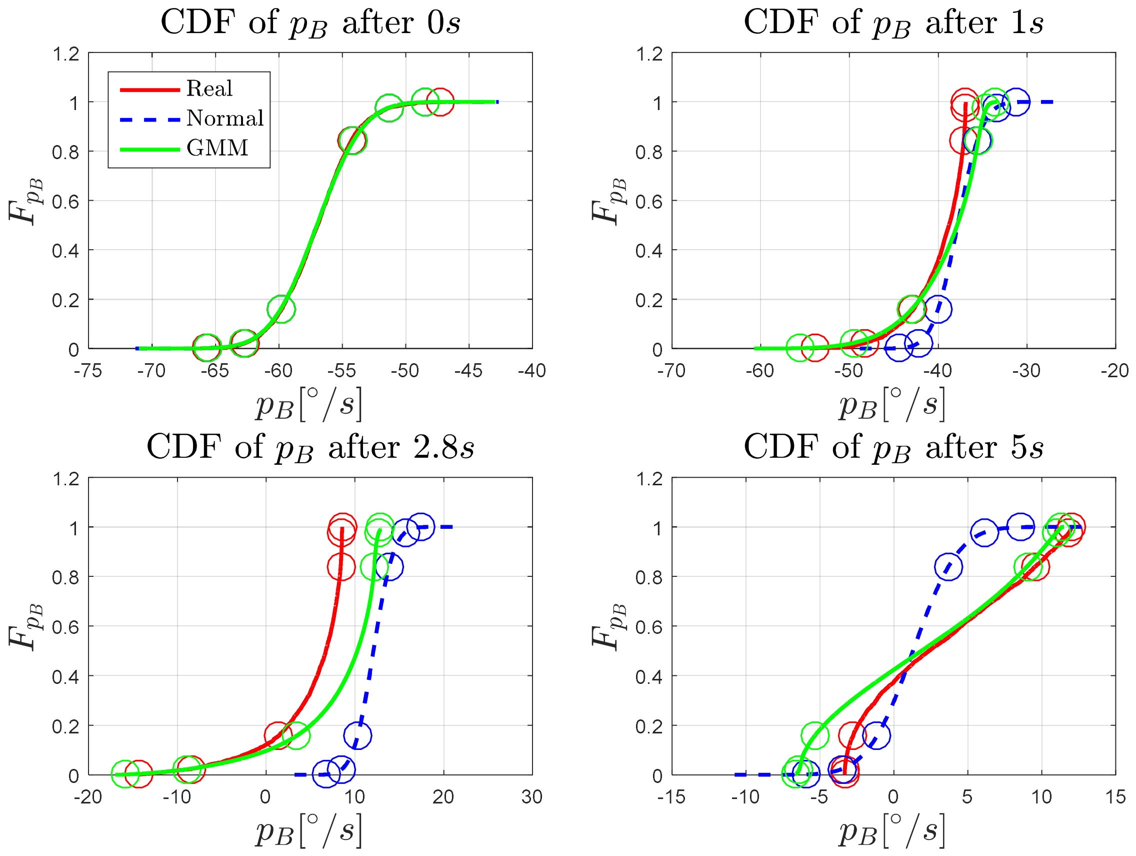


Figure 5-16: Predicted pdf of the roll rate p_B in the wing-rock example after 0s, 1s, 2.8s and 5.0s

Assessing the presented figures already provides good inside into the prediction capability of the GMM. For a numerical assessment, this thesis introduces the *Grade of Circumvallation* (GoC). The metric is intended to penalize underestimation. The underlying idea is that an overestimation leads to more responses from the monitor, but underestimation might lead to a missed requirement violation and put the system in danger. Hence, for the purpose of this thesis, overestimation is preferred to underestimation. Consider the following test function:

$$f_{x,CL}(j) := \begin{cases} F_{x,app}^{-1}(p_p) + \Delta x & \text{if } p_p \geq 0.5 \\ F_{x,app}^{-1}(p_p) - \Delta x & \text{if } p_p < 0.5 \end{cases}. \tag{5-17}$$

Here, $F_{x,app}^{-1}(p_p)$ represents the evaluation of the inverse distribution function of the approximating distribution for the state x, p_p represents the percentile and Δx represents a relaxation term, which artificially allows to increase the area of $F_{x,app}^{-1}(p_p)$. The relaxation term makes the monitor more responsive, but decreases the amount of missed alarms. Consider the following mapping of the test function to a binary decision variable $\delta_{x,CL}(p_p)$:

$$\delta_{x,CL}(p_p) := \begin{cases} 1 & \text{if } F_{x,app}^{-1}(p_p) + \Delta x \geq F_x^{-1}(p_p) \ \wedge \ p_p \geq 0.5 \\ 0 & \text{if } F_{x,app}^{-1}(p_p) + \Delta x < F_x^{-1}(p_p) \ \wedge \ p_p \geq 0.5 \\ 1 & \text{if } F_{x,app}^{-1}(p_p) - \Delta x \geq F_x^{-1}(p_p) \ \wedge \ p_p < 0.5 \\ 0 & \text{if } F_{x,app}^{-1}(p_p) - \Delta x < F_x^{-1}(p_p) \ \wedge \ p_p < 0.5 \end{cases}. \tag{5-18}$$

In order to evaluate one channel, the respective percentile pair needs to be evaluated. E.g. to evaluate the 67% channel, the GoC needs to be calculated for $D_{p_p} \coloneqq \{16.5, 83.5\}$. The GoC for the j −th channel is defined as

$$GoC_{x_j} = \frac{1}{2k_{end}} \sum_{k=1}^{k_{end}} \sum_{p_p \in D_{p_p}} \delta_{x_k,CL}(p_p). \tag{5-19}$$

Note that the GoC is normalized, non-negative and assumes a value between zero and one. For interpretational purposes, $GoC_{x_{67}} = 1$ means that the 67% channel of the simulated state trajectories is completely engulfed by the 67% channel of the predictive distribution. In contrast, $GoC_{x_{67}} = 0$ means that the distribution of the simulated state trajectories is completely underestimated during the simulation. Note that this does not imply that the predicted and the real channel do not overlap. Rather the predicted channel is too narrow. On the other hand, $GoC_{x_{67}} = 0.5$ can represent the case that the predicted and simulated channel do not overlap at all. While this seems adverse to the purpose of a metric at first, in this case one of the predicted channel borders still overestimates one of the channel borders of the trajectory distribution.

The relaxation term in (5-17) not only enables the artificial increase of overestimation, but also allows for insights into the errors made during the prediction. In fact, approximation and numerical errors might lead to an adverse evaluation of (5-18) even though they are small in comparison. The following argument attempts to highlight this concept.

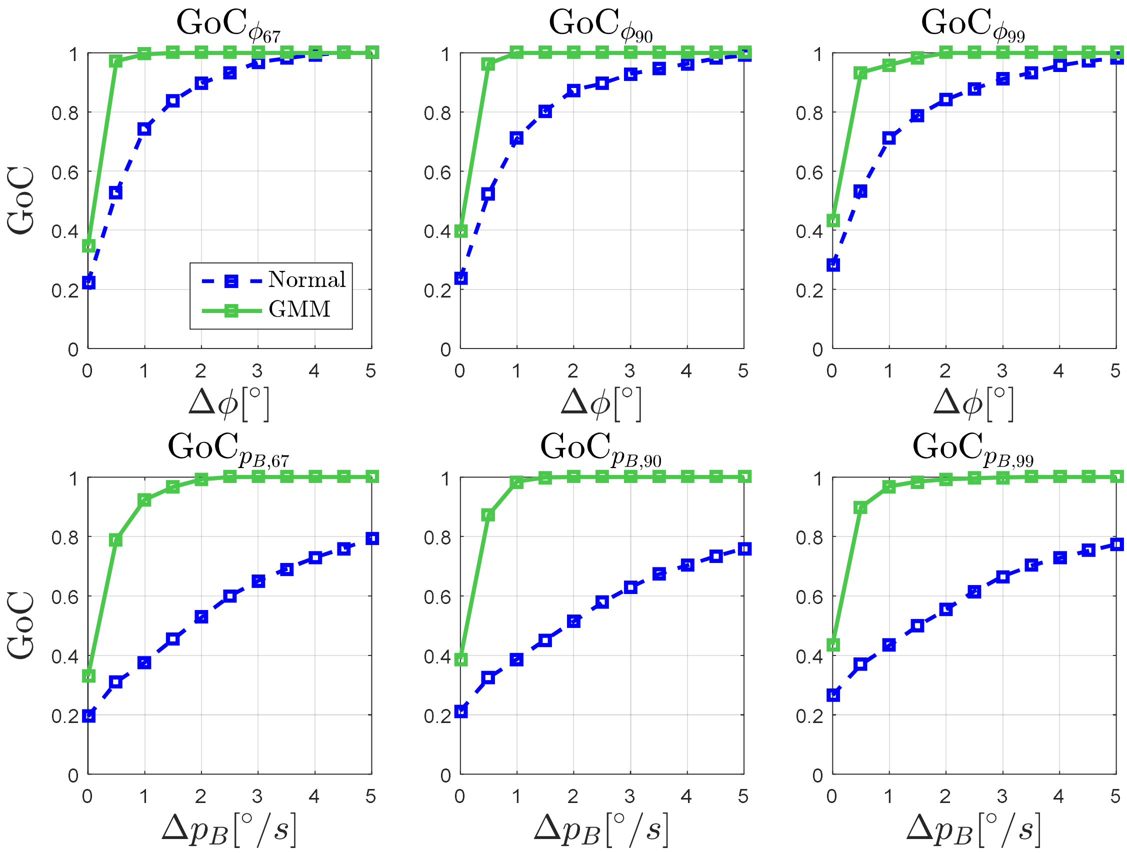


Figure 5-17: GoC of the plant states with varying relaxation term for the wing-rock example

Figure 5-17 shows the GoC for both plant states, the three channels, the GMM as well as the Gaussian approximation and varying numerical values for the relaxation term Δx. The GoC is calculated for the first 15 seconds into the simulation. In conjunction with Figure 5-17, Table 5-4 lists the numerical values of the GoC at various Δx. It can be seen that for all channels and without relaxation term the GoC for the GMM approximation is roughly twice as large as for the Gaussian approximation. Still, the GMM as well as the normal results lie within a similar region. This raises the hypothesis that using a normal distribution for the forecast is sufficient for prediction and the additional cost of using a GMM can be saved. As already demonstrated, this is not the case. In fact, the introduction of the relaxation term with even only small values $\Delta\phi = 0.5°, \Delta p_B = 0.5°/s$ already doubles or triples the GoC values for the GMM. The GoC for the Gaussian approximation increases only slightly. This indicates that the channels predicted by the GMM deviate only slightly from the simulated ones. Increasing the relaxation term further quickly leads to the GoC approaching one. This indicates that small approximation errors occur during the forecast, which prevent the predicted GMM to completely represent the true state trajectory distribution. The small approximation errors are a direct consequence of the neural network approximation property (see i.a. [130], [69], [165]). In contrast, even for large relaxation values, the normal distribution mostly sustains large errors and is thus inferior to GMMs in performance. In essence, the preceding argument highlights that the designer should consider the impact of approximation errors on the GMM based forecast results by e.g. introducing a relaxation term in the form of a slack variable.

State / Forecast Basis	$GoC_{x_{67}}$	$GoC_{x_{90}}$	$GoC_{x_{99}}$
$\boldsymbol{\Delta\phi = 0°, \Delta p_B = 0°/s}$			
ϕ – GMM	0.346	0.396	0.435
ϕ – Normal	0.222	0.235	0.281
p_B – GMM	0.329	0.388	0.436
p_B – Normal	0.195	0.213	0.266
$\boldsymbol{\Delta\phi = 0.5°, \Delta p_B = 0.5°/s}$			
ϕ – GMM	0.973	0.964	0.934
ϕ – Normal	0.526	0.524	0.534
p_B – GMM	0.790	0.873	0.900
p_B – Normal	0.310	0.325	0.369
$\boldsymbol{\Delta\phi = 1°, \Delta p_B = 1°/s}$			
ϕ – GMM	0.996	1.0	0.960
ϕ – Normal	0.745	0.711	0.711
p_B – GMM	0.924	0.985	0.969
p_B – Normal	0.377	0.388	0.433
$\boldsymbol{\Delta\phi = 2°, \Delta p_B = 2°}/s$			
ϕ – GMM	1.0	1.0	1.0
ϕ – Normal	0.897	0.804	0.789
p_B – GMM	0.992	1.0	0.993
p_B – Normal	0.528	0.514	0.555

Table 5-4: GoC in the wing-rock simulation example

Evaluation of the indicator function

Finally, in order to evaluate if a potentially undesired situation occurs within the prediction horizon, the indicator functions and the monitoring output from section 4.5.3 are computed. For the sake of highlighting the concept, the state requirements are arbitrarily set. The roll angle shall not exceed $\phi_{req,high} = 30°$ and shall not fall below $\phi_{req,low} = -15°$. Similarly, the roll rate shall not exceed $p_{B,high} = 60\frac{\circ}{s}$ and shall not fall below $p_{B,high} = -30\frac{\circ}{s}$. Table 5-5 summarizes the parameters used for the indicator function and the monitoring output.

Figure 5-18 depicts the predicted probability of a roll angle requirement violation according to equation (4-102) within $15s$ of the simulation. It can be seen that the initial conditions already violate the upper state requirement. Consequently, the probability of a requirement violation is returned as 1. Note that the presented example is constructed and intended to demonstrate the capabilities of the prediction. In reality, the initial state would lie within the boundaries of the requirements. After about $1s$, the predicted roll angle distribution enters the corridor between the upper and lower requirement. As a consequence, the probability of a requirement violation drops to zero. At around $2.5s$ the predicted mean trajectory is seen to violate the lower state requirement. While, the estimated probability of a requirement violation increases, it still remains lower than 1 as a huge portion of the distribution remains within the formulated requirements. This highlights the fact, that the mean of a distribution seldom inherits enough information in order to grasp a certain situation. Finally, the predicted state distribution converges to the zero roll angle command. Consequently, the probability of a requirement violation returns to zero. Similarly, Figure 5-19 depicts the predicted probability of a roll rate requirement violation according to equation (4-102) within $15s$ of the simulation. The results and conclusions mirror the roll angle case.

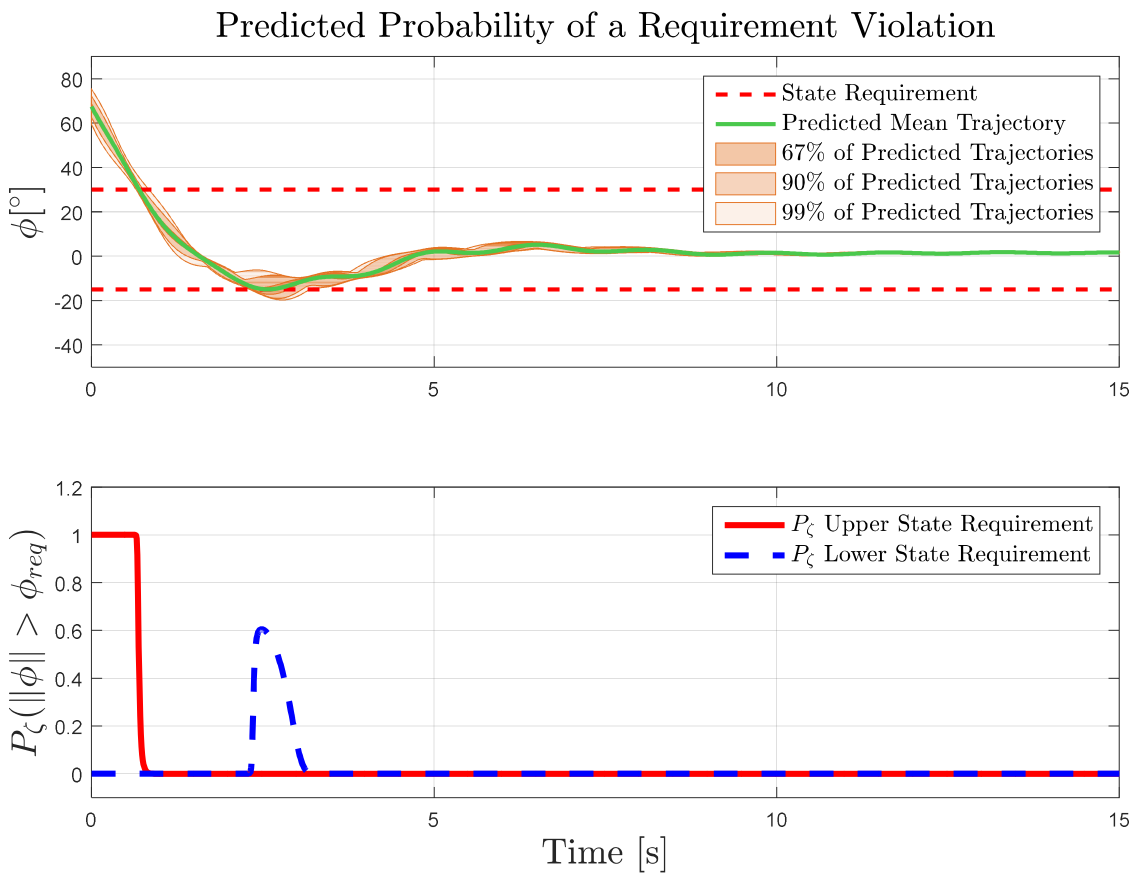


Figure 5-18: Probability of a roll angle requirement violation in the wing-rock example

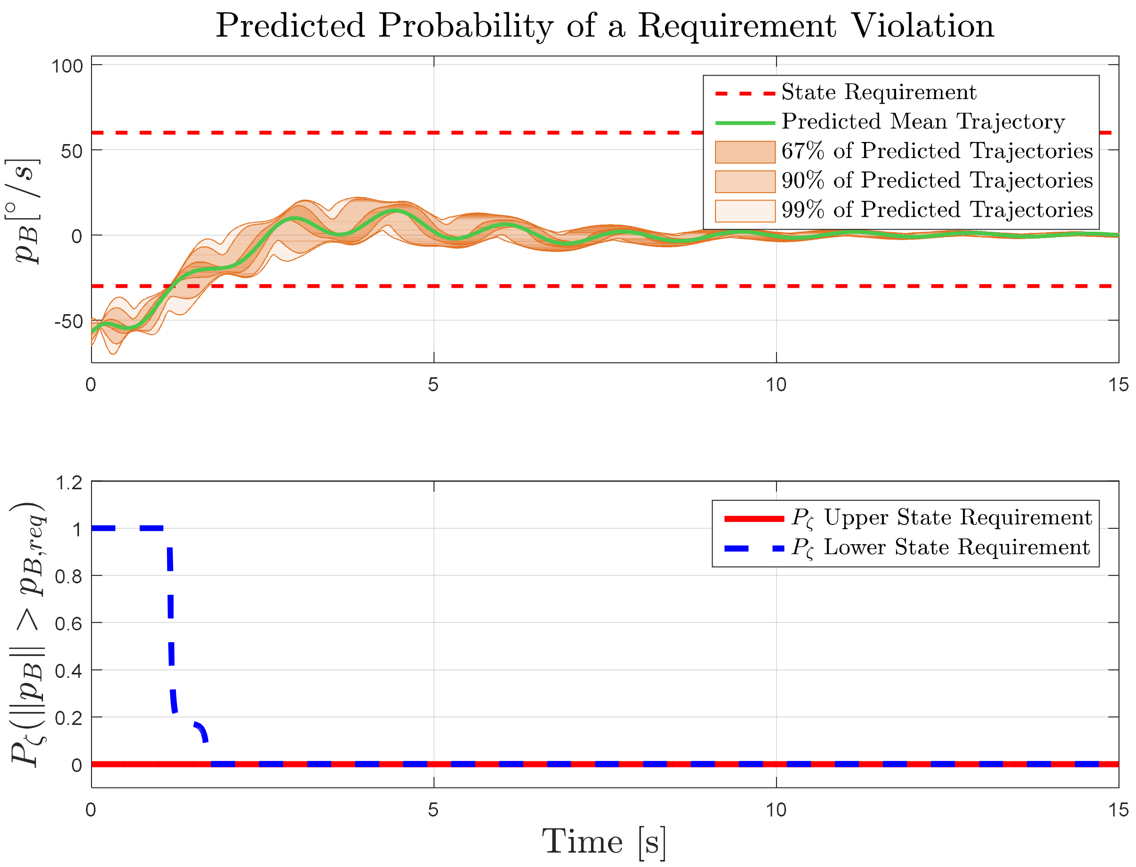


Figure 5-19: Probability of a roll rate requirement violation in the wing-rock example

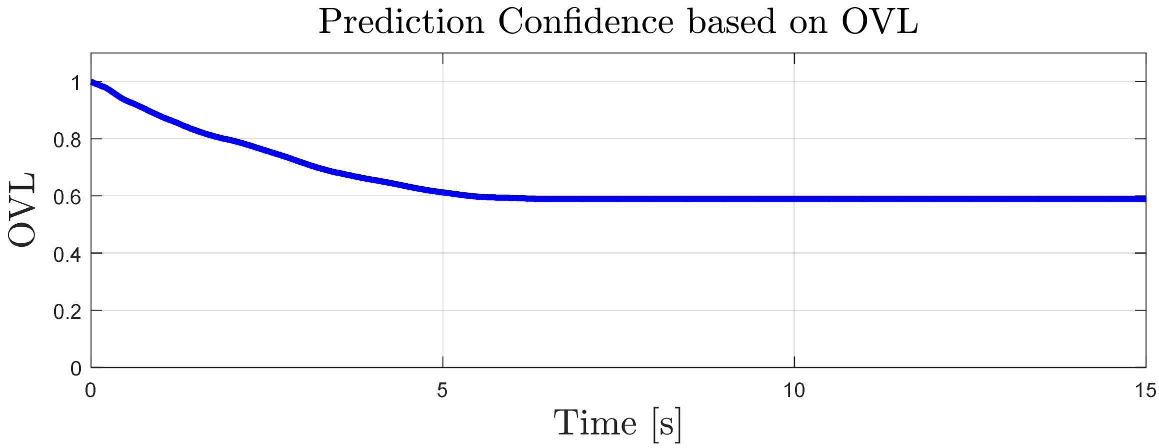


Figure 5-20: Predicted Confidence based on OVL in the wing rock example

Simulation Parameter	Numerical Value
Indicator Function	
δ_ζ	0.7
α_{ind}	1
β_{ind}	4000
δ_{ind}	20
$\phi_{req,low}$	$-15°$
$\phi_{req,high}$	$30°$
$p_{B,low}$	$-30°/s$
$p_{B,high}$	$60°/s$

Table 5-5: List of parameters used for the indicator function and monitoring output in the wing rock simulation example

Using the parameters in Table 5-5 for the indicator functions of equations (4-98)-(4-102), leads to a warning by the monitor. This can be attributed to the fact that the indicator function takes the full prediction horizon into account (i.e. $\alpha_{ind} = 1$ and $\beta_{ind} = 4000$). Within this horizon, the probability limit $\delta_\zeta = 0.7$ is exceeded in at least $\delta_{ind} = 20$ time steps.

In order to derive a confidence in the prediction, the *Overlapping Coefficient* (OVL) from section 4.5.2 is computed. The OVL result at each prediction step is rounded at the fourth decimal. Figure 5-20 shows the resulting OVL curve. The OVL is seen to decrease to a value of approximately 0.6 within $5s$ into the simulation. In this phase, the variance of the single mixture elements is still large enough that the linearization of the nonlinear system dynamics is not completely warranted over the complete support of each mixture element. After $5s$, the OVL remains approximately constant. This can be approximated to the decreasing variance in the mixture elements. Furthermore, numerical errors are mitigated by rounding the single OVL results at the fourth decimal.

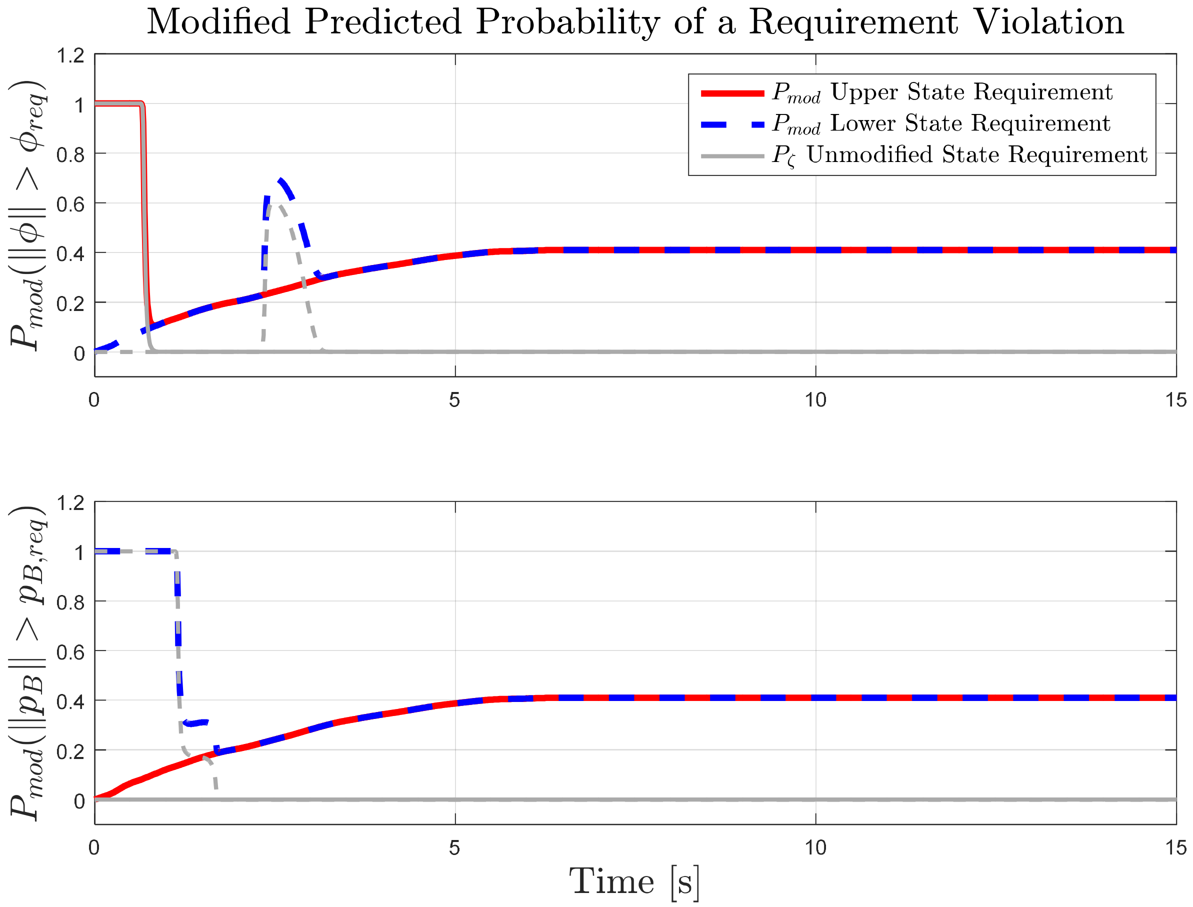


Figure 5-21: Modified predicted probability of a requirement violation in the wing rock example

In a final step, the predicted probability of a requirement violation of Figure 5-18 and Figure 5-19 is modified by the OVL according to (4-109) in section 4.5.3. Figure 5-21 shows the result of the modified probability P_{mod} as well as the non-modified curves of P_{ζ}. Note that contrary to equation (4-109), here the probability of a requirement violation (and not abidance) of P_{mod} is depicted, in order to be comparable with the formulation of P_{ζ} in (4-102). It can be seen that P_{mod} is always larger than its counterpart P_{ζ}. This is due to the OVL, which increases the conservatism in the prediction. After approximately $5s$ into the simulation, P_{mod} converges to the numerical value of the OVL. This can be attributed to the fact that P_{ζ} in turn converged to 0. A non-zero P_{mod} can now be interpreted to represent an uncertainty in the prediction of the monitor and that a requirement violation is admissible.

The presented case highlights the fact that a small OVL value in the beginning of the prediction horizon can lead to a high uncertainty at the end. On the other hand, the OVL is unmindful and independent of the state requirements. That is, in the case that the predicted state distribution was far away from any formulated limit, the OVL still increases P_{mod} drastically. Still, applying the indicator functions of section 4.5.1 the monitor issues a warning. However, the characteristics of the warnings between P_{mod} and P_{ζ} will be similar as long as δ_{ζ} is kept at $\delta_{\zeta} = 0.7$. Hence, choosing P_{mod} over P_{ζ} in this case has only minor influence on the outcome. The information about the uncertainty in the prediction can still be helpful to the operator. While using the OVL raises the information content, in the authors' opinion, a multitude of different techniques and rules can and should be conceived for each specific use case.

Simulation Parameter	Numerical Value
GP Regression	
p_H	100
σ_n^2	0.05
γ	0.0005
Λ_i	0.25

Table 5-6: List of parameters used GP regression in the wing rock simulation example

5.1.3 Simulation with uncertainty

This section expands the model in 5.1.2 by lifting the assumption that the nonlinerity is known. Instead, the monitor uses GP regression as described in section 4.4.3 in order to approximate the latter online. Apart from that, the same scenario of section 5.1.2 is used.

The data storage algorithm checks if new data points are sufficiently different to already stored data by employing equation (4-93). Other than that, the history stack is fixed and does not discard already stored data points. The bandwidth of the kernels is selected equally to be $\Lambda_i = 0.25$. Table 5-6 summarizes the parameters necessary for recreating the GP regression results.

Figure 5-22 (left) shows the wing rock uncertainty of equation (5-2). In turn, Figure 5-22 (right) shows the approximated mean of the wing rock uncertainty after 40 seconds into the simulation using GP regression and the approach depicted in section 4.4.3. The red dots represent recorded data points. After 40 seconds, the data management algorithm recorded a total of 43 data points. It can be seen that the GP representation starts to resemble the true uncertainty in areas where data points were recorded. However, the approximation quality is hard to gauge from a 3D-plot.

Figure 5-23 shows a contour diagram of the relative approximation error in percent between the true wing rock uncertainty and the GP mean representation. Snapshots of the uncertainty approximation are taken after 2.5, 10, 20 and 40 seconds. Furthermore, the recorded data points at the specific time instants are depicted. It can be seen that the relative approximation error reduces to $0 - 10\%$ in the vicinity of recorded data points. As time progresses, the system visits different areas of the state space, where the GP regression approach of section 4.4.3 continues to decrease the approximation error. However, in areas without data, the approximation error remains large. At first this appears to be a negative property. However, the goal of the GP regression is not to form a global representation of the wing rock uncertainty. Instead, the data management of section 4.4.3.3 ensures that once a yet unvisited region in the state space is visited, the uncertainty approximation is updated with the new information. In the same way, the data management algorithm prevents the storage of every single data point and therefore ensures that each data point increases the information content in the history stack. Note that if the basis of the uncertainty in equation (5-2) was known, a global representation of the uncertainty could be created. This special case has been demonstrated in section 2.2.1.1 and Figure 2-5.

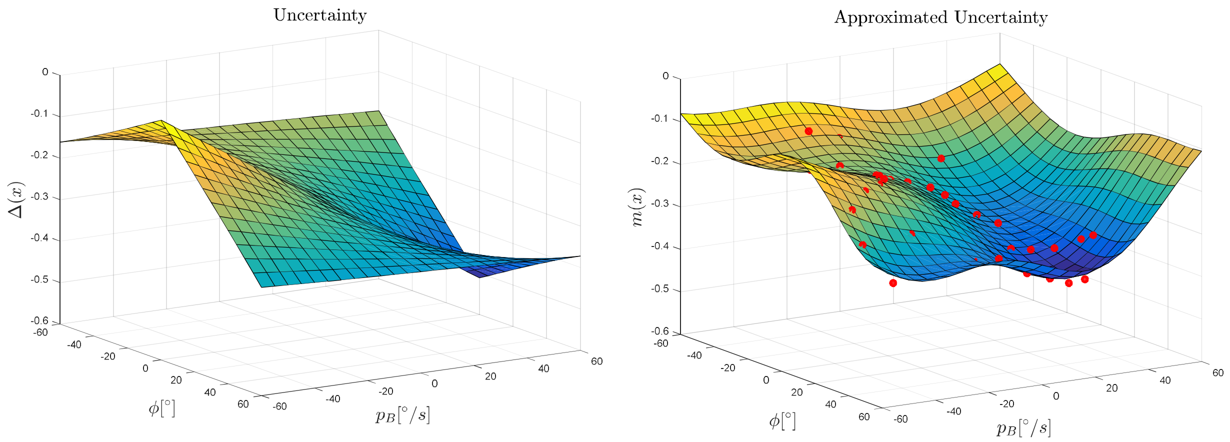


Figure 5-22: Wing rock uncertainty (left) and its approximation after 40 seconds into the simulation (right)

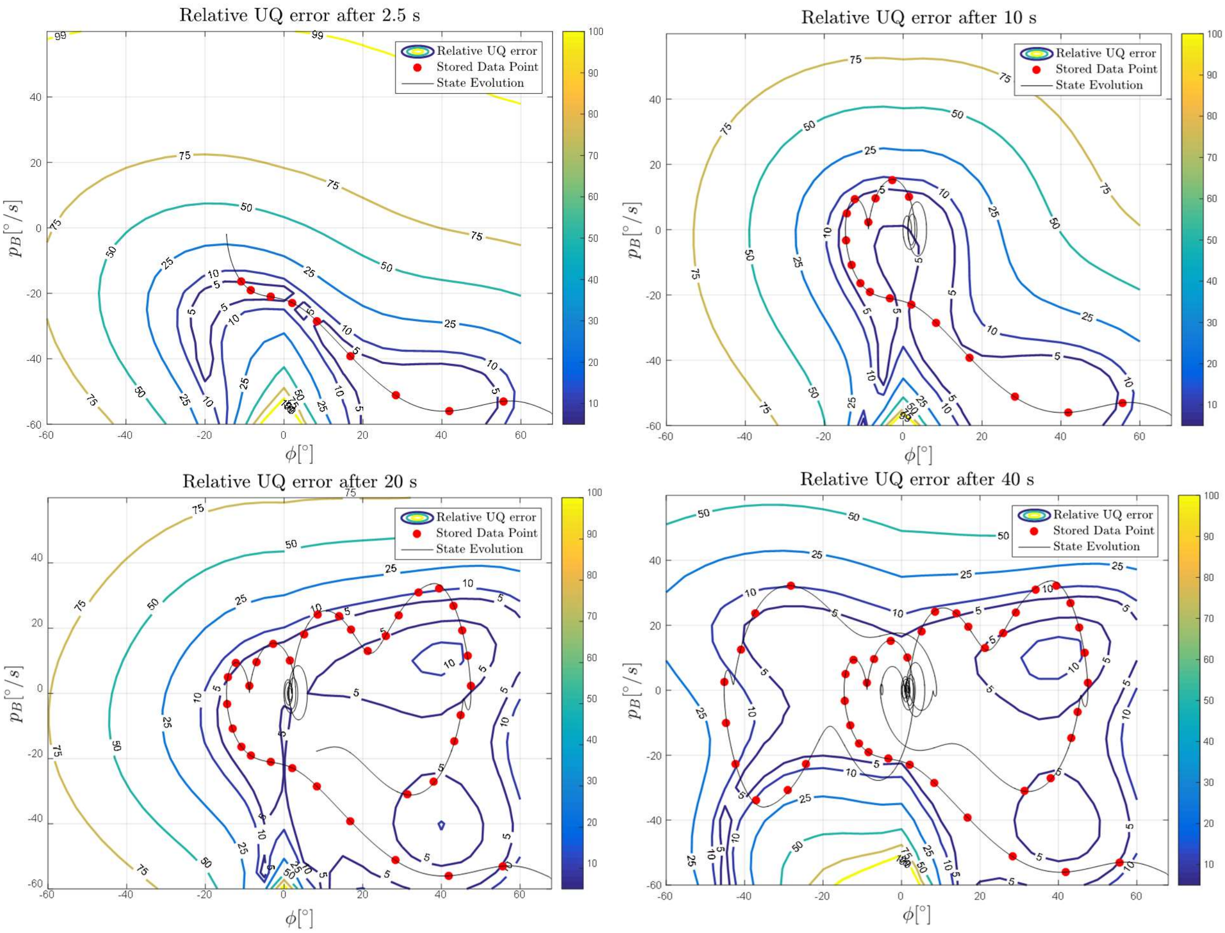


Figure 5-23: Contour plot of the uncertainty approximation error in the wing rock simulation example after 2.5, 10, 20 and 40 seconds

Figure 5-24 shows the posterior standard deviation originating from GP regression. Again, snapshots of the standard deviation are taken after 2.5, 10, 20 and 40 seconds into the simulation. Furthermore, the recorded data points at the specific time instants are depicted. It can be seen that the standard deviation decreases to values below 0.2 within the vicinity of data points. Furthermore, in areas with a high density of stored data points the posterior standard deviation is seen to decrease below 0.1. Section 5.1.2 treated the uncertainty to be known. The modelling uncertainty was modelled as a deterministic quantity. In contrast, GP regression treats the uncertainty as a stochastic variable. As a consequence, the posterior standard deviation will add conservatism to the predictions.

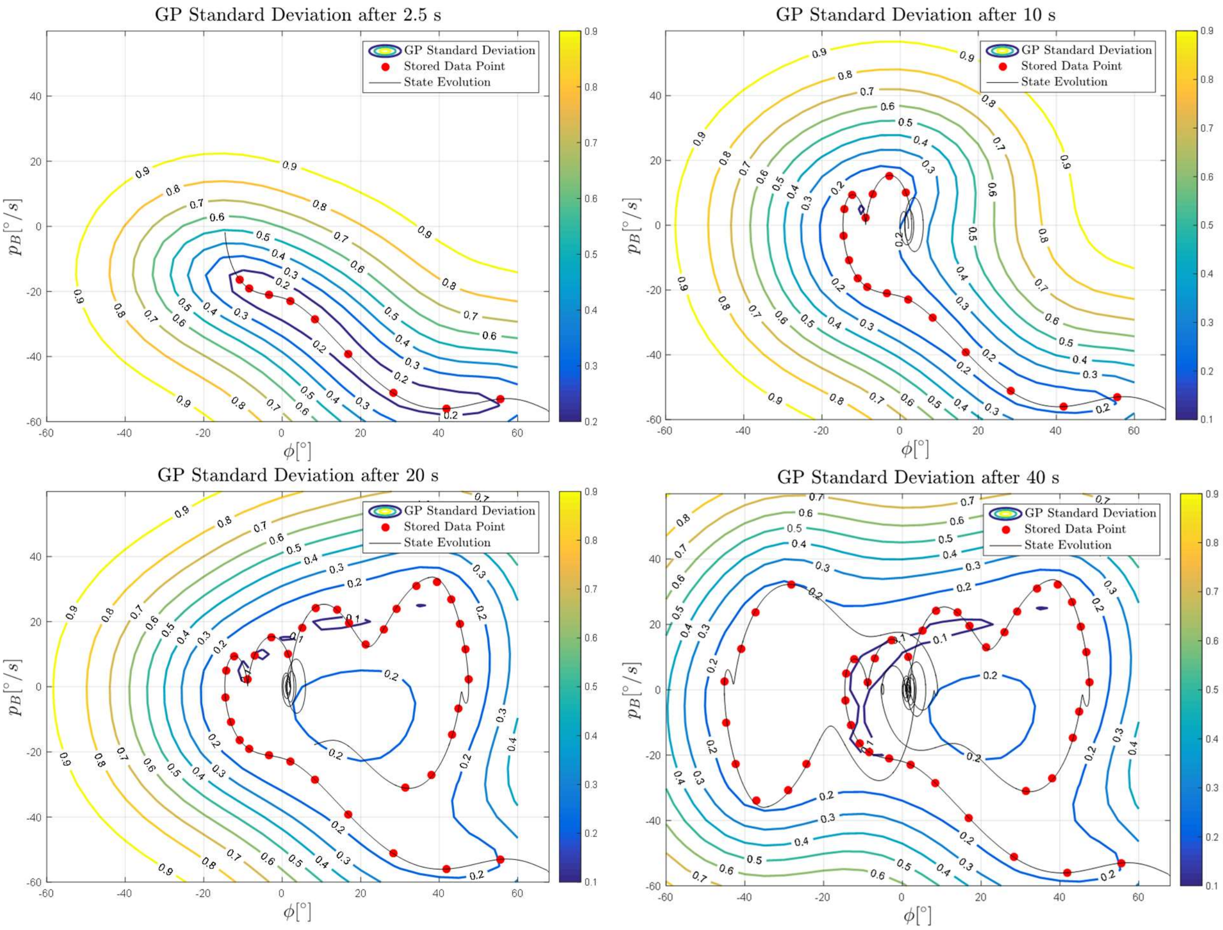


Figure 5-24: Contour plot of the GP standard deviation in the wing rock simulation example after 2.5, 10, 20 and 40 seconds

Figure 5-25 and Figure 5-26 show a highlighted version of the roll angle and roll rate evolution and prediction for the first $15s$ of the simulation using the GP regression based uncertainty model. The predicted mean roll angle matches the simulated one with only minor errors. Similarly, the monitoring algorithm is able to predict the mean roll rate with only little error. Still, the error in the mean increases when compared to the case of the known uncertainty in section 5.1.2. This is due to the fact, that the GP regression itself is not flawless and comprises an error in the estimation of the mean. In contrast, the posterior variance of the GP regression adds more conservativism to the forecast. This can be seen as the 67%, 90% and 99% channels are generally larger than in section 5.1.2. Furthermore, the predicted trajectory channels are seen to not shrink, but converge to a constant width, when compared to section 5.1.2. This can be attributed to the fact, that the prediction inherits an additional variance from the GP posterior model, which is relatively independent on time and only changes by the selected data points. In fact, increasing the density of the data points in the data selection algorithm and thus decreasing the posterior variance of the GP regression can lead to tighter channels in the prediction. As a direct consequence, the predicted channels will be overly conservative once the simulated state trajectories converge to the mean. For completeness, the prediction of the adaptive parameters as well as the full prediction of the states is depicted in Figure H-7-Figure H-9 in Appendix H.5.

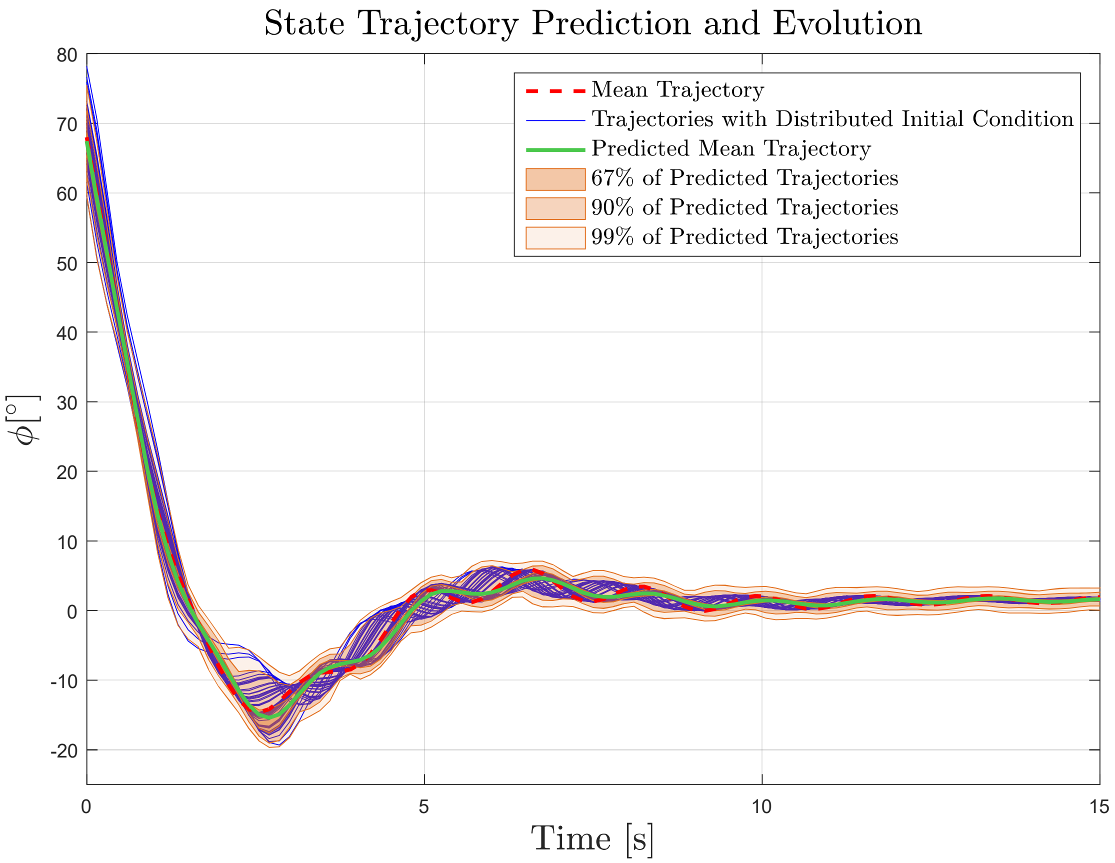


Figure 5-25: Predicted roll angle ϕ over 15 seconds with GP regression based uncertainty estimation

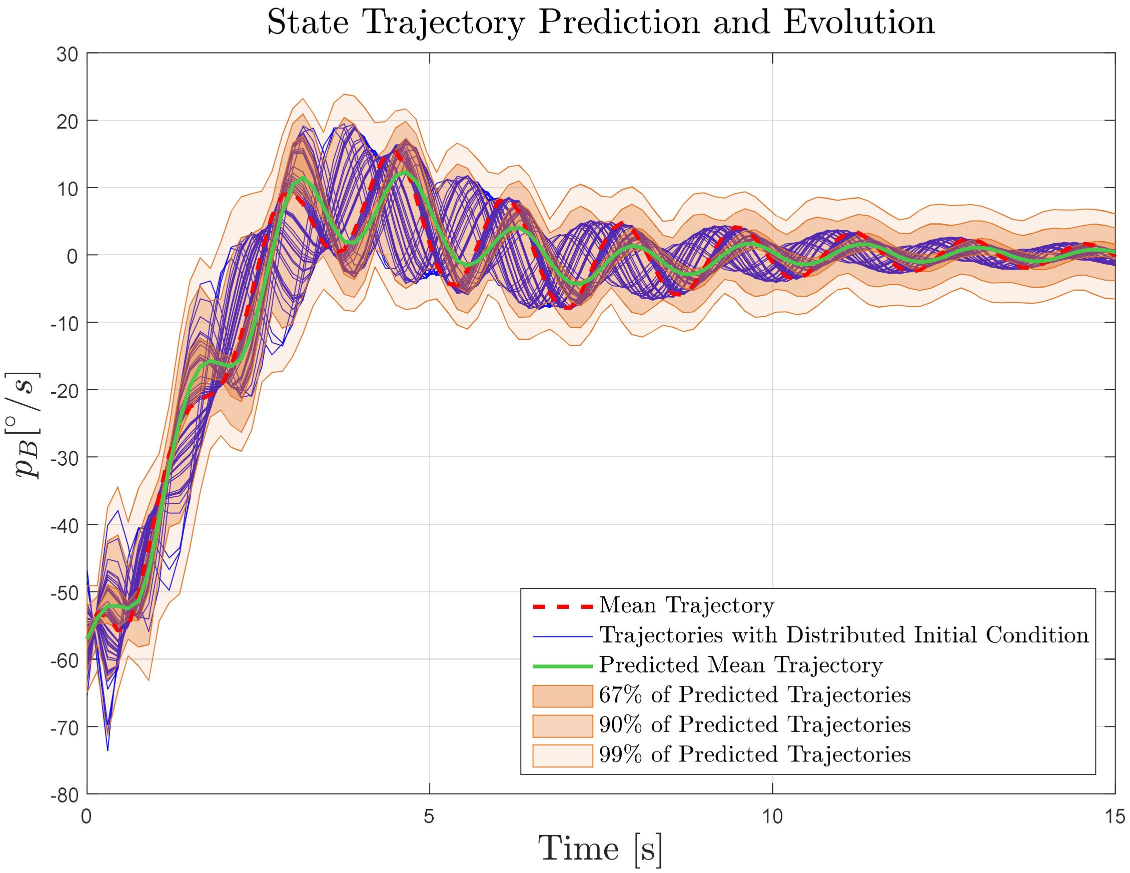


Figure 5-26: Predicted roll rate p_B over 15 seconds with GP regression based uncertainty estimation

	Roll Angle ϕ			Roll Rate p_B		
	$E_{AM}[°]$	$E_{\mathcal{L}_2}[°]$	$E_{\mathcal{L}_\infty}[°]$	$E_{AM}[°/s]$	$E_{\mathcal{L}_2}[°/s]$	$E_{\mathcal{L}_\infty}[°/s]$
GMM $(15s)$	0.45	0.0155	1.69	1.57	0.0521	4.94
Normal $(15s)$	0.76	0.0250	2.72	2.61	0.0840	7.54
GMM $(40s)$	0.30	0.0068	1.69	0.92	0.0215	4.94
Normal $(40s)$	0.42	0.0100	2.72	1.34	0.0330	7.54

Table 5-7: Average mean error, $\mathcal{L}_2$ –like norm and maximum mean error for the prediction of the mean trajectory in the wing-rock simulation example with GP regression based uncertainty estimation

Table 5-7 shows the evaluation of the metrics E_{AM}, $E_{\mathcal{L}_2}$ and $E_{\mathcal{L}_\infty}$ for both scenarios, the GMM forecast as well as for the prediction with a normal distribution only. The metrics are evaluated after $15s$ and $40s$. Again, it can be seen that the application of the GMM compared to a single normal distribution reduces each error metric by approximately $30\% - 40\%$ for each plant state. When compared to the results of section 5.1.2, the error values in general increased. This can be attributed to the fact that the uncertainty is no longer known, but needs to be approximated from gathered data. Furthermore, in contrast to the results of section 5.1.2, the average mean error decreases as the simulation progresses in time. This leads to the suggestion that the GP regression based uncertainty approximation adversely influences the error in the beginning of the simulation in particular. Still, the error levels for the average mean error as well as the maximum error appear to be acceptable in most applications. Again, in order to assess the risk or probability of a requirement violation, the full distribution needs to be taken into account.

To assess the prediction performance and potential underestimation of the monitor, again the GoC of section 5.1.2 is used. Again, both cases of using a GMM or a simple normal distribution for the forecast are compared. Figure 5-27 shows the results for both plant states and varying numerical values for the relaxation term Δx. In conjunction with Figure 5-27, Table 5-8 lists the numerical values of the GoC at various Δx. It can be seen that without relaxation term both, the GMM and the normal distribution scenario, achieve higher GoC values when compared to the simulation in section 5.1.2. This can be attributed to the property of the GP regression, which increases the conservatism of the prediction due to the posterior predictive variance. Still, the general results of the GoC evaluation are similar to the simulation case in section 5.1.2. The introduction of the relaxation term with only small values $\Delta\phi = 0.5°, \Delta p_B = 0.5°/s$ quickly leads the GoC values for the GMM to approach one. In contrast, the GoC of the Gaussian approximation increases only slightly and in most cases does not achieve a GoC of one within the selected parameter space. This indicates that the GMM encompasses the simulated state trajectory channels well.

In order to evaluate if a potentially undesired situation occurs within the prediction horizon, the indicator functions and the monitoring output from section 4.5.3 are computed. For the sake of highlighting the concept, the state requirements are set to be equal to the ones in section 5.1.2. The roll angle shall not exceed a limit of $\phi_{req,high} = 30°$ and shall not fall below $\phi_{req,low} = -15°$. Similarly, the roll rate shall not exceed $p_{B,high} = 60\frac{°}{s}$ and shall not fall below $p_{B,high} = -30\frac{°}{s}$.

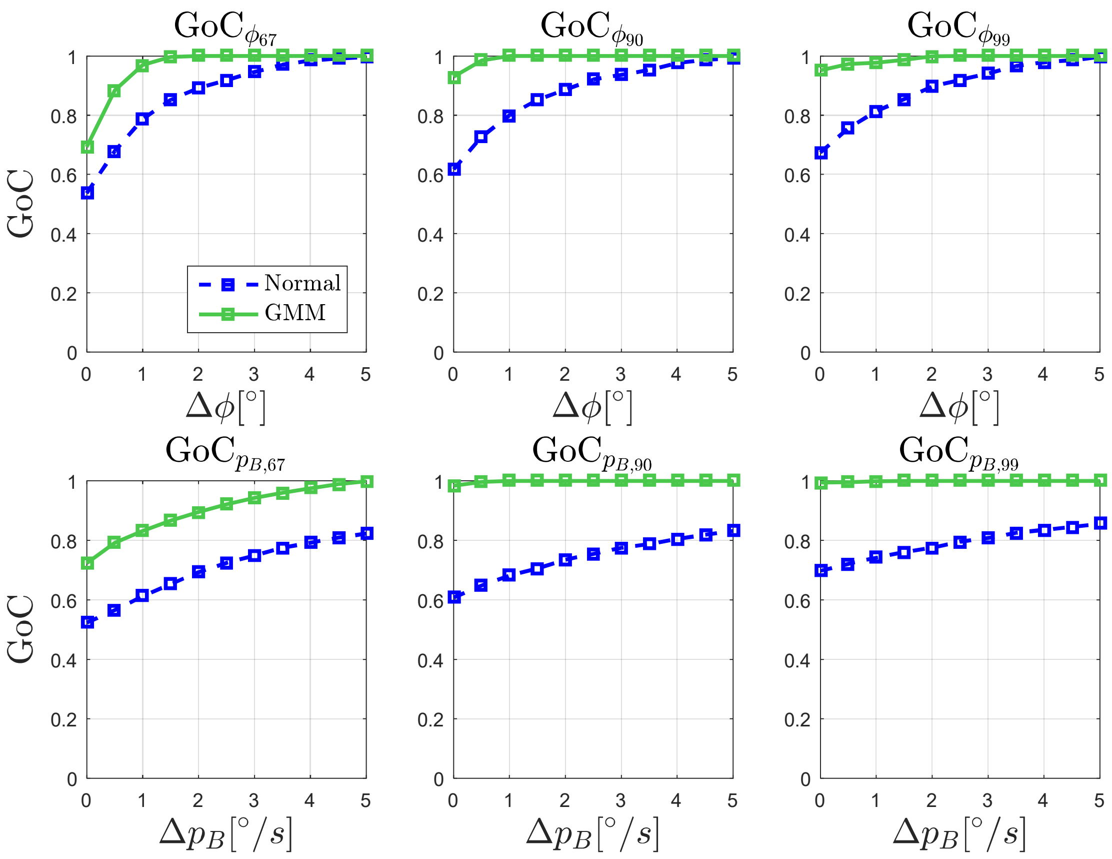


Figure 5-27: GoC of the plant states with relaxation term for the wing-rock example with GP regression

Forecast Basis	$GoC_{x_{67}}$	$GoC_{x_{90}}$	$GoC_{x_{99}}$
$\boldsymbol{\Delta\phi = 0°, \Delta p_B = 0°/s}$			
ϕ – GMM	0.693	0.929	0.951
ϕ – Normal	0.536	0.618	0.674
p_B – GMM	0.723	0.983	0.994
p_B – Normal	0.523	0.608	0.698
$\boldsymbol{\Delta\phi = 0.5°, \Delta p_B = 0.5°/s}$			
ϕ – GMM	0.884	0.985	0.973
ϕ – Normal	0.679	0.727	0.756
p_B – GMM	0.792	0.997	0.996
p_B – Normal	0.565	0.648	0.721
$\boldsymbol{\Delta\phi = 1°, \Delta p_B = 1°/s}$			
ϕ – GMM	0.969	1.0	0.978
ϕ – Normal	0.787	0.800	0.814
p_B – GMM	0.832	1.0	0.999
p_B – Normal	0.612	0.682	0.742
$\boldsymbol{\Delta\phi = 2°, \Delta p_B = 2°/s}$			
ϕ – GMM	1.0	1.0	1.0
ϕ – Normal	0.892	0.886	0.897
p_B – GMM	0.895	1.0	1.0
p_B – Normal	0.693	0.735	0.775

Table 5-8: GoC in the wing-rock simulation example with GP regression based uncertainty approximation

Figure 5-28 depicts the predicted probability of a roll angle requirement violation according to equation (4-102) within $15s$ of the simulation. The results only deviate slightly from the simulation with known nonlinearity in section 5.1.2. I.e. from an initial requirement violation, the monitor correctly predicts an imminent exit of the predicted distribution from the channel formed by the upper and lower state requirement before eventually returning into it. Also the predicted probability of an imminent requirement violation only deviates slightly from the results of section 5.1.2. The same assessment can be made when comparing the predicted probability of a roll rate requirement violation shown in Figure 5-29 with its counterpart in section 5.1.2. This leads to the hypothesis that even though the approximation of the uncertainty by GP regression introduces errors into the system, the final result is only influenced in a minor way. While this seems to be true in this case, it is not universal. In particular, if the state requirements are selected to be tighter, the additional variance from the uncertainty approximation may lead to higher probabilities of imminent requirement violation and therefore more frequent warnings by the monitor.

Figure 5-30 shows the resulting OVL curve. The general progress of the curve is seen to be similar to the results in section 5.1.2. However, the OVL is seen to decrease to a slightly higher value of approximately 0.65 within $5s$ into the simulation. This can be attributed to the fact, that the wingrock uncertainty is not continuous. In the first simulation case, the full GMM was propagated through this non-continuous function, leading to propagation errors. In contrast, the GP model is continuous. Since the OVL is only calculated from the models themselves and not the true system, lesser propagation errors are made when the GMM is propagated through the GP. This in turn leads to larger values for the OVL.

Finally, Figure 5-31 shows the predicted probability of a requirement violation P_{mod}, which is modified by the OVL according to (4-109). The resulting curves again resemble the results of section 5.1.2. The biggest difference is the fact that the modified probability P_{mod} converges to a slightly lower value of approximately 0.35 instead of 0.40. This again can be attributed to the fact that the GP model is continuous and the arguments made above. Note, that tighter state requirements also effect the modified probability P_{mod} and can lead to a drastic increase in issued warnings by the monitor.

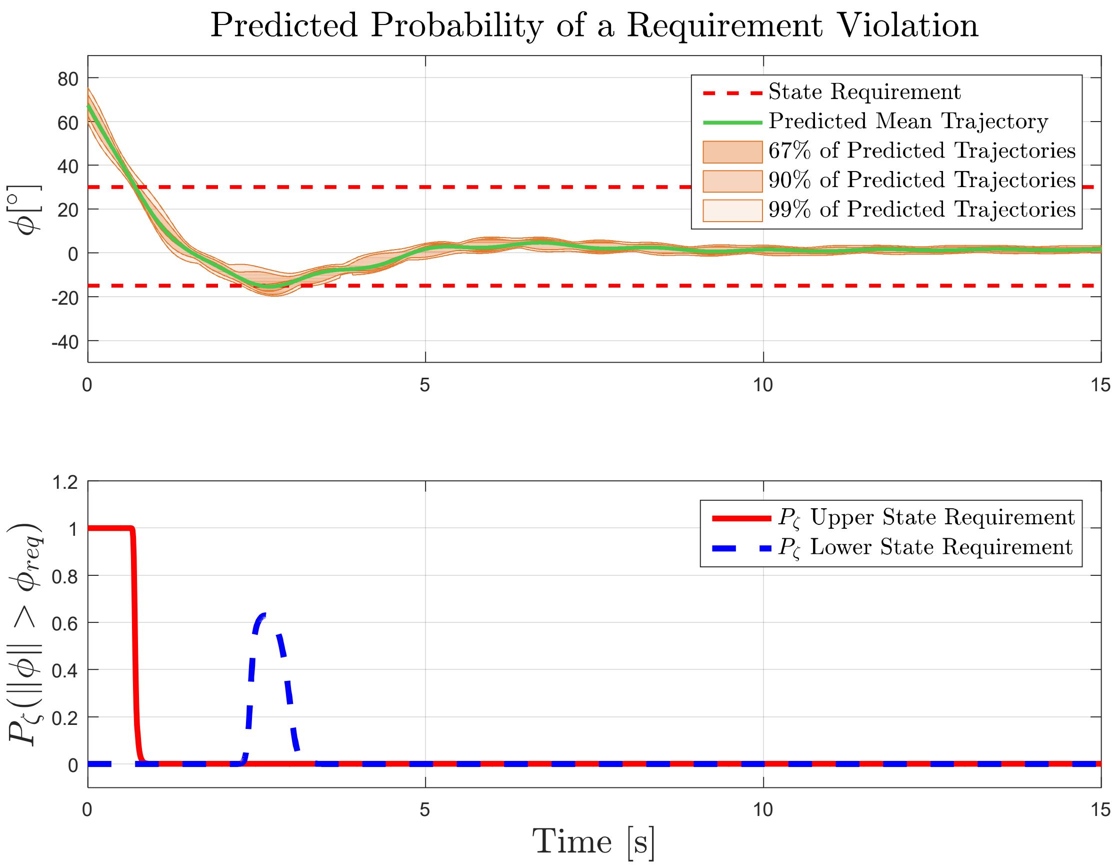


Figure 5-28: Probability of a roll angle requirement violation in the wing-rock example with GP regression

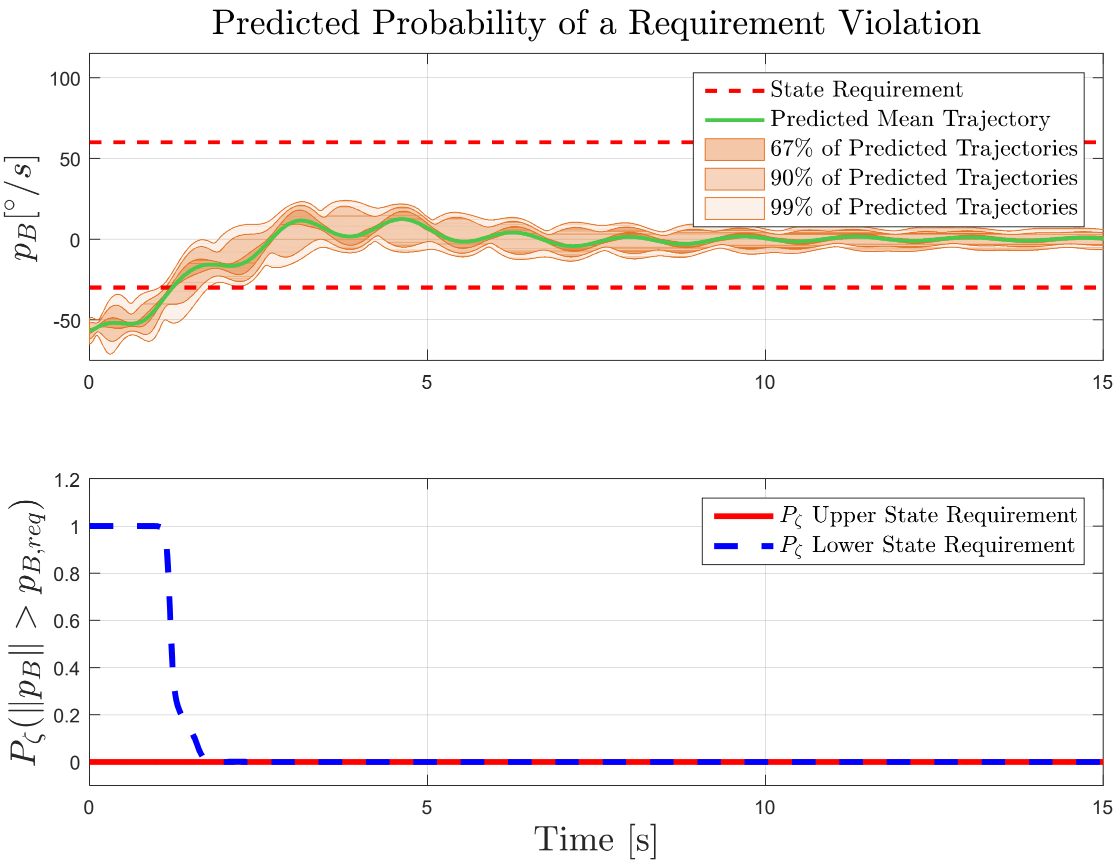


Figure 5-29: Probability of a roll rate requirement violation in the wing-rock example with GP regression

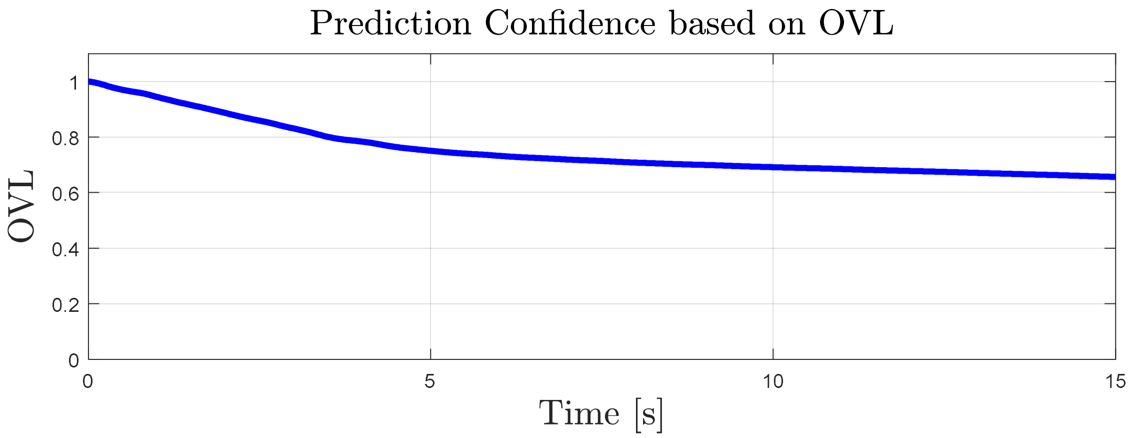


Figure 5-30: Predicted Confidence based on OVL in the wing rock example with GP regression based uncertainty approximation

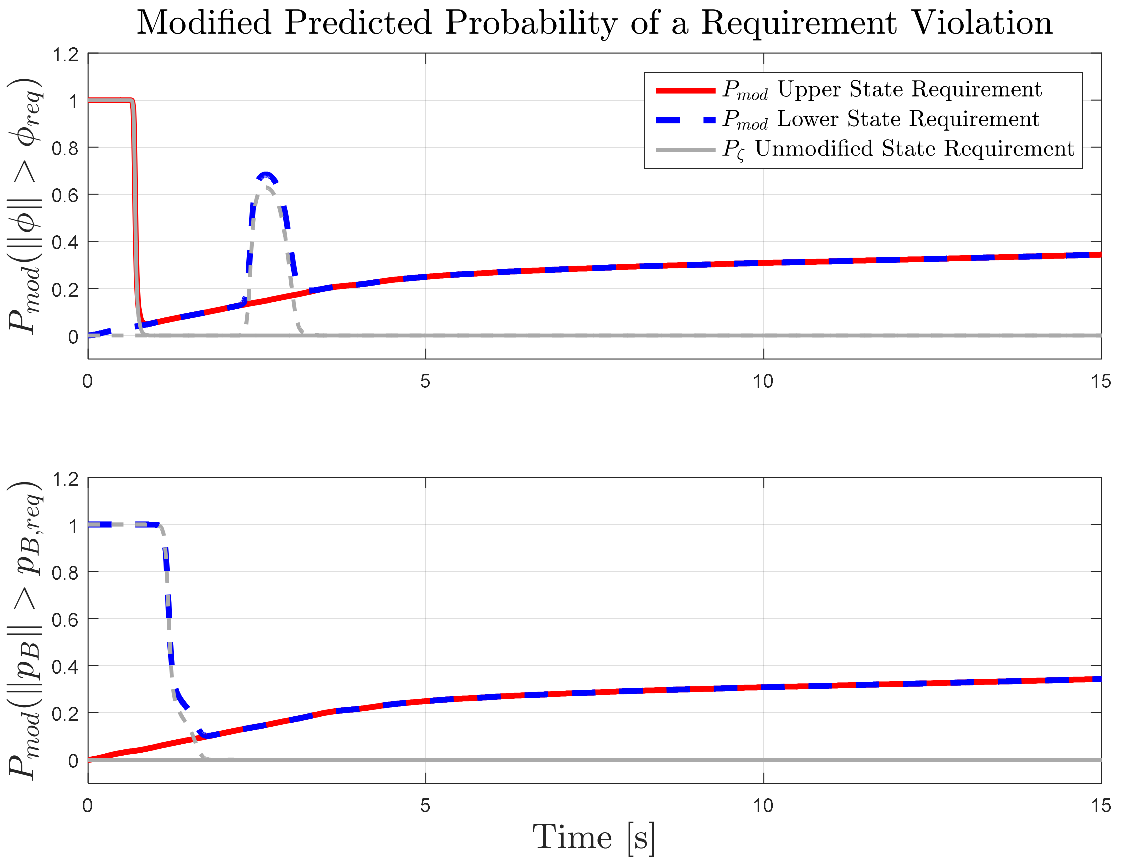


Figure 5-31: Modified predicted probability of a requirement violation in the wing rock example with GP regression based uncertainty approximation

5.2 Short Period with predictor based MRAC

This section adopts the simulation example of [128] in order to corroborate the monitoring algorithm in the case of predictor based MRAC. The simulation model represents the short period of a F-16 fighter aircraft:

$$\begin{bmatrix}\dot{\alpha}_K \\ \dot{q}_K\end{bmatrix} = \begin{bmatrix}\frac{Z_\alpha}{V} & 1+\frac{Z_q}{V} \\ M_\alpha & M_q\end{bmatrix}\begin{bmatrix}\alpha_K \\ q_K\end{bmatrix} + \begin{bmatrix}\frac{Z_\eta}{V} \\ M_\eta\end{bmatrix}\eta_C \tag{5-20}$$

Here, α_K represents the angle of attack, q_K represents the roll rate and η is the control input representing the elevator deflection. Furthermore, V represents the trimmed airspeed, Z_α, Z_q and Z_η represent the partial derivatives of the aerodynamic vertical force Z and M_α, M_q and M_η represent the partial derivatives of the pitching moment M with respect to the states and the control input. The numerical values for the simulation model are adopted from [200]. The goal of the controller is to track a pitch rate command q_{cmd}, which acts as the exogenous input to the system. In [128], Lavretsky constructs an extended open-loop dynamical model by adding integrated pitch rate tracking error dynamics $\dot{e}_I = q - q_{cmd}$ to the system in (5-20):

$$\begin{bmatrix}\dot{e}_I \\ \dot{\alpha}_K \\ \dot{q}_K\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 0 & 1 \\ 0 & \frac{Z_\alpha}{V} & 1+\frac{Z_q}{V} \\ 0 & M_\alpha & M_q\end{bmatrix}}_{\boldsymbol{A}_P}\begin{bmatrix}e_I \\ \alpha_K \\ q_K\end{bmatrix} + \underbrace{\begin{bmatrix}0 \\ \frac{Z_\eta}{V} \\ M_\eta\end{bmatrix}}_{\boldsymbol{B}_P}\left(\eta_C - \Delta(\alpha, q)\right) + \underbrace{\begin{bmatrix}-1 \\ 0 \\ 0\end{bmatrix}}_{\boldsymbol{B}_{rm}} q_{cmd}. \tag{5-21}$$

The numerical values for the parameters in (5-21) are taken from [200] and [128]. The uncertainty $\Delta(\alpha_K, q_K)$ is partly adopted from [128]:

$$\Delta(\alpha_K, q_K) = \Theta_1^*\alpha_K + \Theta_2^* q_K + \Theta_3^* e^{-\frac{\left(\alpha_K - 2°\frac{\pi}{180}\right)^2}{2(0.0233)^2}}. \tag{5-22}$$

Note, that the input to (5-22) is assumed to be in rad or $\frac{rad}{s}$. The control architecture of section 3.2 with the control law (3-11) such that $\eta_C = \eta_{bl} + \eta_{ad}$ is employed. Note, that the control design deviates from [128] as this thesis focusses on tracking control for a baseline design. Therefore consider the following reference model:

$$\begin{bmatrix}\dot{e}_{I,rm} \\ \dot{\alpha}_{rm} \\ \dot{q}_{rm}\end{bmatrix} = \boldsymbol{A}_{rm}\begin{bmatrix}e_{I,rm} \\ \alpha_{rm} \\ q_{rm}\end{bmatrix} + \boldsymbol{B}_{rm} q_{cmd}. \tag{5-23}$$

Consequently, the baseline control η_{bl} signal matches the one in (3-12). The adaptive controller η_{ad} uses the structure of (3-15), where the basis function $\boldsymbol{\varphi}$ is given by the basis of the uncertainty in (5-22):

$$\boldsymbol{\varphi} = \begin{bmatrix}\alpha & q & e^{-\frac{\left(\alpha - 2°\frac{\pi}{180}\right)^2}{2(0.0233)^2}}\end{bmatrix}^T. \tag{5-24}$$

Based on (3-33), the predictor dynamics are given by

$$\begin{bmatrix} \dot{\hat{e}}_I \\ \dot{\hat{\alpha}} \\ \dot{\hat{q}} \end{bmatrix} = \boldsymbol{A}_{prd} \left(\begin{bmatrix} \dot{\hat{e}}_I \\ \dot{\hat{\alpha}} \\ \dot{\hat{q}} \end{bmatrix} - \begin{bmatrix} \dot{e}_I \\ \dot{\alpha} \\ \dot{q} \end{bmatrix} \right) + \boldsymbol{A}_{rm} \begin{bmatrix} \dot{e}_I \\ \dot{\alpha} \\ \dot{q} \end{bmatrix} + \boldsymbol{B}_{rm} q_{cmd}. \tag{5-25}$$

The update law of (3-38) uses a σ-modification with $k(\mathbf{\Theta}) = -\sigma\mathbf{\Theta},$ such that

$$\dot{\mathbf{\Theta}} = -\mathbf{\Gamma}\boldsymbol{\varphi}(\boldsymbol{x}_P)(\boldsymbol{e}^T\boldsymbol{P} - \boldsymbol{e}_P^T\boldsymbol{P}_P)\boldsymbol{B}_P - \mathbf{\Gamma}\sigma\mathbf{\Theta}. \tag{5-26}$$

The residual predictor error is given by

$$\boldsymbol{e}_P = \begin{bmatrix} \hat{e}_I \\ \hat{\alpha} \\ \hat{q} \end{bmatrix} - \begin{bmatrix} e_I \\ \alpha \\ q \end{bmatrix}. \tag{5-27}$$

The simulation runs for a total of $30s$ with a time step of $0.01s$. The exogenous reference input commands a pitch rate doublet, with $5\frac{\circ}{s}$ between seconds 1 and 11 and $-5\frac{\circ}{s}$ between seconds 11 to 21, respectively. Lavretsky commands a second doublet in [128], which is omitted here, since the results match the first doublet. The plant states are initialized at $\alpha_K(t_0) = 0^\circ$ and $q_K(t_0) = 0\frac{\circ}{s}$. The reference model states, the predictor states as well as the adaptive weights are initialized at zero. Table 5-2 summarizes the relevant simulation parameters for the plant and the controller as well as the respective initial conditions for a concise overview.

In order to form a distribution of simulation trajectories and thus a comparable baseline for the monitoring algorithm, the initial conditions of the plant states are independently, normally distributed with $\alpha_K(t_0) \sim N(0^\circ, (1^\circ)^2)$ and $q_K(t_0) \sim N\left(0\frac{\circ}{s}, \left(2\frac{\circ}{s}\right)^2\right)$. The simulation is repeated 2500 times. The states of the reference model as well as the adaptive weights are still initialized at zero.

The online GP regression algorithm of section 4.4.3 with SE kernel functions is applied in order to form a belief on the uncertainty during the run time of the monitor. The data storage algorithm checks if new data points are sufficiently different to already stored data by employing equation (4-93). Other than that, the history stack is fixed and does not discard already stored data points. The bandwidth of the kernels is selected equally to be $\Lambda_i = 0.25$.

The GMM used in this simulation example consists of 99 mixture elements. The initial standard deviation of each mixture element is selected to be $\sigma_{i,0} = 0.03$. The mixture elements are equidistantly distributed in the range of 4σ of the initial state distribution $\alpha_K(t_0) \sim N\left(0^\circ, \left(1\frac{\circ}{s}\right)^2\right)$ and $q_K(t_0) \sim N\left(0\frac{\circ}{s}, \left(2\frac{\circ}{s}\right)^2\right)$. By solving the quadratic program in (2-5), the GMM is fitted to the initial state distribution of the simulation.

The prediction horizon is set to $t_{pred} = 29.99s$ with a time step of $\Delta t_{pred} = 0.01s$. This results in $k_{end} = 2999$ prediction steps. Note, that the initial time is not included in the prediction horizon. Hence, predicting for $29.99s$ results in a prediction to the end of the simulation. The system dynamics (5-21), (5-25) and (5-26) are according to (4-30). The GMM is propagated through the linearized surrogate dynamics and the estimated uncertainty according to the update laws in (4-82) and (4-90).

Simulation Parameter	Numerical Value
General Simulation	
t_0	$0s$
dt	$0.01s$
t_{end}	$30s$
$\boldsymbol{A}_P$	$\begin{bmatrix} 0 & 0 & 1 \\ 0 & -1.0189 & 0.9051 \\ 0 & 0.8223 & -1.0774 \end{bmatrix}$
$\boldsymbol{B}_P$	$\begin{bmatrix} 0 \\ -0.0022 \\ -0.1756 \end{bmatrix}$
$[\alpha_K(t_0) \quad q_K(t_0)]$	$\left[0° \quad 0\frac{°}{s}\right]$
$\boldsymbol{\Theta}^*$	$[4.6839 \quad 6.1373 \quad 0.5]$
Baseline Controller	
$\boldsymbol{A}_{rm} = \boldsymbol{A}_e$	$\begin{bmatrix} 0 & 0 & 1 \\ -0.0311 & -1.0262 & 0.8850 \\ -2.4834 & 0.2370 & -2.6800 \end{bmatrix}$
$\boldsymbol{B}_{rm}$	$\begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}$
$[e_{I,rm}(t_0) \quad \alpha_{K,rm}(t_0) \quad q_{K,rm}(t_0)]$	$\left[0 \quad 0° \quad 0\frac{°}{s}\right]$
$\boldsymbol{K}_e^T$	$[-14.1421 \quad -3.3331 \quad -9.1263]$
Adaptive Controller	
$\boldsymbol{A}_{prd}$	$\begin{bmatrix} 0 & 0 & 3 \\ -0.0933 & -3.0786 & 2.6550 \\ -7.4502 & 0.7110 & -8.0400 \end{bmatrix}$
$\boldsymbol{\Gamma}$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 400 & 0 & 0 \\ 0 & 0 & 400 & 0 \\ 0 & 0 & 0 & 20 \end{bmatrix}$
$\boldsymbol{Q} \mid \boldsymbol{P}$	$\begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 800 \end{bmatrix} \mid \begin{bmatrix} 389.75 & -14.84 & 0.21 \\ -14.84 & 1.90 & 6.13 \\ 0.21 & 6.13 & 151.36 \end{bmatrix}$
$\boldsymbol{P}_{prd}$	$\begin{bmatrix} 1169.3 & -44.5235 & 0.6180 \\ -44.5235 & 5.7079 & 18.3857 \\ 0.6180 & 18.3857 & 454.0632 \end{bmatrix}$
σ	0.1
$\boldsymbol{\Theta}^*(t_0)$	$[0 \quad 0 \quad 0]$

Table 5-9: List of parameters used in the PMRAC simulation example

Simulation Parameter	Numerical Value
Prediction	
t_{pred}	$29.99s$
dt_{pred}	$0.01s$
GMM	
m (Number of mixture elements)	99
$\sigma_{i,0}$	0.03
Initial distribution of the mixture elements μ_i	Equidistant within 4σ of the initial state distribution
GP Regression	
p_H	100
σ_n^2	0.05
γ	0.0005
Λ_i	0.25
Indicator Function	
δ_ζ	0.7
α_{ind}	1
β_{ind}	3000
δ_{ind}	20
$\alpha_{K,low}$	$-5°$
$\alpha_{K,high}$	$5°$
$q_{B,low}$	$-7°/s$
$q_{B,high}$	$7°/s$

Table 5-10: List of monitoring parameters used in the PMRAC simulation example

For the purpose of the simulation example, the state limits are arbitrarily selected. The angle of attack shall not exceed a limit of $\alpha_{K,high} = 5°$ and shall not fall below $\alpha_{K,low} = -5°$. Similarly, the pitch rate shall not exceed $q_{K,high} = 7\frac{\circ}{s}$ and shall not fall below $q_{K,high} = -7\frac{\circ}{s}$. As a baseline, the parameters of the wing-rock example of section 5.1 are used for the indicator function. Since the simulation time is smaller, β_{ind} is chosen to be $\beta_{ind} = 3000$. Table 5-10 summarizes all parameters of the monitoring algorithm.

Figure 5-32 shows the reference command, the state of the reference model as well as the evolution of the state trajectories over the simulation horizon. It can be seen that despite distributed initial conditions, the plant state trajectories stay bounded. This suggests that the PMRAC based adaptive controller is able to stabilize the system and track the reference model under various conditions. After 15 seconds the distributed state trajectories are seen to be nearly indistinguishable from the mean. This can be attributed to the fact that the adaptive controller is able to cancel the uncertainty pointwise in time. This in turn allows the baseline controller to reduce the tracking error.

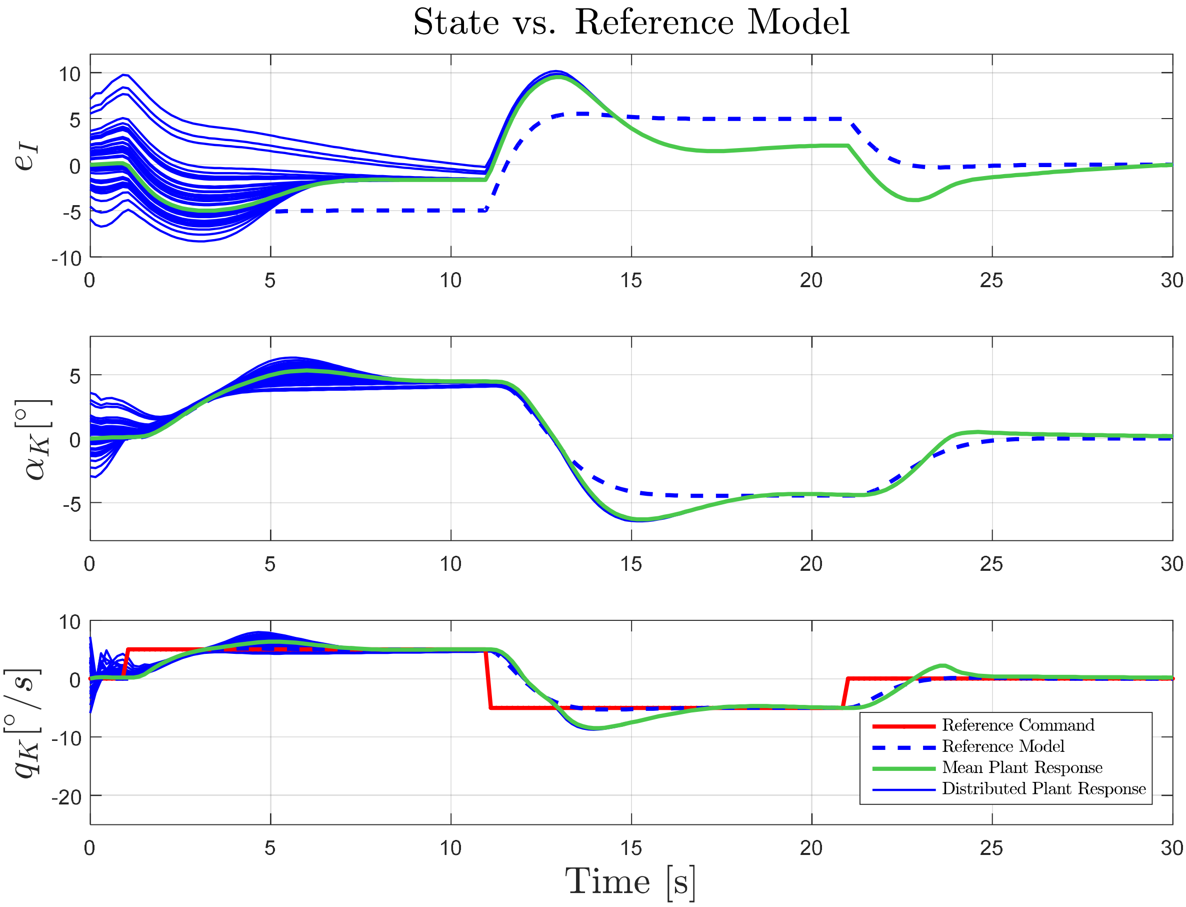


Figure 5-32: Tracking performance of PMRAC for the short period example with normally distributed initial conditions

Figure 5-33, Figure 5-34 and Figure 5-35 respectively show the integral error, the angle of attach and the pitch rate trajectory distribution as a histogram at selected time instants, namely after $0.2s$, $1s$, $5s$ and at $7.5s$. Furthermore, the respective approximating normal distribution is depicted. The later originates from taking the mean and variance of the state trajectory distribution at the mentioned time instants. It indicates the result if the data is assumed to be normally distributed.

It can be seen after $0.2s$ into the simulation, the distribution of the integral error e_I as well as the angle of attack α_K only slightly differs from a normal distribution. In contrast, the pitch rate is seen to be heavily skewed towards the right as well as truncated at approximately $0.3\ \frac{°}{s}$. This can be attributed to the nonlinear dynamics of the system. In fact, the nonlinear uncertainty primarily affects the pitch rate dynamics. The reason for this is twofold. First, since $|M_\eta| \gg |Z_\eta|$, the effect of the uncertainty on the dynamics of α_K is sufficiently smaller than its effect on q_K. Even though the dynamics of α_K depend on the pitch rate q_K, a deterioration from the normal distribution takes time. Secondly, the dynamics of the integral error are only dependent on the pitch rate q_K. Comparable to the angle of attack, a deterioration from the initial normal distribution requires time.

From $0.2s$ to $1s$ the angle of attack as well as the pitch rate distribution become heavily skewed to the left, while the distribution of the integral error trajectories remains comparable to a Gaussian. This indicates that significant errors could be made by neglecting the nonlinear nature of the system during the prediction.

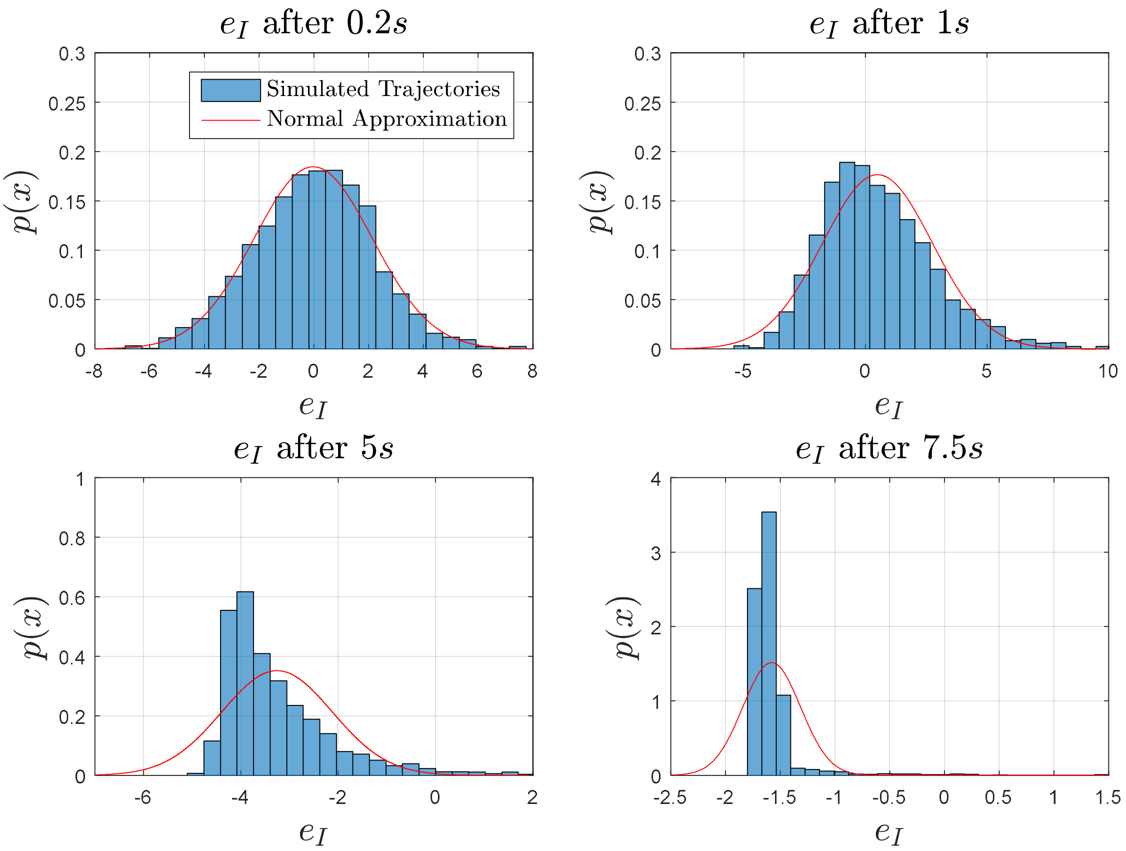


Figure 5-33: Distribution of the integral error $e_I$in the short period example after 0.2s, 1s, 5.0s and 7.5s

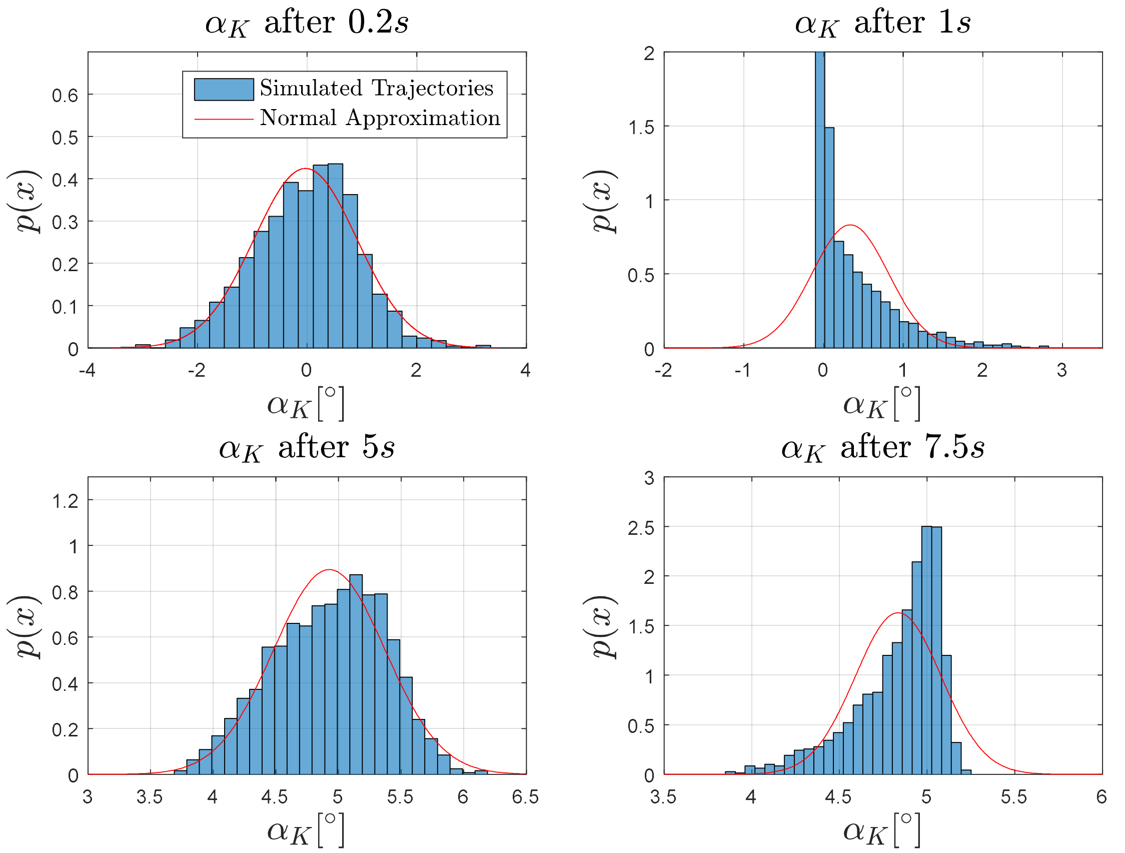


Figure 5-34: Distribution of the angle of attach $\alpha_K$in the short period example after 0.2s, 1s, 5.0s and 7.5s

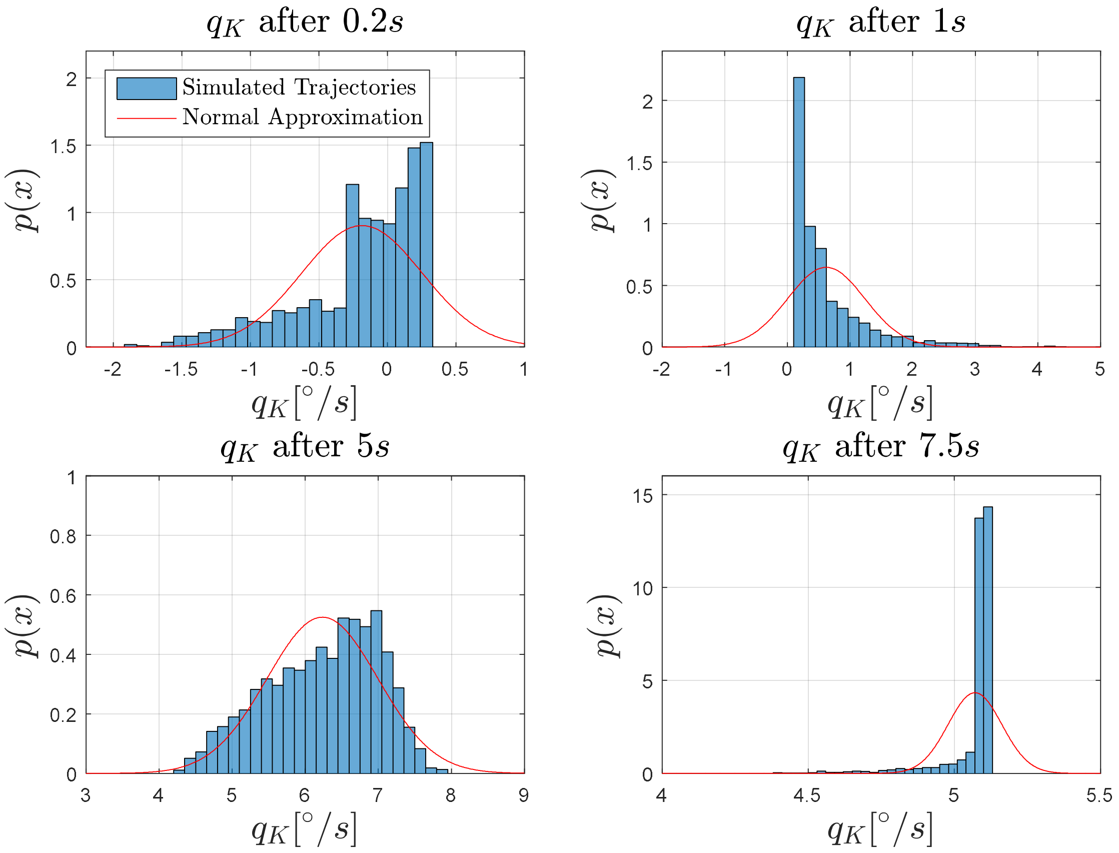


Figure 5-35: Distribution of the pitch rate $q_K$in the short period example after 0.2s, 1s, 5.0s and 7.5s

After $5s$ into the simulation the angle of attack and pitch rate distributions approach a normal distribution again. In contrast, the distribution of the integral error is skewed to the left. In any case, the range of the support decreases by up to 50%. This indicates that the single trajectories are converging to the mean trajectory. Finally, after $7.5s$ all state distributions are heavily skewed. Nevertheless, the range of the support decreased further. In any case, approximating the single state distributions by Gaussians now neglects outliers as well as introduces conservatism. The latter results from the truncation of the state distributions.

Figure 5-36(left) shows the employed uncertainty of equation (5-22). In turn, Figure 5-36 (right) shows the uncertainty approximation after 30 seconds into the simulation using GP regression and the approach depicted in section 4.4.3. The red dots represent recorded data points. After 30 seconds, the data management algorithm recorded a total of 25 data points. The GP is seen to start assuming the general form of the uncertainty, i.e. its general twist. However, from the 3D plot alone it is hard to gauge the quality of the approximation.

Figure 5-37(left) shows a contour diagram of the relative approximation error in percent between the true uncertainty and the GP mean representation. Furthermore, the recorded data points are depicted. It can be seen that the relative approximation error reduces to approximately $0 - 10\%$ in the vicinity of recorded data points. However, in areas without data, the approximation error remains large. This indicates, that the GP approximates the uncertainty well where data points are available. Continuing this line of thought, by increasing the density of the recorded data points and also their distribution through the state space, the subjective approximation quality of the uncertainty in the 3D plot of Figure 5-36 (right) would also increase.

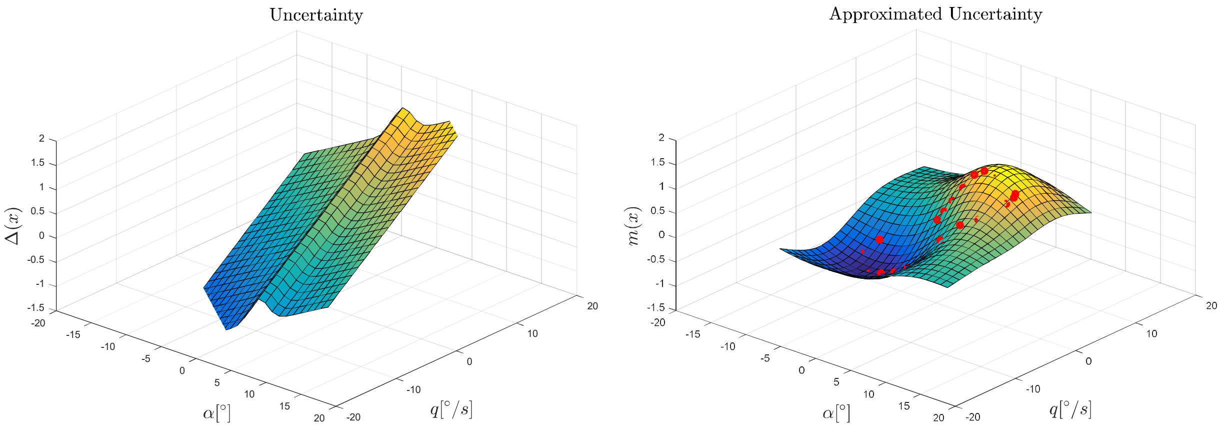


Figure 5-36: Uncertainty (left) and its approximation (right) in the short period simulation example

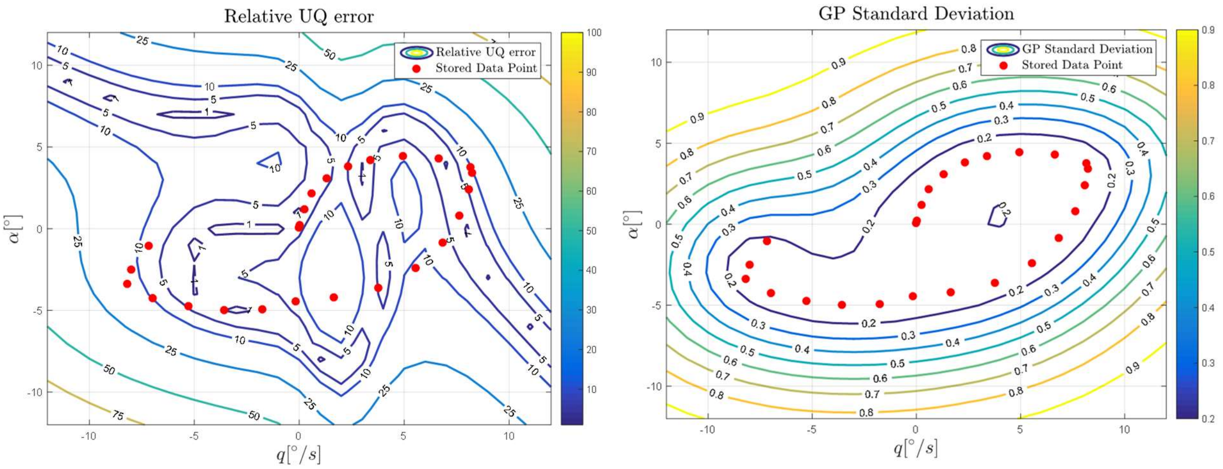


Figure 5-37: Relative UQ error (left) and the approximated posterior standard deviation (right) for the GPR in the short period simulation example

Finally, Figure 5-37(right) shows a contour plot of the posterior standard deviation resulting from the GP regression. Again, the recorded data points are depicted. It can be seen that the standard deviation significantly decreases in areas of the state space where data is available. In fact, increasing the density of the data points further would lead to a further decrease in the posterior standard deviation. In contrast, areas without data exhibit a large posterior standard deviation. The posterior standard deviation is often interpreted to resemble a confidence of the posterior GP model.

Figure 5-38 shows the predicted evolution of the integral error, the angle of attack and the pitch rate. In particular, for each state, it depicts the predicted mean trajectory as well as the area in which 67%, 90% and 99% of all state trajectories are predicted to reside in at each time instant. Furthermore, the plot depicts the mean trajectory as well as the evolution of the state trajectories for distributed initial conditions. It can be seen that the predicted areas contain almost all simulated state trajectories. This indicates, that the GMM based forecast is able to predict the state distribution well, even in times in which the latter is represented by a Gaussian. The predicted area is seen to shrink over time until it is nearly indistinguishable from the mean after approximately 10 seconds. This can be attributed to the fact, that the adaptive controller is able to stabilize the system and achieve at least point-wise in time cancellation of the uncertainty under various initial conditions.

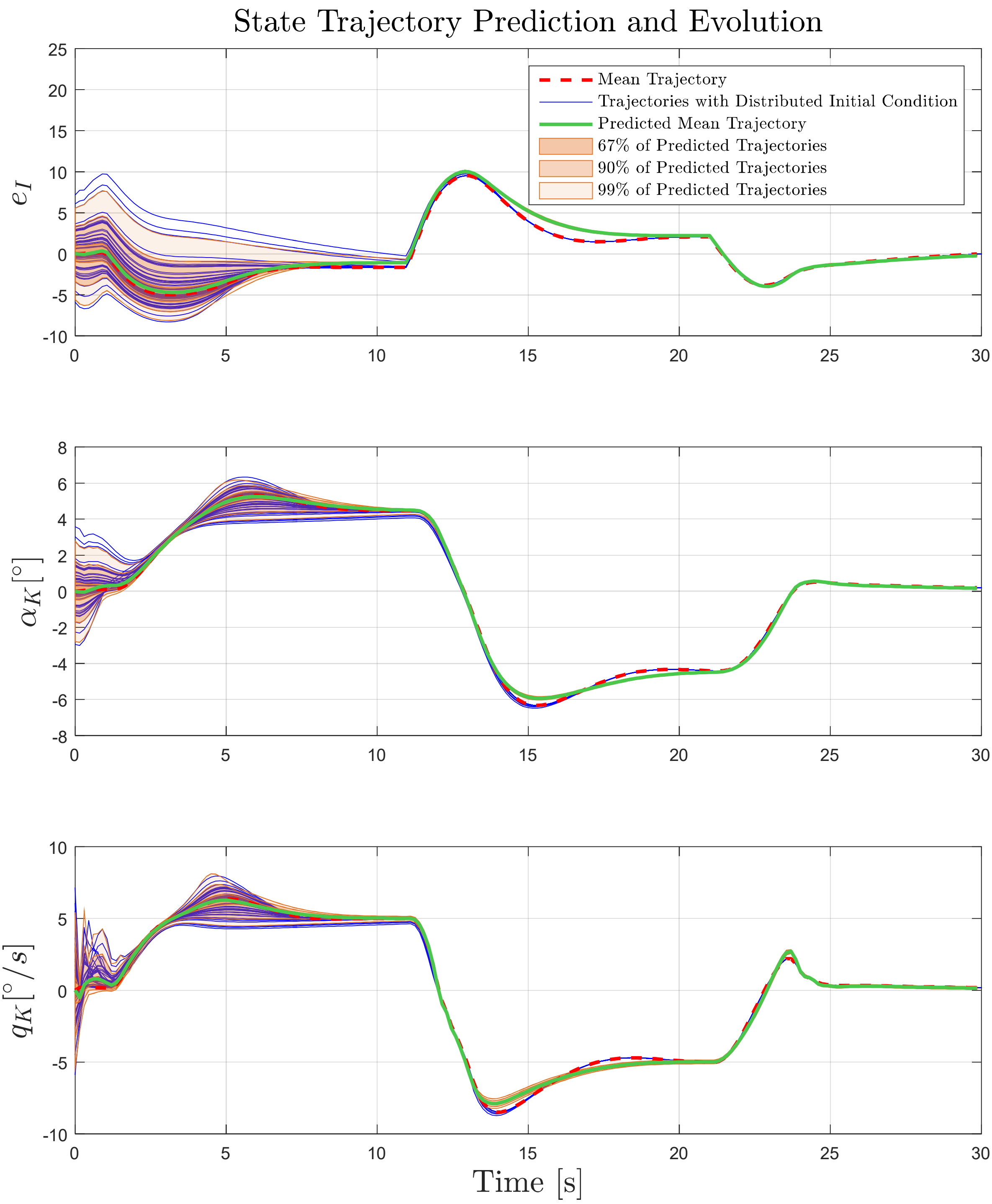


Figure 5-38: Integral error e_I, angle of attach α_K and pitch rate q_K predicted over 30 seconds in the short period simulation example

For the majority of the simulation, the predicted mean is nearly indistinguishable from the simulated one. However, after approximately 12 seconds, minor deviations between the predicted and simulated mean occur. This can be attributed to the uncertainty approximation error resulting from the GP regression algorithm as well as accumulating prediction errors. In any case, the deviations remain small. For the sake of completeness, Figure H-10 and Figure H-11 in Appendix H.6 show the predicted evolution of the adaptive parameters and the predicted predictor states, respectively.

	Integral Error e_I			Angle of Attack α_K			Pitch Rate q_K		
	$E_{AM}[°]$	$E_{\mathcal{L}_2}[°]$	$E_{\mathcal{L}\infty}[°]$	E_{AM} $[°/s]$	$E_{\mathcal{L}_2}$ $[°/s]$	$E_{\mathcal{L}\infty}$ $[°/s]$	E_{AM} $[°/s]$	$E_{\mathcal{L}_2}$ $[°/s]$	$E_{\mathcal{L}\infty}$ $[°/s]$
GMM $(10s)$	0.17	0.0072	0.53	0.058	0.0025	0.16	0.078	0.0033	0.23
Normal $(10s)$	0.17	0.0074	0.55	0.060	0.0026	0.16	0.082	0.0034	0.23
GMM $(30s)$	0.35	0.0097	1.52	0.10	0.0026	0.40	0.15	0.0041	0.70
Normal $(30s)$	0.36	0.0097	1.52	0.10	0.0026	0.40	0.15	0.0041	0.70

Table 5-11: Average mean error, $\mathcal{L}_2$ −like norm and maximum mean error for the prediction of the mean trajectory in the pitch rate simulation example

Table 5-11 shows the evaluation of the metrics E_{AM}, $E_{\mathcal{L}_2}$ and $E_{\mathcal{L}\infty}$ for both scenarios, the GMM forecast as well as for the prediction with a normal distribution only. The metrics are evaluated after $10s$ and $30s$. It can be seen that the application of the GMM compared to a single normal distribution reduces only some error metrics. Furthermore, the reduction is only small. However, employing a GMM also never increased an error metric, thus indicating that a GMM forecast most likely leads to an increased prediction performance. The reason for the small effect of employing a GMM can be found in the approximation error of the GP posterior mean. As the online generated uncertainty model exhibits errors, both, the GMM model as well as the forecast with a normal distribution, are affected similarly. This becomes particularly apparent after approximately $12s$ into the simulation, when the predicted mean deviates from the simulated one (see Figure 5-6). The error metrics only take a deviation from the mean into account. As a result, if the error from the uncertainty approximation dominates the error arising from the different forecast models, and both forecast models are affected by the uncertainty approximation error similarly, then the error metrics tend to be similar as well. Hence, the metrics E_{AM}, $E_{\mathcal{L}_2}$ and $E_{\mathcal{L}\infty}$ alone do not yield a full analysis of the simulation.

In order to interpret the prediction further, again the cdf of the plant states are analysed at different time instants. Figure 5-39, Figure 5-40 and Figure 5-41 respectively show the approximation of the plant state cdf for the integral error, the angle of attack and the pitch rate at different prediction steps, namely after $0.2s, 1s$,$5s$ and $7.5s$. The plots depict the cdf of the simulated trajectories, the predicted cdf using a GMM and the predicted cdf using a Gaussian instead of the GMM. Again, the circles indicate the borders of the channels encompassing the central 67%, 90% and 99% of the state trajectories.

In Figure 5-39 and Figure 5-40 it can be seen that the cdfs of the GMM, the normal and the simulated trajectories are nearly indistinguishable after 0.2 seconds into the simulation. Only minor errors occur for the outermost circles. Furthermore, the cdfs resemble the shape of a normal cdf. This is backed up by the analysis of the state pdf (see Figure 5-33 and Figure 5-34). In contrast, the cdf of the pitch rate in Figure 5-41 already deviates from a normal cdf. This can be attributed to the fact, that the pitch rate is directly affected by the nonlinearity in form of the uncertainty and the adaptive control signal. The GMM is seen to be able to approximate the shape of the shape of the state trajectory cdf.

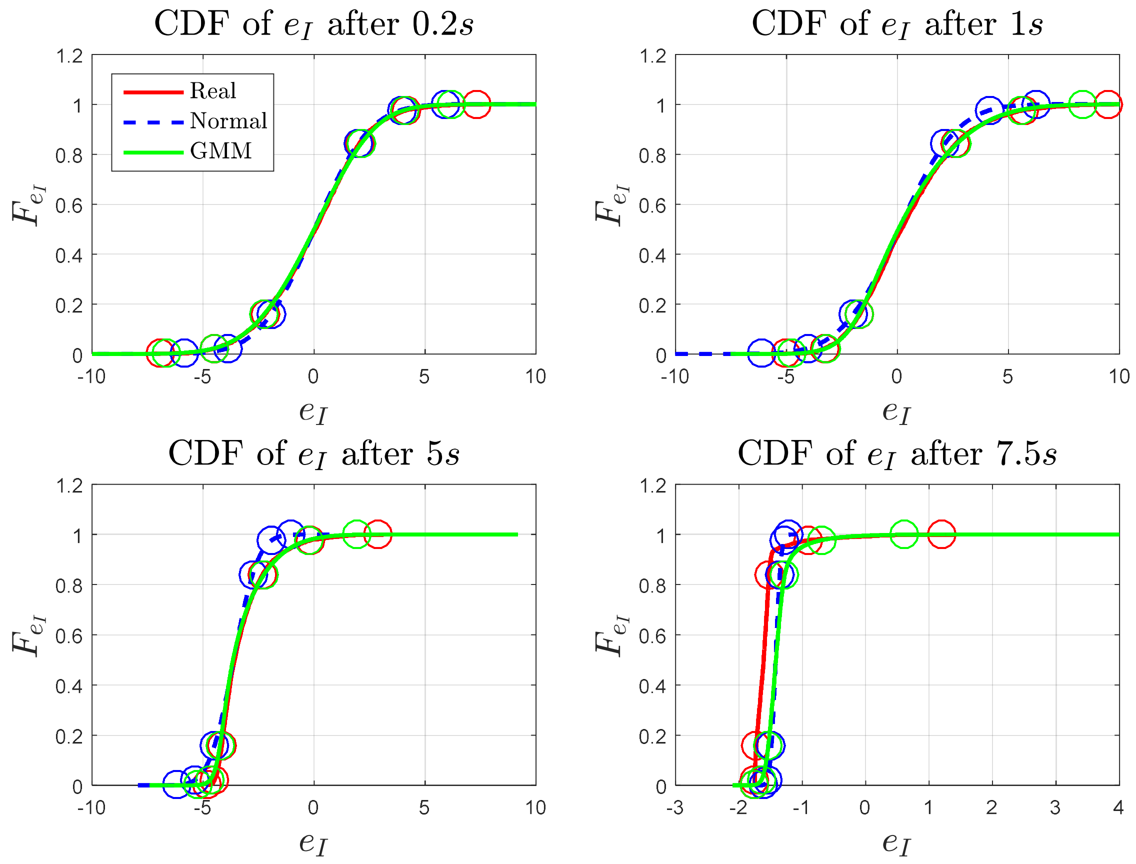


Figure 5-39: Predicted cdf of the integral error $e_I$in the short period example after 0.2s, 1s, 5s and 7.5s

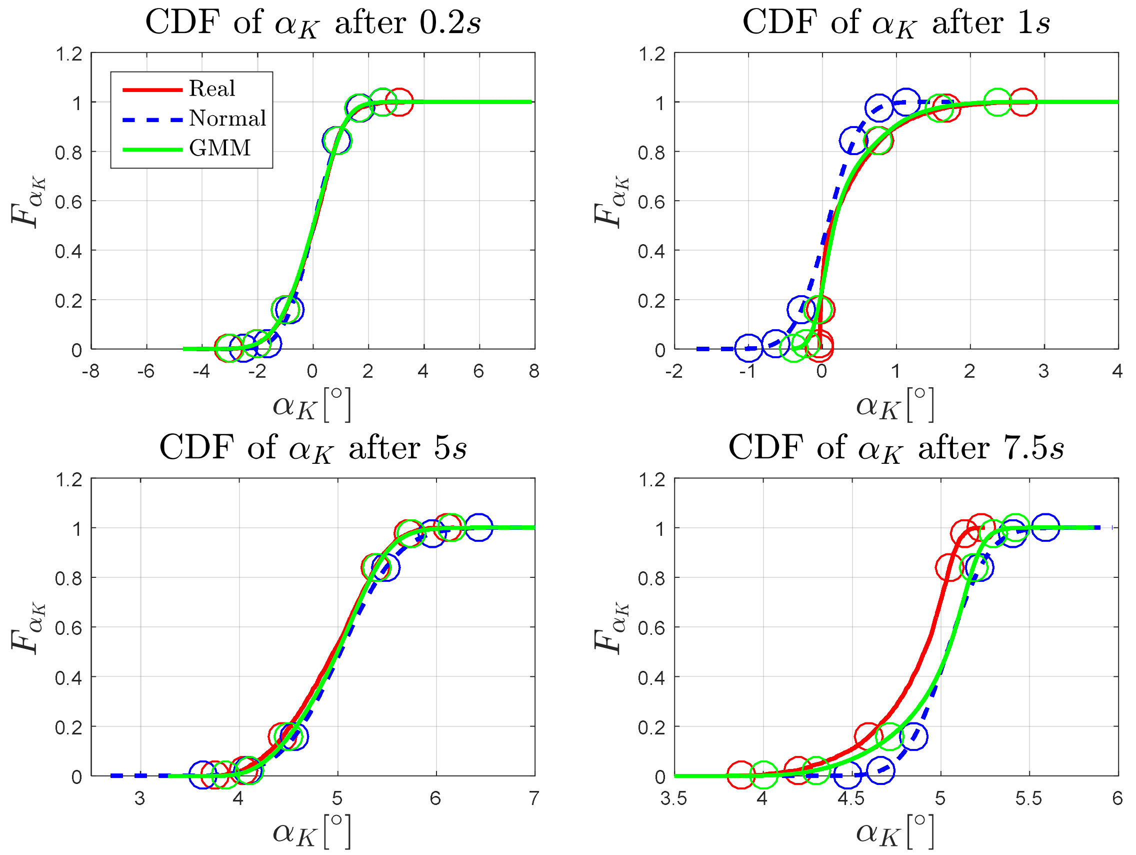


Figure 5-40: Predicted cdf of the angle of attack $\alpha_K$in the short period example after 0.2s, 1s, 5s and 7.5s

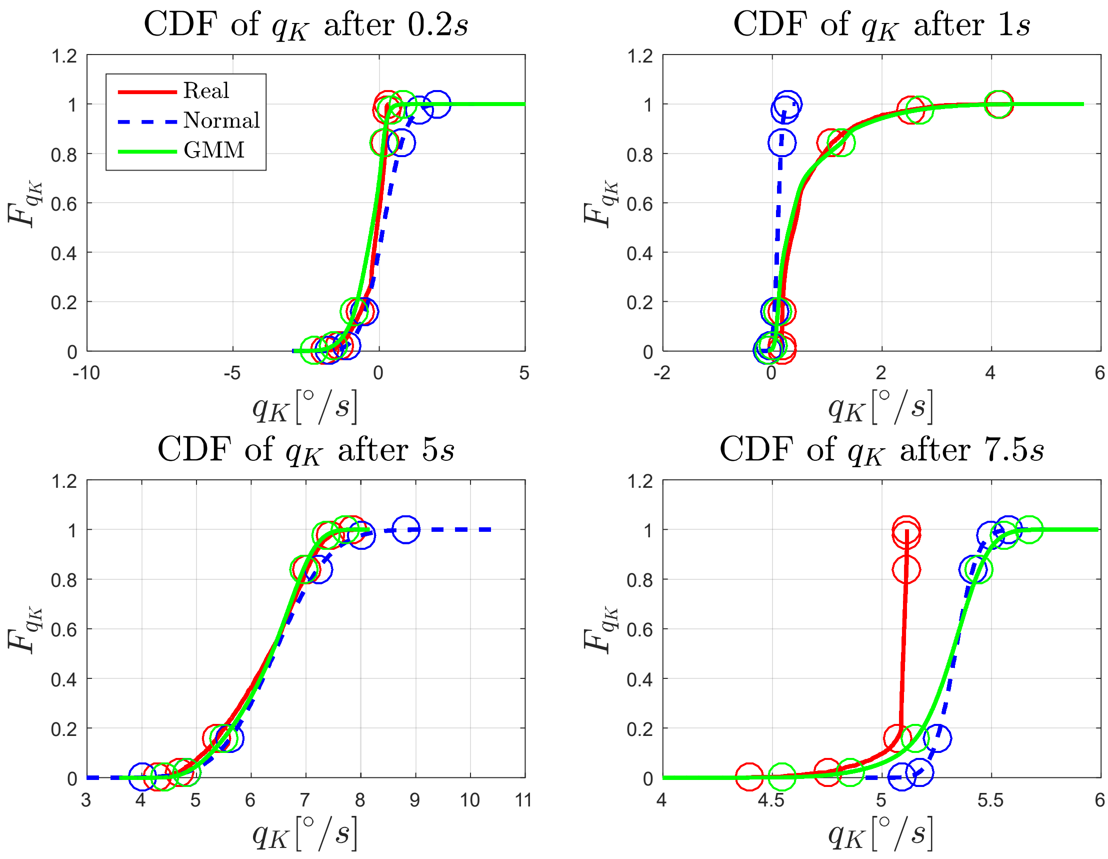


Figure 5-41: Predicted cdf of the pitch rate q_K in the short period example after 0.2s, 1s, 5s and 7.5s

During the simulation, the cdfs of the state trajectories deteriorate from the shape of a normal cdf. In this, the GMM approach is seen to approximate the resulting cdf closer than using a normal distribution based forecast. Still errors in the approximation capability of the presented approach occur. This can e.g. be seen in the pitch rate cdf after 7.5 seconds in Figure 5-41. In this, the simulated state trajectories are truncated above at approximately $5.2\frac{°}{s}$. Neither the GMM nor the normal distribution based approach is able to accurately represent the truncation. Furthermore, both forecast approaches underestimate the state distribution. Both errors can mainly be attributed to the uncertainty approximation error introduced by the GP regression. Furthermore, in order to form a concise reference, the simulation needs to be performed infinitely often. Since this is physically not possible, errors might occur when building the reference.

To assess the prediction performance and potential underestimation of the monitor further, again the GoC of section 5.1.2 is used. Figure 5-42 shows the results for the three plant states and varying numerical values for the relaxation term Δx. In conjunction with Figure 5-42, Table 5-12 lists the numerical values of the GoC at various Δx. It can be seen that for a vanishing relaxation term, forecasting the normal distribution sometimes leads to a higher GoC than using a GMM forecast. This can be attributed to initial approximation errors of the GMM. In that, the initial GMM is selected to approximate the initial normally distributed states, while the normal distribution forecast is initialized with the initial state distribution itself.

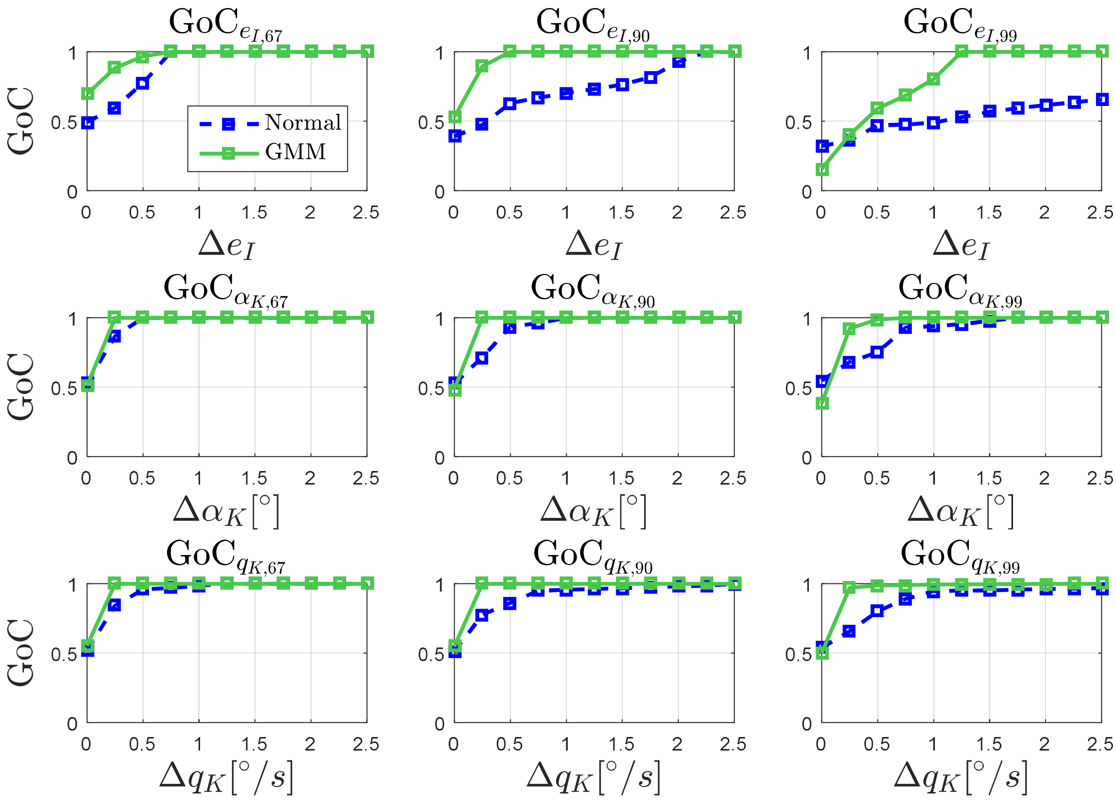


Figure 5-42: GoC of the plant states with relaxation term for the short period example

Nevertheless, the introduction of the relaxation term with only small values $\Delta e_I = 0.5, \Delta\alpha_K = 0.25°, \Delta q_K = 0.25°/s$ quickly sees the GoC values for the GMM to approach one. In contrast, the GoC of the Gaussian approximation increases slower and in most cases does not achieve a GoC of one within the selected parameter space. This indicates that the GMM encompasses the simulated state trajectory channels well. Note that the slow GoC increase in the 99% channel of the integral error results from the error in the posterior mean of the GP regression algorithm. As the latter leads to an error in the mean prediction (see Figure 5-6), the GoC is also affected. While it should be a goal to reduce this error, in the opinion of the author, the achievement of satisfactory prediction performance for the physical plant states α_K and q_K should be rated higher than the integral error e_I.

Figure 5-43 depicts the predicted probability of an angle of attack requirement violation according to equation (4-102) within $30s$ of the simulation. It can be seen that the predicted channels start crossing the upper limit after approximately 4 seconds and the lower limit at approximately 13 seconds. Consequently, the probability of a requirement violation is highest around those time instants. After the predicted distribution returns within the state limits set by the requirements, the probability of a requirement violation reduces to zero.

Figure 5-44 depicts the probability of a pitch rate requirement violation. Similar to the angle of attack, the predicted pitch rate distribution first crosses the upper state requirement limit after approximately 4 seconds and the lower limit at approximately 13 seconds. Note that when compared to the angle of attack, a significantly smaller part of the predicted pdf crosses the upper state requirement. As a consequence, the predicted probability of an upper state requirement violation is significantly lower for the pitch rate.

State / Forecast Basis	$GoC_{x_{67}}$	$GoC_{x_{90}}$	$GoC_{x_{99}}$
$\boldsymbol{\Delta e_I = 0, \Delta\alpha_K = 0°, \Delta q_K = 0°/s}$			
e_I– GMM	0.6955	0.5270	0.1550
e_I – Normal	0.4930	0.3910	0.3210
α_K– GMM	0.5080	0.4830	0.3825
α_K – Normal	0.5300	0.5265	0.5470
q_K – GMM	0.5520	0.5475	0.4990
q_K – Normal	0.5165	0.5105	0.5410
$\boldsymbol{\Delta e_I = 0.25, \Delta\alpha_K = 0.25°, \Delta q_K = 0.25°/s}$			
e_I– GMM	0.8845	0.8960	0.1550
e_I – Normal	0.5980	0.4740	0.3605
α_K– GMM	1.0	1.0	0.9240
α_K – Normal	0.8655	0.7150	0.6810
q_K – GMM	1.0	1.0	0.9705
q_K – Normal	0.8425	0.7775	0.6570
$\boldsymbol{\Delta e_I = 1, \Delta\alpha_K = 1°, \Delta q_K = 1°/s}$			
e_I– GMM	1.0	1.0	0.8040
e_I – Normal	1.0	0.7035	0.4900
α_K– GMM	1.0	1.0	1.0
α_K – Normal	1.0	1.0	0.9405
q_K – GMM	1.0	1.0	0.9935
q_K – Normal	0.9870	0.9565	0.9470
$\boldsymbol{\Delta e_I = 2, \Delta\alpha_K = 2°, \Delta q_K = 2°/s}$			
e_I– GMM	1.0	1.0	1.0
e_I – Normal	1.0	0.9275	0.6160
α_K– GMM	1.0	1.0	1.0
α_K – Normal	1.0	1.0	1.0
q_K – GMM	1.0	1.0	0.9980
q_K – Normal	1.0	0.9800	0.9615

Table 5-12: Numerical values for the GoC with relaxation term for the short period simulation example

Figure 5-45 shows the resulting OVL curve. The OVL curve is seen to be monotonically and nearly linearly decreasing for the most part of the simulation. Within the time frame of 5 and 7 seconds a steeper decrease can be seen. This can be attributed to the fact, that the variance of the mixture elements is significantly larger than during the rest of the simulation. As a consequence, the linear surrogate model of the nonlinear dynamics has to be valid over a larger support when compared to mixture elements with lower variance. As a result, the propagation of the mixture elements through the linearized models results leads to an increase in the OVL.

Finally, Figure 5-46 shows the predicted probability of a requirement violation P_{mod}, which is modified by the OVL according to (4-109). The resulting curves are seen to be more conservative. That is, with a higher probability P_{mod}, the chance of the monitor issuing a warning increases. In fact, with the parameter set of Table 5-10, the monitor issues a warning for the upper and lower requirement violation of the angle of attack. Furthermore, it issues a warning for the lower requirement violation of the pitch rate. In contrast, since $\delta_\zeta = 0.7$, no warning for the violation of the upper pitch rate requirement is issued.

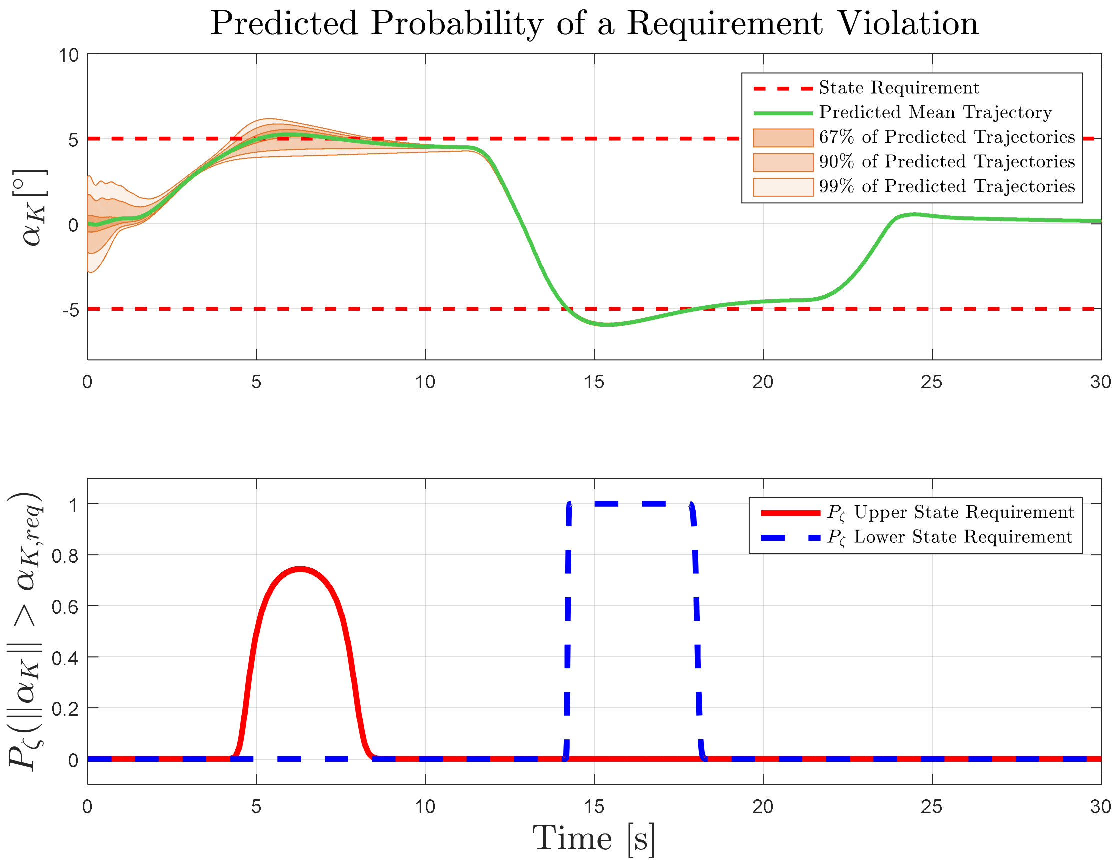


Figure 5-43: Probability of an angle of attack requirement violation in the short period example

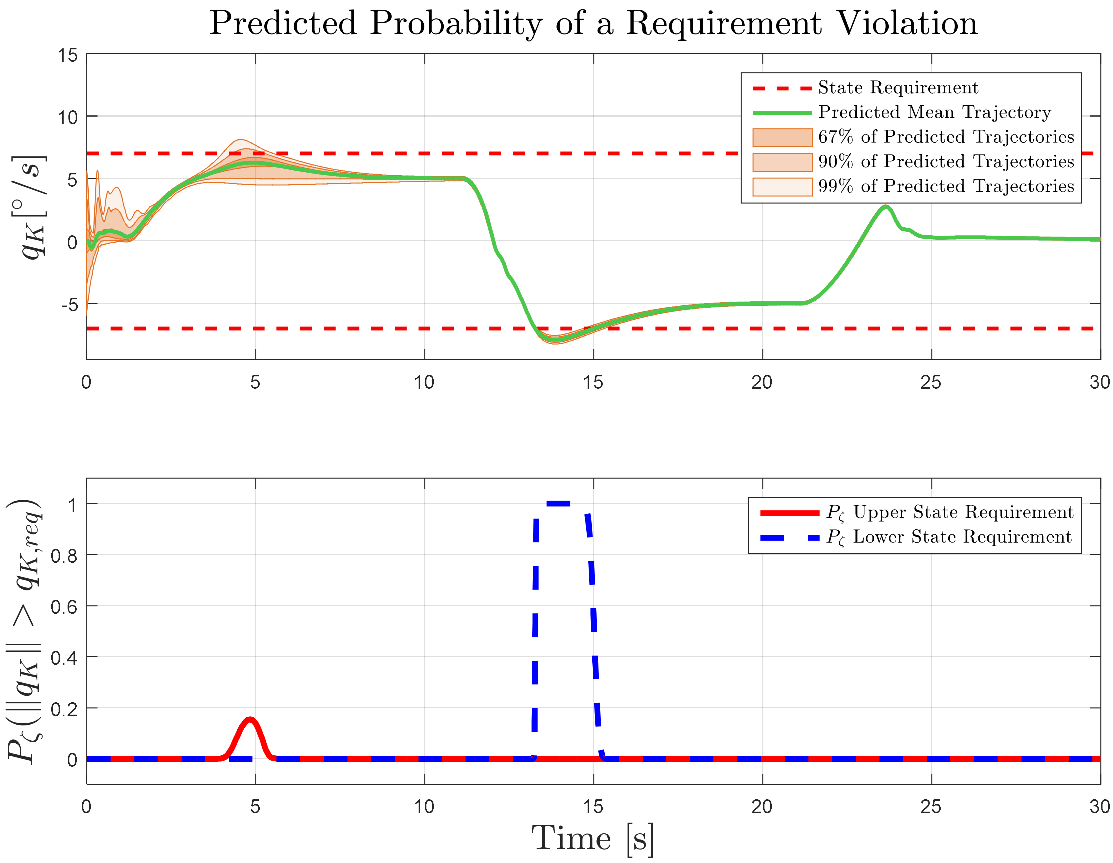


Figure 5-44: Probability of a pitch rate requirement violation in the short period example

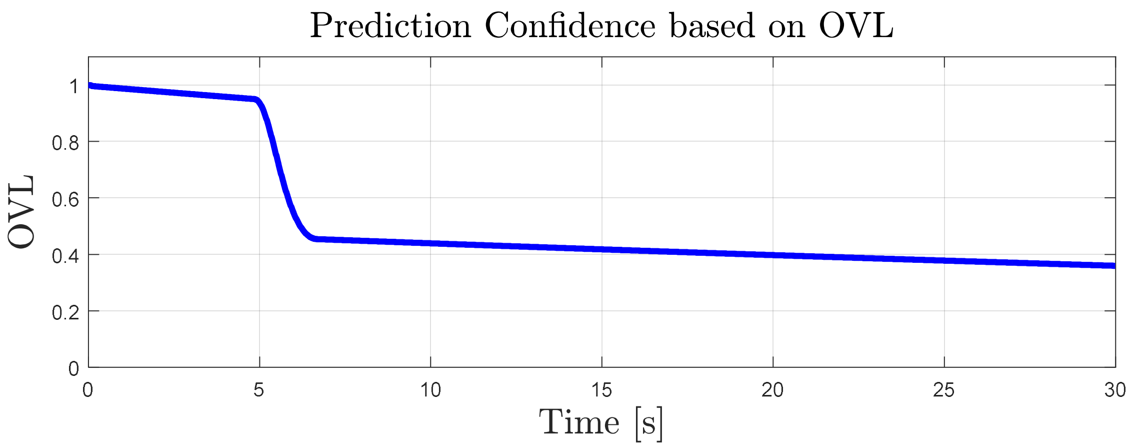


Figure 5-45: Predicted Confidence based on OVL in the short period example

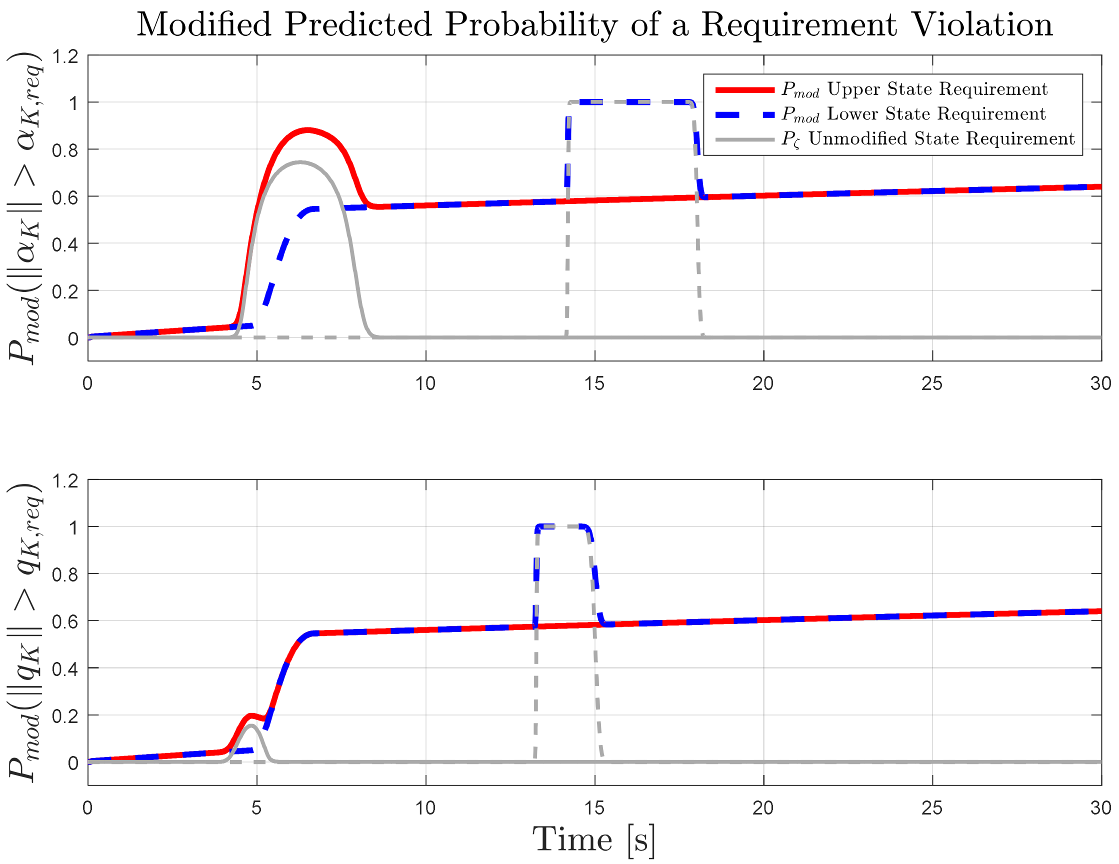


Figure 5-46: Modified predicted probability of a requirement violation in the short period example

5.3 Conclusion

Chapter 5 applies the proposed monitoring approach of section 4 to two simulation examples. In particular, section 5.1 applies the monitoring approach to the roll dynamics of a fighter aircraft subject to wingrock motion, which in turn is augmented by direct MRAC. Section 5.2 applies the monitor to a linearly controlled short period model of a F-16 fighter aircraft, which is augmented by PMRAC. In order to analyse the monitor, two references are generated. First, in order to establish a ground truth, the dynamical systems are simulated 2500 times with distributed initial conditions. Note, that the number of simulation runs is arbitrarily selected. To form an exact representation of the state distributions the simulation needs to be performed infinitely often, which in turn is physically impossible. Secondly, a forecast model based on a normal distribution is used in order to compare against the GMM centred approach.

In order to compare the results, different mean focussed error metrics are used. Since the latter neglect higher statistical moments, the GoC is introduced as a measure of under-approximation. The underlying idea is that the monitor is preferred to issue a false positive warning rather than missing a real threat. With the same goal in mind, the OVL from section 4.5.2 is used in order to self-assess a confidence over the prediction horizon. By using the OVL and adjusting the probability of an imminent requirement violation generated by the GMM forecast, a more conservative result is achieved.

In order to propagate the forecast model, the known system dynamics are linearized at the mean of the mixture elements in order to form a surrogate model. The GMM is iteratively propagated through the latter in order to achieve a prediction of the system states. The results of this propagation were seen to approximate the distribution of the predicted system states with only minor errors. Furthermore, the GMM is able to represent complex distributions. In contrast, using a forecast model based on a single normal distribution results in significant under estimation of the state distributions. Even adding a relaxation term does not necessarily solve this problem.

GP regression is applied, in order to form a belief on the uncertainty online. In both simulation examples, GP regression is seen to be able to model the uncertainty in the vicinity of recorded data points. Furthermore, the posterior variance allows accounting for uncertainty in the approximation. Hence, only a small amount of data points is required in order to form the representation. The simulation results demonstrate that incorporating the posterior distribution of the GP into the forecast indeed allows predicting the state distribution for the complete system. The results corroborate the hypothesis that splitting the forecast models into the known system dynamics and inferring the modelling uncertainty from online gathered data in fact represents a valid approach for predicting the system states.

While the presented approach shows promising results, some disadvantages arise. The GP regression is shown to increase the conservatism of the approach. The posterior variance of the GP directly influences the variance of the mixture elements. If the former is large, also the latter increases. Hence, the predicted pdf exhibits a larger support than intrinsically necessary. As a result, the false positive warnings of the monitor increase. On the other hand, the GP posterior variance depends on the hyperparameters as well as the number of stored points. Careful tuning can lead to a GP regression algorithm, which experiences a small posterior variance while still approximating the uncertainty correctly.

The number of required mixture elements is not known a-priori. It is shown that during parts of the simulation a single mixture element could suffice, thus reducing the computational complexity. Hypothetically, the latter is true if the system dynamics are very well approximated by linear dynamics. The monitor fixes the number of employed mixture elements. Future derivatives of the approach could try adjusting the number of mixture elements dynamically.

The OVL is intended as an approach to self-assess the quality of the prediction. As formulated in section 4.5.2 it needs to be precomputed. The underlying computation increases exponentially with the number of system parameters. While this does not constraint the presented simulation examples, systems with larger dimensions require more computational power. On the other hand, the OVL is presented as one possible way to gauge the confidence into the prediction. The proposed monitor allows for an easy integration of different approaches.

In the presented approaches, the requirements are formulated as constant values. Future approaches can incorporate more advanced requirement formulations as can be found in e.g. [17, p. 163].

All in all the proposed approach is shown to be able to predict the system states and act as a monitor for imminent requirement violation in adaptively controlled systems. Hence, it presents an alternative way of moving from an offline analysis to verification during runtime. Simultaneously, the mentioned disadvantages and short comings offer room for further development.

6 Application to Multirotor Systems

This section is intended to corroborate the results of the numerical simulation of section 5 by applying the monitoring algorithm to multirotor systems. Figure 6-1 shows Ascending Technologies' ([7]) quadcopter Hummingbird as well as the hexacopter Firefly as used by the Institute of *Flight System Dynamics* (FSD) at TU München. To corroborate the results of the numerical simulation, the monitoring algorithm is applied to the rotational dynamics of both systems.

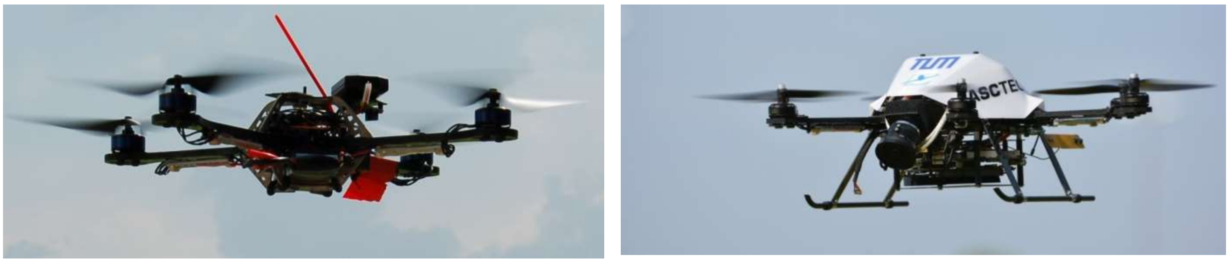


Figure 6-1: Quadcopter Hummingbird (left) and Hexacopter Firefly (right) as used at FSD

The outline of this section is as follows. Section 6.1 formulates the rotational dynamics of the multirotor systems. Section 6.2 presents the application of the monitoring algorithm to the adaptively controlled quadcopter. Similarly, section 6.3 presents the application of the online monitor to an adaptive controller applied to the hexacopter system. Section 6.4 concludes this chapter.

Parts of this section appeared in [148], [146] and [150].

6.1 Rotational Dynamics of the Multirotor Systems

This section derives the rotational dynamics dynamics of a multirotor system. The naming convention of Appendix F applies. The following derivation is based on ([94]).

In order to derive the dynamical model of a multirotor system, various assumptions need to be made. This thesis assumes a flat and non-rotating earth. As a result, the transport rate and the angular velocity of the earth can be neglected. These assumptions are justified, since the presented multirotor system only flies short distances and only operates for a short amount of time. Each multirotor system is treated as a rigid body, that is, the relative motion of mass elements is considered to be zero and the relative position between mass elements stays constant.

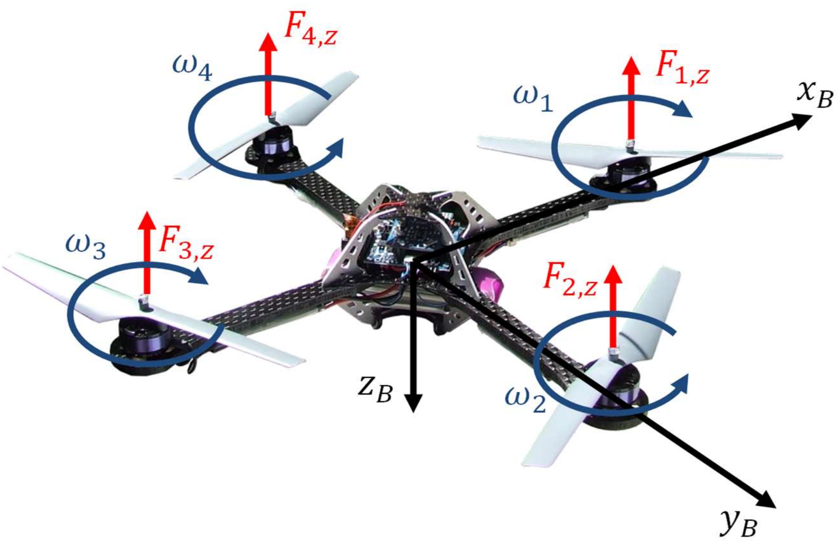


Figure 6-2: Forces and rotational rate of a multirotor propulsion system exemplified on a quadcopter

Figure 6-2 shows the forces and the rotational rate of a multirotor propulsion system exemplified on the Hummingbird. The following four assumptions, which are common for this kind of multirotor architecture (see i.a. [148], [88], [1]), are made throughout this section:

1. The motor dynamics are considerably faster than the rigid-body dynamics and can be neglected.
2. The rotation axis of each propeller is parallel to the body-fixed z-axis.
3. The propulsive forces within the rotor plane are small.
4. The thrust produced by the i-the rotor and the moment about its rotation axis can be modeled to be proportional to the square of its angular rate ω_i.

These assumptions form the basis for a model of the propulsive forces and moments. The forces $F_{i,z}$ and moments $M_{i,z}$ produced by the i –th propeller are:

$$F_{i,z} = -k_T \lambda_i \omega_i^2$$
$$M_{i,z} = -sgn(\omega_i)\lambda_i k_M \omega_i^2. \tag{6-1}$$

Here ω_i denotes the rotational rate of the i-th propeller and k_T and k_M represent the thrust and torque parameters of each rotor, respectively. The parameter $\lambda_i \in (0 \dots 1)$ describes the level of motor degradation. This thesis does not focus on fault tolerant flight control. Hence, λ_i is set equal to 1. Let $\vec{\boldsymbol{r}}^{GP_i} \in \mathbb{R}^3$ denote the position vector of the i-th propeller with respect to the center of gravity of the multirotor system. Furthemore, let l denote the length of the arms and let α_i be the angle between the x_B –axis and the respective arm. The propulsion moments are

$$\left(\vec{\boldsymbol{M}}_P^G\right)_B = \sum_{i=1}^{k}\left(\begin{pmatrix} l\cos(\alpha_i) \\ l\sin(\alpha_i) \\ 0 \end{pmatrix} \times \begin{pmatrix} 0 \\ 0 \\ F_{i,z} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ M_{i,z} \end{pmatrix}\right). \tag{6-2}$$

Solving equation (6-2) for the quadcopter Hummingbird results in

$$\left(\vec{\boldsymbol{M}}_P^G\right)_B = \underbrace{\begin{bmatrix} 0 & -lk_T & 0 & lk_T \\ lk_T & 0 & -lk_T & 0 \\ -k_M & k_M & -k_M & k_M \end{bmatrix}}_{\boldsymbol{C}_{Mu}} \underbrace{\begin{pmatrix} \omega_1^2 \\ \vdots \\ \omega_4^2 \end{pmatrix}}_{\boldsymbol{u}}. \tag{6-3}$$

Similarly, solving (6-2) for the hexacopter Firefly yields

$$\left(\vec{\boldsymbol{M}}_P^G\right)_B = \underbrace{\begin{bmatrix} -0.5lk_T & -lk_T & -0.5lk_T & 0.5lk_T & lk_T & 0.5lk_T \\ \frac{\sqrt{3}}{2}lk_T & 0 & -\frac{\sqrt{3}}{2}lk_T & -\frac{\sqrt{3}}{2}k_T & 0 & \frac{\sqrt{3}}{2}k_T \\ k_M & -k_M & k_M & -k_M & k_M & -k_M \end{bmatrix}}_{\boldsymbol{B}_M} \underbrace{\begin{pmatrix} \omega_1^2 \\ \vdots \\ \omega_6^2 \end{pmatrix}}_{\boldsymbol{u}}. \tag{6-4}$$

In any case, the expressions for the propulsion moments are simplified to

$$\left(\vec{\boldsymbol{M}}_P^G\right)_B = \boldsymbol{C}_{Mu}\boldsymbol{u}. \tag{6-5}$$

Additional moments $\vec{\boldsymbol{M}}_N^G$ acting on the multirotor system can be collected in the linear model (([94])

$$\left(\vec{\boldsymbol{M}}_N^G\right)_B = \boldsymbol{C}_{M\omega}(\vec{\boldsymbol{\omega}}_K^{IB})_B. \tag{6-6}$$

Here $(\vec{\boldsymbol{\omega}}_K^{IB})_B$ denotes the rotational rates of the quadcopter and $\boldsymbol{C}_{M\omega}$ denotes a matrix with unknown parameters. The total external moments acting on the multirotor system are

$$\left(\vec{\boldsymbol{M}}_T^G\right)_B = \left(\vec{\boldsymbol{M}}_P^G\right)_B + \left(\vec{\boldsymbol{M}}_N^G\right)_B. \tag{6-7}$$

Applying the principle of conservation of angular momentum with respect to the center of gravity yields the rotational dynamics of the multirotor (see e.g. [25]):

$$\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)_B^B = \left(\mathbf{I}_{BB}^G\right)^{-1}\left(\left(\vec{\boldsymbol{M}}_T^G\right)_B - (\vec{\boldsymbol{\omega}}_K^{IB})_B \times \left[\left(\mathbf{I}_{BB}^G\right)\cdot(\vec{\boldsymbol{\omega}}_K^{IB})_B\right]\right). \tag{6-8}$$

Inserting (6-7) into (6-8) yields

$$\begin{aligned} \left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)_B^B &= \left(\mathbf{I}_{BB}^G\right)^{-1}\left(\left(\vec{\boldsymbol{M}}_P^G\right)_B + \left(\vec{\boldsymbol{M}}_N^G\right)_B - (\vec{\boldsymbol{\omega}}_K^{IB})_B \times \left[\left(\mathbf{I}_{BB}^G\right)\cdot(\vec{\boldsymbol{\omega}}_K^{IB})_B\right]\right) \\ &= \left(\mathbf{I}_{BB}^G\right)^{-1}\left(\boldsymbol{C}_{Mu}\boldsymbol{u} + \boldsymbol{C}_{M\omega}(\vec{\boldsymbol{\omega}}_K^{IB})_B - (\vec{\boldsymbol{\omega}}_K^{IB})_B \times \left[\left(\mathbf{I}_{BB}^G\right)\cdot(\vec{\boldsymbol{\omega}}_K^{IB})_B\right]\right). \end{aligned} \tag{6-9}$$

Rearranging terms yields the rotational dynamics

$$\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)_B^B = \left(\mathbf{I}_{BB}^G\right)^{-1}\boldsymbol{C}_{M\omega}(\vec{\boldsymbol{\omega}}_K^{IB})_B + \left(\mathbf{I}_{BB}^G\right)^{-1}\boldsymbol{C}_{Mu}\boldsymbol{u} - \left(\mathbf{I}_{BB}^G\right)^{-1}(\vec{\boldsymbol{\omega}}_K^{IB})_B \times \left[\left(\mathbf{I}_{BB}^G\right)\cdot(\vec{\boldsymbol{\omega}}_K^{IB})_B\right]. \tag{6-10}$$

Due to the assumption of a non-rotating and flat earth we have $(\vec{\boldsymbol{\omega}}_K^{IB})_B = \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B$. As a consequence, equation (6-10) becomes

$$\left(\dot{\vec{\boldsymbol{\omega}}}_K^{OB}\right)_B^B = \left(\mathbf{I}_{BB}^G\right)^{-1}\boldsymbol{C}_{M\omega}\left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B + \left(\mathbf{I}_{BB}^G\right)^{-1}\boldsymbol{C}_{Mu}\boldsymbol{u} - \left(\mathbf{I}_{BB}^G\right)^{-1}\left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B \times \left[\left(\mathbf{I}_{BB}^G\right)\cdot\left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B\right] \tag{6-11}$$

For simplicity, the elements of $\left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B$ will be denoted as $\left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B = [p_B \quad q_B \quad r_B]^T$ in the following. Also, the plant states $\left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B$ will be represented by $\boldsymbol{x}_P$ sometimes to simplify notations.

6.2 Quadcopter Application

This section details the application of the monitoring algorithm to the adaptively controlled quadcopter Hummingbird. The control architecture consists of a fully adaptive rate controller, which in turn is used within lectures at FSD ([94]). The presented flight test has been reported in [150] using a different monitoring approach. Hence, the data presented in this chapter has been priorly recorded during the flight tests in [150]. Here, it is used along with the monitoring algorithm proposed within this thesis. In the authors opinion, this still presents a viable use case, as the monitoring algorithm can also run on a ground station. This way, warnings can still be issued to the controlling pilot.

The outline of this section is as follows: Section 6.2.1 details the control architecture used for the quadcopter. Section 6.2.2 presents the application of the monitoring algorithm. Finally, section 6.2.3 depicts the experimental setup and results.

6.2.1 Quadcopter: Control Architecture

The control architecture used in [94] deviates from section 3.2 as no baseline (tracking) controller is applied. Rather, a full adaptive controller with direct MRAC is used in order to make the system dynamics behave like the a-priori selected reference model dynamics:

$$\underbrace{\begin{bmatrix} \dot{p}_{rm} \\ \dot{q}_{rm} \\ \dot{r}_{rm} \end{bmatrix}}_{\dot{x}_{rm}} = \boldsymbol{A}_{rm} \underbrace{\begin{bmatrix} p_{rm} \\ q_{rm} \\ r_{rm} \end{bmatrix}}_{x_{rm}} + \boldsymbol{B}_{rm} \underbrace{\begin{bmatrix} p_{cmd} \\ q_{cmd} \\ r_{cmd} \end{bmatrix}}_{\boldsymbol{r}}. \tag{6-12}$$

Here, $\boldsymbol{x}_{rm} = [p_{rm} \quad q_{rm} \quad r_{rm}]$ represents the reference model states and $\boldsymbol{r} = [p_{cmd} \quad q_{cmd} \quad r_{cmd}]^T$ represents the vector of exogenous reference inputs for the three rotational axes. In order to proceed with the design of the control architecture, equation (6-11) is further analyzed and simplified. For the controller design, $\left(\mathbf{I}_{BB}^G\right)$ is assumed to be a diagonal matrix such that

$$\left(\mathbf{I}_{BB}^G\right) = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}. \tag{6-13}$$

First, solving the cross product $\left(\mathbf{I}_{BB}^G\right)^{-1}\left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B \times \left[\left(\mathbf{I}_{BB}^G\right) \cdot \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B\right]$ in equation (6-11) yields

$$\left(\mathbf{I}_{BB}^G\right)^{-1}\left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B \times \left[\left(\mathbf{I}_{BB}^G\right) \cdot \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B\right] = \tag{6-14}$$

$$= \underbrace{\begin{bmatrix} \frac{(I_{zz} - I_{yy})}{I_{xx}} & 0 & 0 \\ 0 & \frac{(I_{xx} - I_{zz})}{I_{yy}} & 0 \\ 0 & 0 & \frac{(I_{yy} - I_{xx})}{I_{zz}} \end{bmatrix}}_{\boldsymbol{\Theta}^*_{\boldsymbol{\varphi},\boldsymbol{I}}} \underbrace{\begin{pmatrix} r_B q_B \\ r_B p_B \\ p_B q_B \end{pmatrix}}_{\boldsymbol{\varphi}(\boldsymbol{x}_P)} = \boldsymbol{\Theta}^{*\,T}_{\boldsymbol{\varphi},\boldsymbol{I}} \boldsymbol{\varphi}(\boldsymbol{x}_P).$$

Secondly, the input term of equation (6-11) is analyzed. Multiplying $\left(\mathbf{I}^G_{BB}\right)^{-1}$ with $\boldsymbol{C}_{Mu}$ yields

$$\left(\mathbf{I}^G_{BB}\right)^{-1} \boldsymbol{C}_{Mu} = \begin{bmatrix} 0 & -\frac{lk_T}{I_{xx}} & 0 & \frac{lk_T}{I_{xx}} \\ \frac{lk_T}{I_{yy}} & 0 & -\frac{lk_T}{I_{yy}} & 0 \\ -\frac{k_M}{I_{zz}} & \frac{k_M}{I_{zz}} & -\frac{k_M}{I_{zz}} & \frac{k_M}{I_{zz}} \end{bmatrix}. \qquad \textbf{(6-15)}$$

In [94], equation (6-15) is decomposed into

$$\left(\mathbf{I}^G_{BB}\right)^{-1} \boldsymbol{C}_{Mu} = \underbrace{\begin{bmatrix} 0 & -1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ -1 & 1 & -1 & 1 \end{bmatrix}}_{\boldsymbol{B}_P} \boldsymbol{\Lambda} = \boldsymbol{B}_P \boldsymbol{\Lambda}. \qquad \textbf{(6-16)}$$

The underlying idea is that the parameters $\boldsymbol{\Lambda} > \boldsymbol{0}$ of the input matrix $\left(\mathbf{I}^G_{BB}\right)^{-1} \boldsymbol{C}_{Mu}$ are unknown, but the structure and sign are defined a-priori. The uncertainty arising from the unknown input parameters $\boldsymbol{\Lambda}$ shall be handled by the adaptive controller. Exchanging terms in equation (6-11) and simplifying with $\boldsymbol{x}_P := \left(\vec{\boldsymbol{\omega}}^{OB}_K\right)_B$ yields

$$\dot{\boldsymbol{x}}_P = \left(\mathbf{I}^G_{BB}\right)^{-1} \boldsymbol{C}_{M\omega} \boldsymbol{x}_P + \boldsymbol{B}_P \boldsymbol{\Lambda} \boldsymbol{u} - \boldsymbol{\Theta}^{*\,T}_{\boldsymbol{\varphi},\boldsymbol{I}} \boldsymbol{\varphi}(\boldsymbol{x}_P) \qquad \textbf{(6-17)}$$

Note, that the right More-Penrose pseudoinverse of $\boldsymbol{B}_P \boldsymbol{\Lambda}$ exists, as $\left(\mathbf{I}^G_{BB}\right)^{-1} \boldsymbol{C}_{Mu}$ has linearly independent rows. As a consequence, $\boldsymbol{B}_P \boldsymbol{\Lambda} (\boldsymbol{B}_P \boldsymbol{\Lambda})^+ = \boldsymbol{I}$. Thus, define

$$\begin{aligned} \boldsymbol{\Theta}^{*\,T}_x &= -(\boldsymbol{B}_P \boldsymbol{\Lambda})^+ \left[\left(\mathbf{I}^G_{BB}\right)^{-1} \boldsymbol{C}_{M\omega} - \boldsymbol{A}_{rm} \right] \\ \boldsymbol{\Theta}^{*\,T}_{\boldsymbol{\varphi}} &= (\boldsymbol{B}_P \boldsymbol{\Lambda})^+ \boldsymbol{\Theta}^{*\,T}_{\boldsymbol{\varphi},\boldsymbol{I}}. \end{aligned} \qquad \textbf{(6-18)}$$

Furthermore, assume the following matching condition exists

$$\boldsymbol{B}_{rm} = \boldsymbol{B}_P \boldsymbol{\Lambda} \boldsymbol{\Theta}^{*T}_r. \qquad \textbf{(6-19)}$$

Adding and subtracting $\boldsymbol{A}_{rm} \boldsymbol{x}_P$ and $\boldsymbol{B}_{rm} \boldsymbol{r}$ in (6-17) and rearranging terms yields

$$\dot{\boldsymbol{x}}_P = \boldsymbol{A}_{rm} \boldsymbol{x}_P + \boldsymbol{B}_{rm} \boldsymbol{r} + \boldsymbol{B}_P \boldsymbol{\Lambda} \left(\boldsymbol{u} - \boldsymbol{\Theta}^{*T}_x \boldsymbol{x}_P - \boldsymbol{\Theta}^{*\,T}_{\boldsymbol{\varphi}} \boldsymbol{\varphi}(\boldsymbol{x}_P) - \boldsymbol{\Theta}^{*T}_r \boldsymbol{r} \right). \qquad \textbf{(6-20)}$$

The control signal $\boldsymbol{u}$ is designed to cancel the uncertain terms in the brackets

$$\boldsymbol{u} = \boldsymbol{\Theta}_x^T \boldsymbol{x}_P + \boldsymbol{\Theta}_\varphi^T \boldsymbol{\varphi}(\boldsymbol{x}_P) + \boldsymbol{\Theta}_r^T \boldsymbol{r} + \boldsymbol{\Theta}_d^T = \underbrace{\begin{bmatrix}\boldsymbol{\Theta}_x^T & \boldsymbol{\Theta}_\varphi^T & \boldsymbol{\Theta}_r^T & \boldsymbol{\Theta}_d^T\end{bmatrix}}_{\boldsymbol{\Theta}^T} \underbrace{\begin{bmatrix}\boldsymbol{x}_P \\ \boldsymbol{\varphi}(\boldsymbol{x}_P) \\ \boldsymbol{r} \\ \boldsymbol{I}_{3x1}\end{bmatrix}}_{\boldsymbol{\Phi}} = \boldsymbol{\Theta}^T \boldsymbol{\Phi}(\boldsymbol{x}_P, \boldsymbol{r}). \tag{6-21}$$

Here, $\boldsymbol{I}_{3x1}$ denotes a vector with ones as the entries. Note that the authors in [94] add a term $\boldsymbol{\Theta}_d$ to the update law, in order to counter disturbances. The parameter update laws follow (3-19) and use a σ-modification with $k(\boldsymbol{\Theta}) = -\sigma\boldsymbol{\Theta}$, such that

$$\dot{\boldsymbol{\Theta}} = -\boldsymbol{\Gamma}\boldsymbol{\Phi}(\boldsymbol{x}_P, \boldsymbol{r})\boldsymbol{e}^T \boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}\sigma\boldsymbol{\Theta}. \tag{6-22}$$

6.2.2 Quadcopter: Application of the Monitoring Algorithm

This section applies the monitoring approach of section 4 to the quadcopter system. The monitoring algorithm is not directly applicable to the plant dynamics (6-20). The reason for this is that the input parameters $\boldsymbol{\Lambda}$ are unknown. As a consequence, the input $\boldsymbol{u}$ cannot completely be decoupled from all uncertainties. This in turn is necessary to separate the system dynamics. The known part will be represented by a linear surrogate model, while the unknown part will be inferred from online gathered data using GP regression. Adding and subtracting $\boldsymbol{B}_P\boldsymbol{u}$ in equation (6-20) allows for achieving the required separation:

$$\begin{aligned}\dot{\boldsymbol{x}}_P &= \boldsymbol{A}_{rm}\boldsymbol{x}_P + \boldsymbol{B}_{rm}\boldsymbol{r} + \boldsymbol{B}_P\boldsymbol{u} - \boldsymbol{B}_P\boldsymbol{u} + \boldsymbol{B}_P\boldsymbol{\Lambda}\boldsymbol{u} - \boldsymbol{B}_P\boldsymbol{\Lambda}\boldsymbol{\Theta}^{*^T}\boldsymbol{\Phi}(\boldsymbol{x}_P, \boldsymbol{r}) \\ &= \boldsymbol{A}_{rm}\boldsymbol{x}_P + \boldsymbol{B}_{rm}\boldsymbol{r} + \boldsymbol{B}_P\boldsymbol{\Theta}^T\boldsymbol{\Phi}(\boldsymbol{x}_P, \boldsymbol{r}) - \boldsymbol{B}_P\underbrace{\left[(\boldsymbol{I} - \boldsymbol{\Lambda})\boldsymbol{u} + \boldsymbol{\Lambda}\boldsymbol{\Theta}^{*^T}\boldsymbol{\Phi}(\boldsymbol{x}_P, \boldsymbol{r})\right]}_{\Delta(\boldsymbol{u}, \boldsymbol{x}_P, \boldsymbol{r})}.\end{aligned} \tag{6-23}$$

In the following, first the linear surrogate model is constructed, which is required in order to propagate a GMM. In a second step, the setup of the GP regression algorithm is detailed.

Construction of the linear surrogate model

In order to propagate the GMM, first the linearized system dynamics need to be derived. Bringing the known dynamics of (6-23) and (6-22) in the form of (4-6) and (4-11) yields

$$\boldsymbol{f}(\boldsymbol{x}_P, \boldsymbol{r}, \boldsymbol{\Theta}) = \begin{bmatrix}\boldsymbol{f}_x(\boldsymbol{x}_P, \boldsymbol{r}, \boldsymbol{\Theta}) \\ \boldsymbol{f}_\Theta(\boldsymbol{x}_P, \boldsymbol{r}, \boldsymbol{\Theta})\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}_{rm}\boldsymbol{x}_P + \boldsymbol{B}_{rm}\boldsymbol{r} + \boldsymbol{B}_P\boldsymbol{\Theta}_{x\varphi}^T\boldsymbol{\Phi}_x(\boldsymbol{x}_P) + \boldsymbol{B}_P\boldsymbol{\Theta}_{rd}^T\boldsymbol{\Phi}_{rd}(\boldsymbol{r}) \\ -\boldsymbol{\Gamma}\boldsymbol{\Phi}(\boldsymbol{x}_P, \boldsymbol{r})\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}\sigma\boldsymbol{\Theta}\end{bmatrix}. \tag{6-24}$$

Here, the adaptive control signal $\boldsymbol{\Theta}^T\boldsymbol{\Phi}(\boldsymbol{x}_P, \boldsymbol{r})$ is split into two parts such that

$$\begin{aligned}\boldsymbol{\Theta}_{x\varphi}^T\boldsymbol{\Phi}_x(\boldsymbol{x}_P) &= \underbrace{\begin{bmatrix}\boldsymbol{\Theta}_x^T & \boldsymbol{\Theta}_\varphi^T\end{bmatrix}}_{\boldsymbol{\Theta}_{x\varphi}^T}\underbrace{\begin{bmatrix}\boldsymbol{x}_P \\ \boldsymbol{\varphi}(\boldsymbol{x}_P)\end{bmatrix}}_{\boldsymbol{\Phi}_x(\boldsymbol{x}_P)}, \\ \boldsymbol{\Theta}_{rd}^T\boldsymbol{\Phi}_{rd}(\boldsymbol{r}) &= \underbrace{\begin{bmatrix}\boldsymbol{\Theta}_r^T & \boldsymbol{\Theta}_d^T\end{bmatrix}}_{\boldsymbol{\Theta}_{rd}^T}\underbrace{\begin{bmatrix}\boldsymbol{r} \\ \boldsymbol{I}_{3x1}\end{bmatrix}}_{\boldsymbol{\Phi}_{rd}(\boldsymbol{r})}.\end{aligned} \tag{6-25}$$

The exogenous reference input $\boldsymbol{r}$ is supplied by a human pilot. Since this thesis does not consider pilot models, the reference input $\boldsymbol{r}$ is considered to be constant over the prediction horizon. The underlying idea is that the term $\boldsymbol{\Theta}_{rd}^T\boldsymbol{\Phi}_{rd}(\boldsymbol{r})$ does not contribute to the covariance update of the system states. Hence, in order to reduce the computational complexity, it is separated from $\boldsymbol{\Theta}_{x\varphi}^T\boldsymbol{\Phi}_x(\boldsymbol{x}_P)$. Adjusting accordingly yields

$$
\begin{aligned}
\boldsymbol{f}(\boldsymbol{x}_P,\boldsymbol{r},\boldsymbol{\Theta}) &= \begin{bmatrix} \boldsymbol{f}_x(\boldsymbol{x}_P,\boldsymbol{r},\boldsymbol{\Theta}) \\ \boldsymbol{f}_{\Theta_{x\varphi}}(\boldsymbol{x}_P,\boldsymbol{r},\boldsymbol{\Theta}_{x\varphi}) \\ \boldsymbol{f}_{\Theta_{rd}}(\boldsymbol{x}_P,\boldsymbol{r},\boldsymbol{\Theta}_{rd}) \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{A}_{rm}\boldsymbol{x}_P + \boldsymbol{B}_{rm}\boldsymbol{r} + \boldsymbol{B}_P\boldsymbol{\Theta}_{x\varphi}^T\boldsymbol{\Phi}_x(\boldsymbol{x}_P) + \boldsymbol{B}_P\boldsymbol{\Theta}_{rd}^T\boldsymbol{\Phi}_{rd}(\boldsymbol{r}) \\ -\boldsymbol{\Gamma}_x\boldsymbol{\Phi}_x(\boldsymbol{x}_P)\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}\sigma\boldsymbol{\Theta}_{x\varphi} \\ -\boldsymbol{\Gamma}_{rd}\boldsymbol{\Phi}_{rd}(\boldsymbol{r})\boldsymbol{e}_l^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}\sigma\boldsymbol{\Theta}_{rd} \end{bmatrix}
\end{aligned}
\tag{6-26}
$$

Here, $\boldsymbol{\Gamma}_x$ and $\boldsymbol{\Gamma}_{rd}$ denote the submatrices of $\boldsymbol{\Gamma}$ with appropriate dimension.

Following, the steps in section 4.3, evaluating $\boldsymbol{f}(\boldsymbol{x}_{P,l},\boldsymbol{r},\boldsymbol{\Theta}_l)$ at the linearization point $\boldsymbol{x}_P = \boldsymbol{x}_{P,l}$ and $\boldsymbol{\Theta} = \boldsymbol{\Theta}_l$ gives

$$
\begin{aligned}
\boldsymbol{f}(\boldsymbol{x}_{P,l},\boldsymbol{r},\boldsymbol{\Theta}_l) &= \begin{bmatrix} \boldsymbol{f}_{x,l}(\boldsymbol{x}_{P,l},\boldsymbol{r},\boldsymbol{\Theta}_l) \\ \boldsymbol{f}_{\Theta_{x\varphi},l}(\boldsymbol{x}_{P,l},\boldsymbol{r},\boldsymbol{\Theta}_{x\varphi,l}) \\ \boldsymbol{f}_{\Theta_{rd},l}(\boldsymbol{x}_{P,l},\boldsymbol{r},\boldsymbol{\Theta}_{rd,l}) \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{A}_{rm}\boldsymbol{x}_{P,l} + \boldsymbol{B}_{rm}\boldsymbol{r} + \boldsymbol{B}_P\boldsymbol{\Theta}_{x\varphi,l}^T\boldsymbol{\Phi}_x(\boldsymbol{x}_{P,l}) + \boldsymbol{B}_P\boldsymbol{\Theta}_{rd,l}^T\boldsymbol{\Phi}_{rd}(\boldsymbol{r}) \\ -\boldsymbol{\Gamma}_x\boldsymbol{\Phi}_x(\boldsymbol{x}_{P,l})\boldsymbol{e}_l^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}_x\sigma\boldsymbol{\Theta}_{x\varphi,l} \\ -\boldsymbol{\Gamma}_{rd}\boldsymbol{\Phi}_{rd}(\boldsymbol{r})\boldsymbol{e}_l^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}_{rd}\sigma\boldsymbol{\Theta}_{rd,l} \end{bmatrix}.
\end{aligned}
\tag{6-27}
$$

In order to construct the matrix $\boldsymbol{A}_{lin}$, the partial derivatives of $\boldsymbol{f}_x,\boldsymbol{f}_{\Theta_{x\varphi}}$ and $\boldsymbol{f}_{\Theta_{rd}}$ with respect to the system states $\boldsymbol{x}_P$ and the controller states $\boldsymbol{\Theta}$ need to be derived. Remember that $\boldsymbol{r}$ is treated as a constant throughout the prediction horizon. The partial derivatives of $\boldsymbol{f}_x$ with respect to the system states $\boldsymbol{x}_P$ and the controller states $\boldsymbol{\Theta}$ are

$$
\begin{aligned}
\left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} &= \boldsymbol{A}_{rm} + \boldsymbol{B}_P\boldsymbol{\Theta}_{x\varphi}^T\left.\frac{\partial \boldsymbol{\Phi}_x(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}}, \\
\left.\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{\Theta}_{x\varphi}}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} &= \boldsymbol{B}_P\boldsymbol{\Phi}_x^T(\boldsymbol{x}_{P,l}), \\
\frac{\partial \boldsymbol{f}_x}{\partial \boldsymbol{\Theta}_{rd}} &= \boldsymbol{B}_P\boldsymbol{\Phi}_{rd}^T(\boldsymbol{r}).
\end{aligned}
\tag{6-28}
$$

The partial derivate $\left.\frac{\partial \boldsymbol{\Phi}_x(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}}$ is

$$
\left.\begin{bmatrix} \dfrac{\partial \boldsymbol{\Phi}_x(\boldsymbol{x}_P)}{\partial p_B} & \dfrac{\partial \boldsymbol{\Phi}_x(\boldsymbol{x}_P)}{\partial q_B} & \dfrac{\partial \boldsymbol{\Phi}_x(\boldsymbol{x}_P)}{\partial r_B} \end{bmatrix}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} = \begin{bmatrix} 1 & 0 & 0 & 0 & r_{B,l} & q_{B,l} \\ 0 & 1 & 0 & p_{B,l} & 0 & q_{B,l} \\ 0 & 0 & 1 & p_{B,l} & r_{B,l} & 0 \end{bmatrix}^T. \tag{6-29}
$$

Similarly, the partial derivatives of $\boldsymbol{f}_{\Theta_{x\varphi}}$ with respect to the system states $\boldsymbol{x}_P$ and the controller states $\boldsymbol{\Theta}$ are

$$\begin{aligned}
\left.\frac{\partial \boldsymbol{f}_{\Theta_{x\varphi}}}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} &= -\boldsymbol{\Gamma}_x \left.\frac{\partial \boldsymbol{\Phi}_x(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} \boldsymbol{x}_{P,l}^T \boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}_x \boldsymbol{\Phi}_x(\boldsymbol{x}_{P,l})\boldsymbol{B}_P^T\boldsymbol{P} \\
&\quad +\boldsymbol{\Gamma}_x \left.\frac{\partial \boldsymbol{\Phi}_x(\boldsymbol{x}_P)}{\partial \boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}} \boldsymbol{x}_{rm}^T \boldsymbol{P}\boldsymbol{B}_P, \\
\left.\frac{\partial \boldsymbol{f}_{\Theta_{x\varphi}}}{\partial \boldsymbol{\Theta}_{x\varphi}}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} &= -\boldsymbol{\Gamma}_x \sigma \qquad\qquad \text{(6-30)} \\
\left.\frac{\partial \boldsymbol{f}_{\Theta_{x\varphi}}}{\partial \boldsymbol{\Theta}_{rd}}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} &= \boldsymbol{0}.
\end{aligned}$$

Finally, the partial derivatives of $\boldsymbol{f}_{\Theta_{rd}}$ with respect to the system states $\boldsymbol{x}_P$ and the controller states $\boldsymbol{\Theta}$ are

$$\begin{aligned}
\left.\frac{\partial \boldsymbol{f}_{\Theta_{rd}}}{\partial \boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} &= -\boldsymbol{\Gamma}\boldsymbol{\Phi}_{rd}(\boldsymbol{r})\boldsymbol{B}_P^T\boldsymbol{P}, \\
\left.\frac{\partial \boldsymbol{f}_{\Theta_{x\varphi}}}{\partial \boldsymbol{\Theta}_{x\varphi}}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} &= \boldsymbol{0}, \qquad\qquad \text{(6-31)} \\
\left.\frac{\partial \boldsymbol{f}_{\Theta_{x\varphi}}}{\partial \boldsymbol{\Theta}_{rd}}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}} &= -\boldsymbol{\Gamma}_{rd}\sigma.
\end{aligned}$$

Application of GP regression to the quadcopter

The monitoring algorithm uses GP regression as described in section 4.4.3 in order to approximate the modelling uncertainty online. Therefore, the latter needs to be estimated at specific points in time. In order to reduce complexity, the uncertainty $\boldsymbol{\Delta}$ is assumed to solely depend on the states $\boldsymbol{x}_P$. This simplification is warranted, as the control input $\boldsymbol{u}$ directly depends on the plant states itself. Rearranging equation (6-23) according to (4-92) yields

$$\widehat{\boldsymbol{\Delta}}(\widehat{\boldsymbol{x}}_{P,k}) = \boldsymbol{B}_P^{+}\left(\boldsymbol{A}_{rm}\widehat{\boldsymbol{x}}_{P,k} + \boldsymbol{B}_{rm}\boldsymbol{r}_k + \boldsymbol{B}_P\boldsymbol{\Theta}_k^T\boldsymbol{\Phi}(\widehat{\boldsymbol{x}}_{P,k},\boldsymbol{r}_k) - \dot{\widehat{\boldsymbol{x}}}_{P,k}\right). \qquad \text{(6-32)}$$

Hereby, the state derivative $\dot{\boldsymbol{x}}_{P,k}$ as well as the state $\boldsymbol{x}_{P,k}$ is estimated by employing an OFPS (see Appendix E.2). The data storage algorithm checks if new data points are sufficiently different to already stored data by employing equation (4-93). Other than that, the history stack is fixed and does not discard already stored data points.

Parameter	Numerical Value
General Parameters	
dt	$0.001s$
$\boldsymbol{A}_P$	$\boldsymbol{0}_{3x3}$
$\boldsymbol{B}_P$	$\begin{bmatrix} 0 & -1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ -1 & 1 & -1 & 1 \end{bmatrix}$
Adaptive Controller and Reference Model	
$\boldsymbol{A}_{rm}$	$\begin{bmatrix} -25 & 0 & 0 \\ 0 & -25 & 0 \\ 0 & 0 & -9 \end{bmatrix}$
$\boldsymbol{B}_{rm}$	$\begin{bmatrix} 25 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 9 \end{bmatrix}$
$[p_{B,rm}(t_0) \quad q_{B,rm}(t_0) \quad r_{B,rm}(t_0)]$	$\left[0\frac{^\circ}{s} \quad 0\frac{^\circ}{s} \quad 0\frac{^\circ}{s}\right]$
$\boldsymbol{\Gamma}_x$	$\begin{bmatrix} 5\boldsymbol{I}_{3x3} & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & 2\boldsymbol{I}_{3x3} \end{bmatrix}$
$\boldsymbol{\Gamma}_{rd}$	$5\boldsymbol{I}_{6x6}$
$\boldsymbol{Q} \mid \boldsymbol{P}$	$\boldsymbol{I}_{3x3} \middle| \begin{bmatrix} 0.02 & 0 & 0 \\ 0 & 0.02 & 0 \\ 0 & 0 & 0.0556 \end{bmatrix}$
σ	0.01
$\boldsymbol{\Theta}(t_0)$	$\boldsymbol{0}_{12x4}$

Table 6-1: List of controller parameters used in the quadcopter experiment

6.2.3 Quadcopter: Experimental Setup and Results

This section details the setup as well as the results of applying the proposed monitoring algorithm to the Quadcopter. The experiment runs for a total of $60s$. A human pilots the quadcopter with a remote control and issues rate commands as well as a thrust command. After approximately $15s$ after turning the system on, the pilot takes off. After take-off the pilot issues various rate commands to excite the system states and stimulate learning of the adaptive parameters. After approximately $45s$ the pilot shall keep the quadcopter steady and hovering over a single point.

The experiment uses the controller presented in section 6.2.2 onboard the quadcopter. The controller runs at a timestep of $0.001s$. The reference model states as well as the adaptive weights are initialized at zero. Table 6-1 summarizes the relevant controller parameters as well as the respective initial conditions for a concise overview. The controller parameters are largely adopted from [94].

For the purpose of this experiment the flight test as recorded in [150] is replayed. The proposed monitoring approach detailed in section 4 is applied. The replay runs on a standard desktop pc (i7-6700K @4.00GHz, 16GB RAM) within ‘real-time’. Note, that the author does not claim real-time performance as defined in computer science. Rather, the computation process of the monitor finishes within one time step on the specified hardware.

Parameter	Numerical Value
Prediction	
t_{pred}	$1s$
dt_{pred}	$0.001s$
GMM	
m (Number of mixture elements)	11
$\sigma_{i,0}$	0.03
Initial distribution of the mixture elements μ_i	Equidistant within 4σ of the initial state distribution
GP Regression	
p_H	100
σ_n^2	0.05
γ	0.0005
Λ_i	0.25
Indicator Function	
δ_ζ	0.7
α_{ind}	1
β_{ind}	1000
δ_{ind}	20
$p_{B,low}/p_{B,high}$	$-45\frac{\circ}{s}/45\frac{\circ}{s}$
$q_{B,low}/q_{B,high}$	$-45\frac{\circ}{s}/45\frac{\circ}{s}$
$r_{B,low}/r_{B,high}$	$-45\frac{\circ}{s}/45\frac{\circ}{s}$

Table 6-2: List of monitoring parameters used in the quadcopter experiment

The proposed monitoring algorithm is applied as described in section 6.2.2. The GMM used in this experiment consists of 11 mixture elements. The initial standard deviation of each mixture element is selected to be $\sigma_{i,0} = 0.03$. The mixture elements are equidistantly distributed in the range of 4σ of the initial state distribution. The latter in turn result from the state estimate of a KF, which is used on the state measurements. Note, that the forward KF is part of the OFPS and does not have to be implemented separately.

The online GP regression algorithm of section 4.4.3 with SE kernel functions is applied in order to form a belief on the uncertainty during the run time of the monitor. The data storage algorithm checks if new data points are sufficiently different to already stored data by employing equation (4-93). Other than that, the history stack is fixed and does not discard already stored data points. The bandwidth of the kernels is selected equally to be $\Lambda_i = 0.25$.

The prediction horizon is set to $t_{pred} = 1s$ with a time step of $\Delta t_{pred} = 0.001s$. This results in $k_{end} = 1000$ prediction steps. Note, that contrary to the simulation examples of section 5, not the complete experiment is predicted, but only a horizon of $t_{pred} = 1s$. Hence, the monitoring algorithm is called frequently. The prediction is triggered every second. Note, that the monitor could also be called in a higher frequency. The GMM is propagated through the linearized surrogate dynamics and the estimated uncertainty according to the update laws in (4-82) and (4-90).

The proposed monitoring algorithm assumes that the reference input is known over the prediction horizon. This is often the case if the reference is generated by an algorithm itself. However, here the reference command originates from a human pilot. As no pilot models are considered within this thesis, the exogenous reference command is assumed to be known through the replay of the experiment. In case that the monitor is applied directly and the command is unknown, the currently available exogenous reference command could also be extrapolated over the prediction horizon.

For the purpose of this experiment, the state limits are arbitrarily selected. Angular rates shall not exceed a limit of $(p_B, q_B, r_B)_{high} = 45\frac{°}{s}°$ and shall not fall below $(p_B, q_B, r_B)_{high} = -45\frac{°}{s}$. As a baseline, the parameters of the wing-rock example of section 5.1 are used for the indicator function. Since the prediction horizon is smaller, β_{ind} is chosen to be $\beta_{ind} = 1000$. Table 6-2 summarizes all parameters of the monitoring algorithm.

Figure 6-3 shows the reference command, the state of the reference model as well as the evolution of the state trajectories over the course of the experiment. The pilot excites the rotational rates within the first $45s$ of the experiment. It can be seen that significant tracking errors of up to $20°/s$ occur in the roll and pitch rate between the seconds 20 and 25. After that, the plant is seen to track the reference model with only minor deviation. The increase in tracking performance can be attributed to the evolution of the adaptive.

Figure 6-4 depicts the evolution of the adaptive parameters. After the switch to the adaptive controller, the weights start to evolve quickly in order to cancel the modelling uncertainty. This in turn is the reason for the improved tracking performance. It can be seen, that $\mathbf{\Theta}_r$ and $\mathbf{\Theta}_\varphi$ converge to certain numerical values within $40s$ and approximately remain at these values for the rest of the experiment. In contrast, $\mathbf{\Theta}_x$ is seen to still change at the end of the experiment. Still, the rate of change is seen to decrease. In contrast, $\mathbf{\Theta}_d$ is seen to oscillate during the whole experiment. In any case, the adaptive parameters stay bounded, which can be attributed to the $\sigma-$Modification in the update laws. Switching off the robustness modification quickly leads to instability ([94], [150]).

Figure 6-5 shows the estimate of the uncertainty for each input channel after 60 seconds into the experiment using the GP regression algorithm of section 4.4.3. The actual GP is dependent on the three rotational rates. In order to depict the estimate, here the yaw rate is kept constant at $r_B = 0°/s$. The resulting estimate is only dependent on the pitch and roll rate. It can be seen that the estimate of channel 2 and 3 as well as 1 and 4 assume a similar shape. The occurrence of similarities in neighbouring motor-propeller parts leads to the suggestion that one side of the quadcopter is effected by a source that increases the modelling uncertainty.

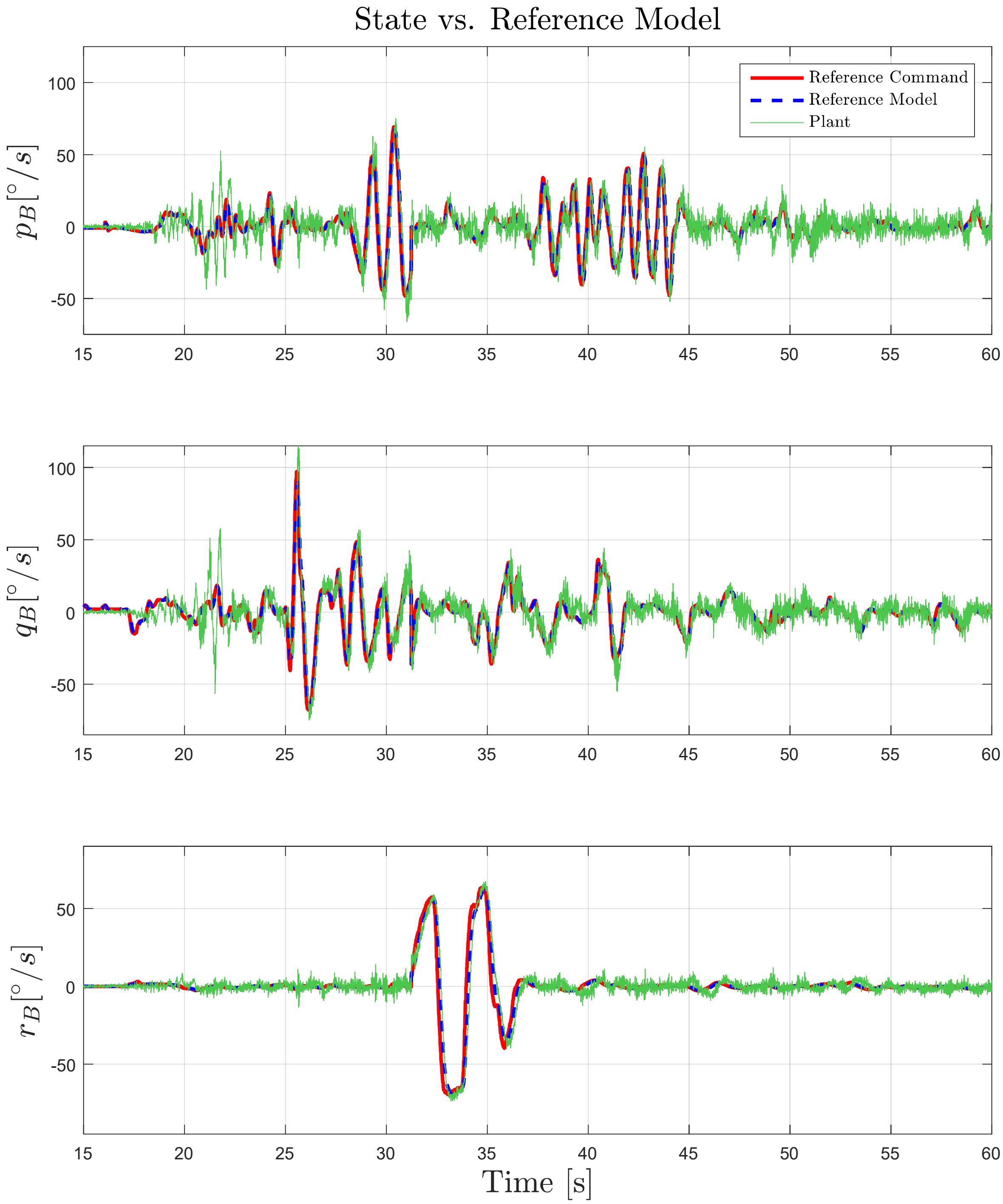


Figure 6-3: Tracking performance of the adaptive controller in the quadcopter experiment

Exemplary for the complete experiment, Figure 6-6 shows the predicted evolution of the pitch rate at 28, 29 and 30 seconds into the experiment. In particular, it depicts the predicted mean trajectory as well as the area in which 67%, 90% and 99% of all state trajectories are predicted to reside in at each time instant. The time instances were specifically selected to resemble the points in time at which the pilot issues roll rate commands. It can be seen that the predicted areas almost always contain the actually measured roll rate. This indicates, that the GMM based forecast is able to predict the roll rate well.

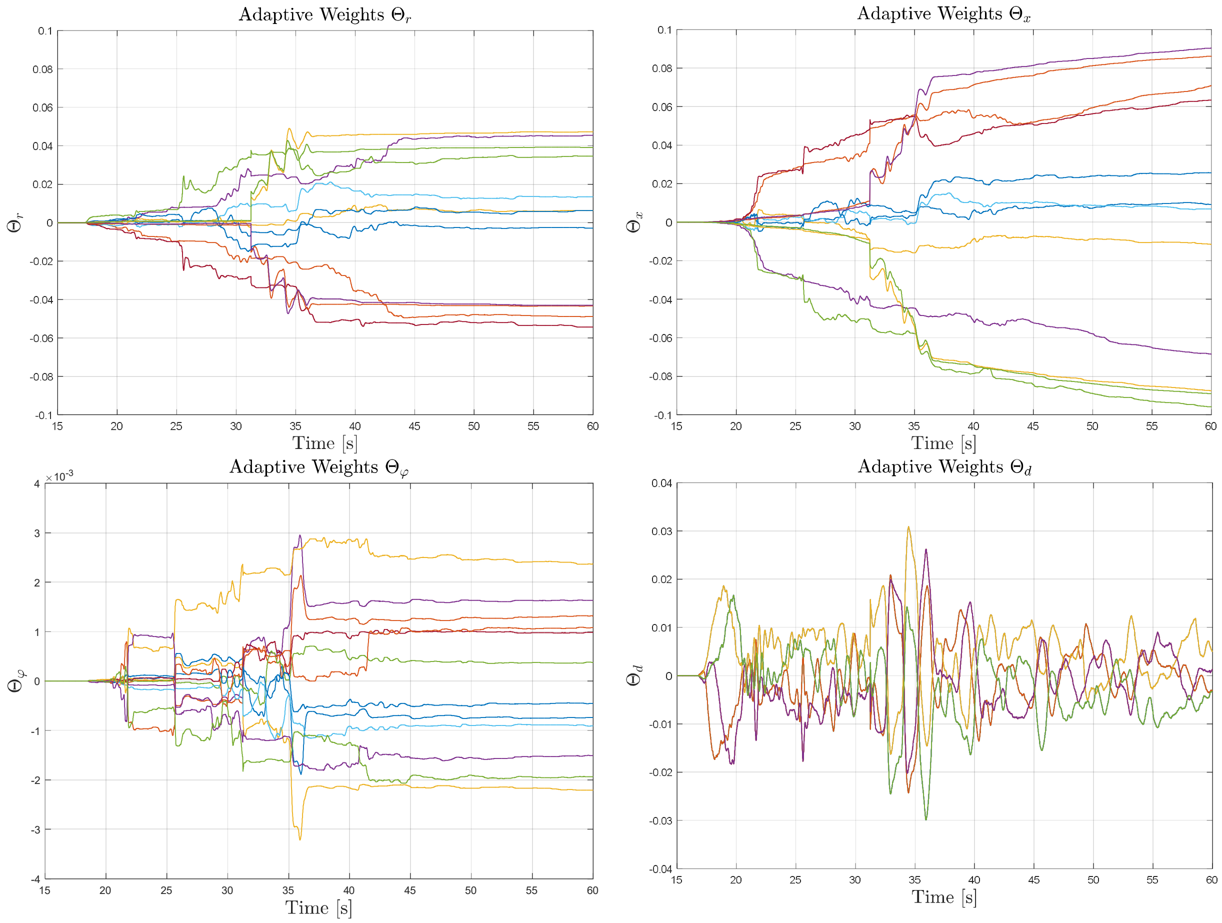


Figure 6-4: Evolution of the adaptive weights in the quadcopter experiment

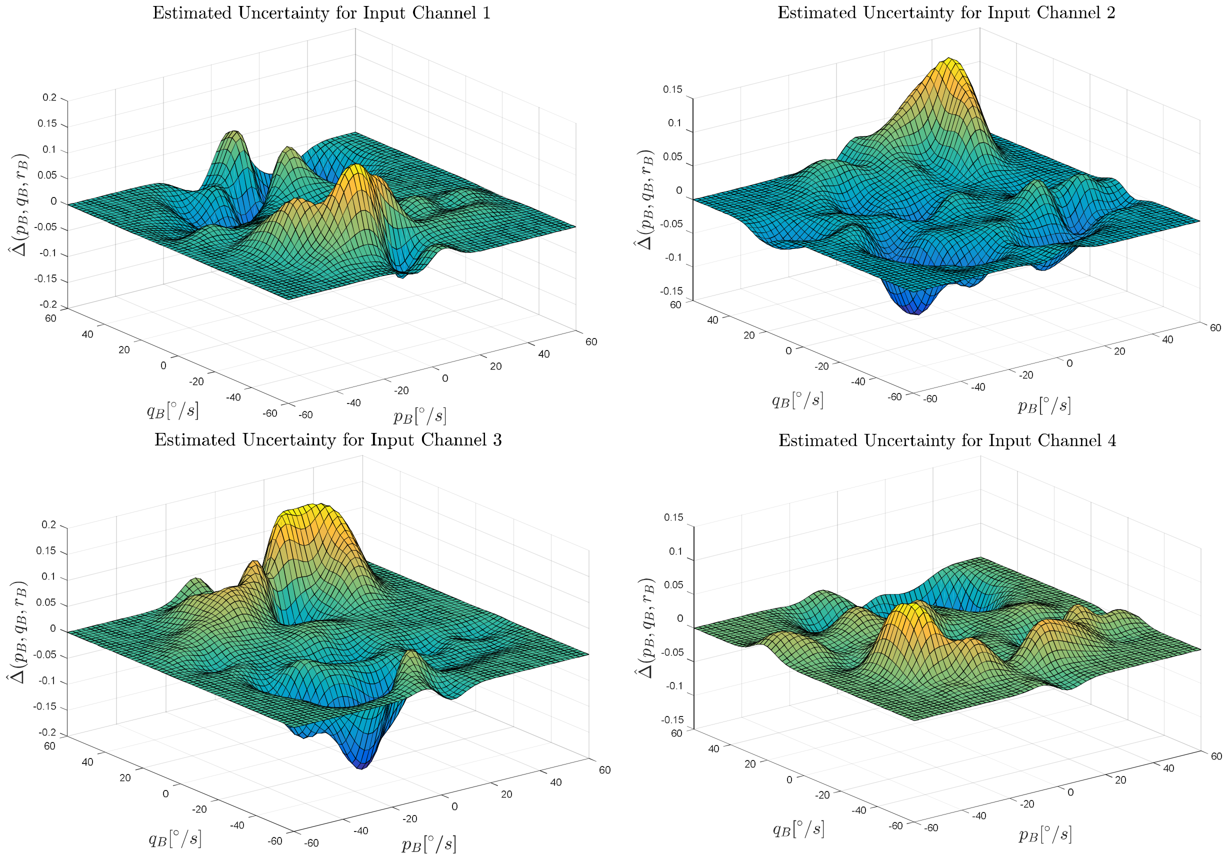


Figure 6-5: Uncertainty estimation result in the quadcopter experiment for a vanishing yaw rate

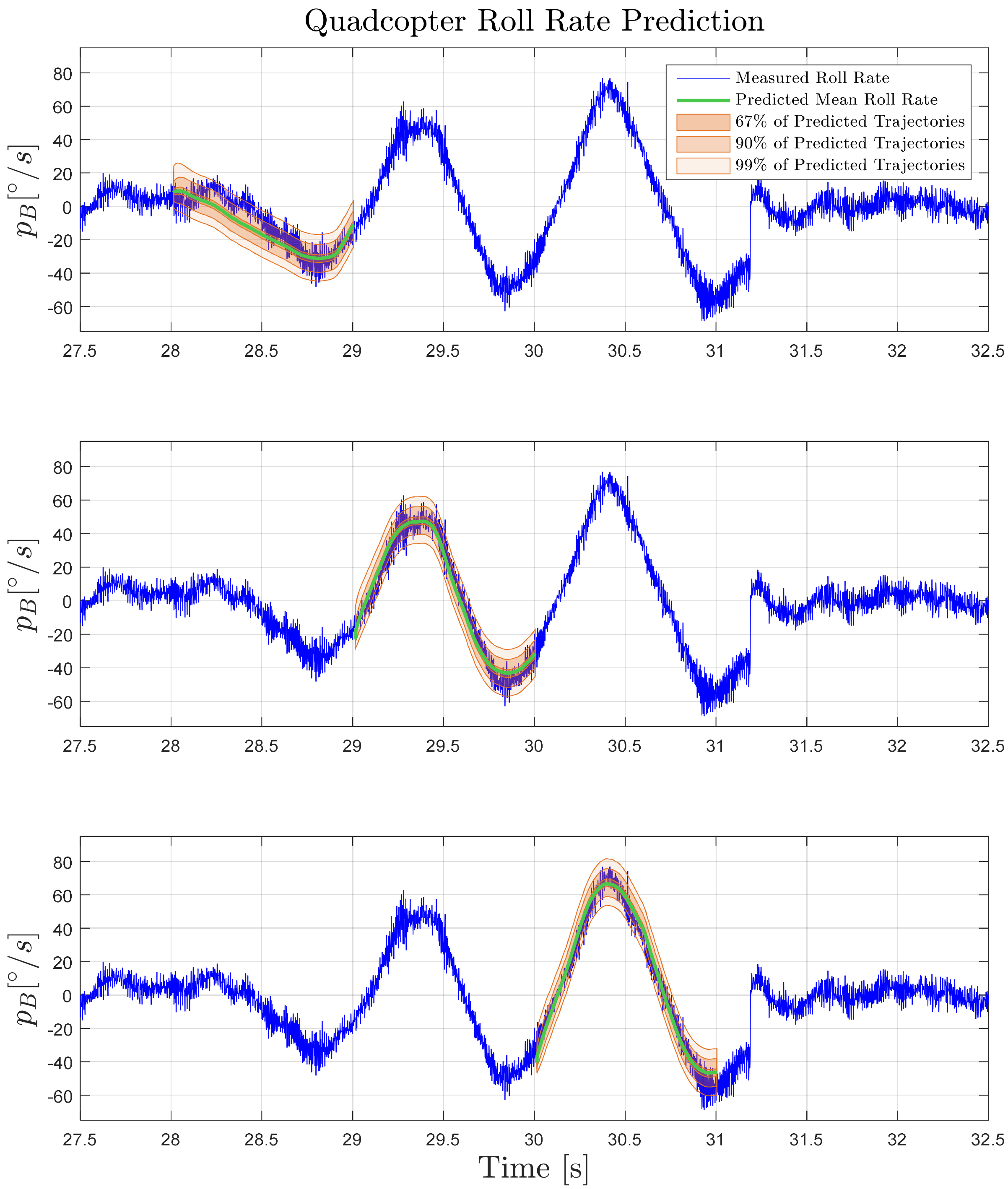


Figure 6-6: Selected predictions for the roll rate in the quadcopter experiment

Note, that the area the roll rate is predicted to reside in corresponds to the system state (i.e. $\boldsymbol{x_P}$). In Figure 6-6, it is compared to the state measurement (i.e. $\boldsymbol{y}$), as a ground truth is not available. In the presented experiment, the measurement is seen to reside within the predicted areas. This results from accounting for measurement noise in the forecast, but is not necessarily the case for each application. The theoretical derivation in section 4 does not lead to the result that the forecast distribution and the measurement distribution have to match. Hence, the distribution of the measurement could also exceed the predicted areas and the result would still be valid.

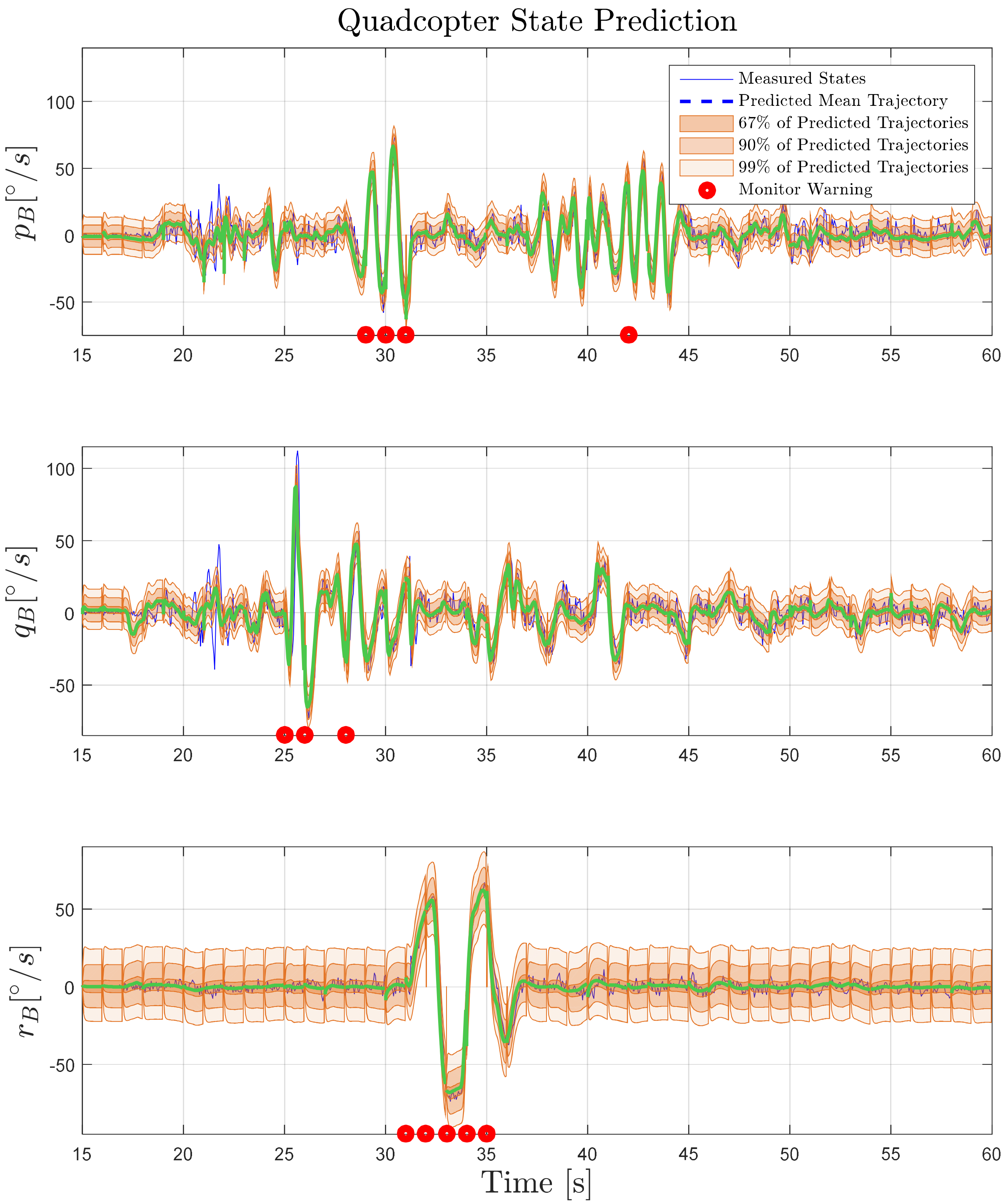


Figure 6-7: Predicted roll, pitch and yaw rate in the quadcopter experiment

For the sake of completeness, Figure 6-7 shows every prediction phase within the time horizon of the experiment. It can be seen that the forecast does not entirely encompass the roll and pitch rate between the seconds 20 and 25. This can be attributed to the GP regression algorithm, which is still recording data points in order to formulate a belief on the uncertainty. To counteract this, a history stack could be prerecorded. After approximately 23 seconds, the forecast is seen to encompass the measured state well. Note, that the area the yaw rate is predicted to reside in is larger and therefore more conservative than for the pitch and roll rate. This is attributed to the consideration of measurement noise in the forecast and a higher variance in the yaw axis of the posterior GP.

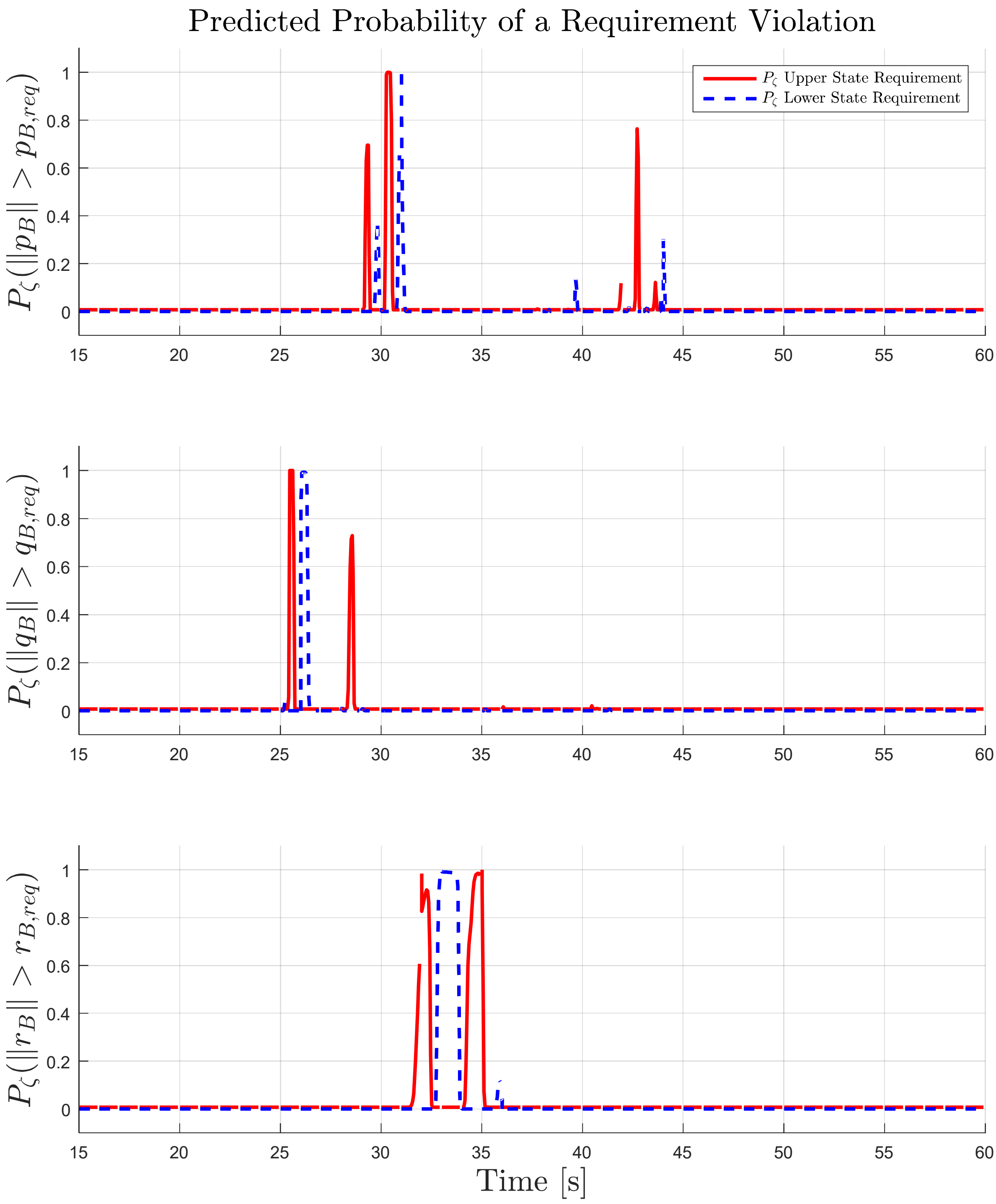


Figure 6-8: Probability of a angular rate requirement violation in the quadcopter experiment

Figure 6-8 depicts the predicted probability of angular rate requirement violation according to equation (4-102). It can be seen that the forecast accurately predicts the probable requirement violations during the pilot inputs. In particular, during seconds 25 and 30 for the pitch rate, during seconds 28 and 32 for the roll rate and during 30 and 36 for the yaw rate. As the prediction horizon as well as the prediction frequency is set to 1 second, the imminent requirement violation is predicted 1 second in advance. After the predicted distributions return within the state limits set by the requirements, the probability of a requirement violation reduces to zero.

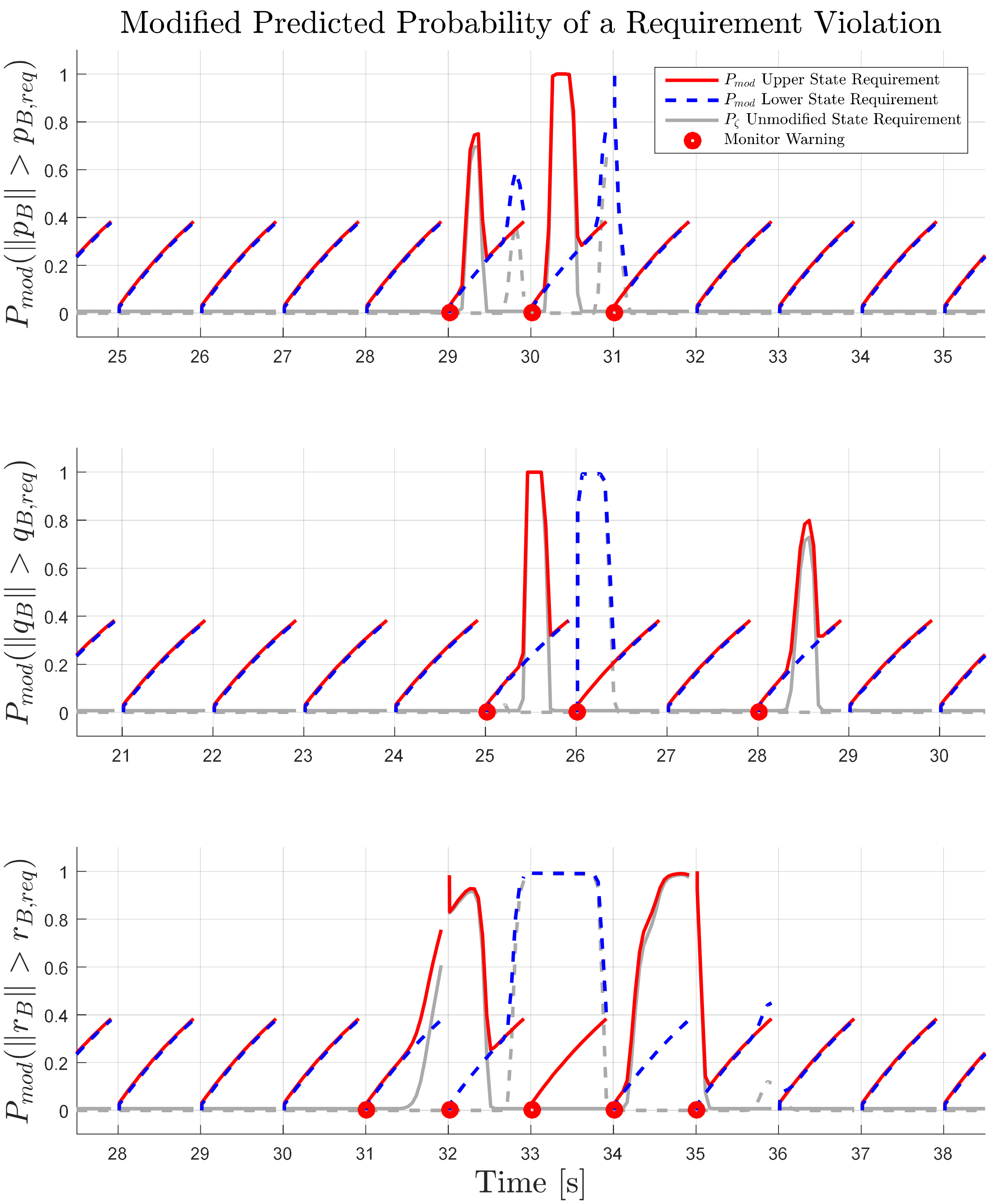


Figure 6-9: OVL modified predicted probability of an angular rate requirement violation in the quadcopter example

Finally, Figure 6-9 shows the predicted probability of a requirement violation P_{mod}, which is modified by the OVL according to (4-109). In particular, the time windows with the highest probabilities of requirement violations are highlighted. The resulting curves are seen to be more conservative. As the monitoring algorithm is called frequently, the OVL is reset to zero each time. This results in the spike pattern visible in Figure 6-9. In fact, with the parameter set of Table 6-2, the monitor issues a warning for all three angular rates. E.g. at second 29 the monitor issues an upper requirement violation of the roll rate within the next second, thus corroborating the results of the simulation examples in section 5.

At this example the importance of the monitoring frequency becomes visible. In the presented case the frequency of monitor calls is equal to the prediction horizon. As a result, if the actual requirement violation occurs only 0.1 seconds after the monitor is called, only approximately 0.1 seconds remain to react. In contrast, if the monitor runs on the same frequency as the controller or if the prediction horizon is doubled, approximately a full second remains for a reaction. However, these case also require more computational resources. In the end, the selection of frequency and horizon parameters heavily depends on the minimal reaction time required by the system in order to counter an imminent requirement violation. This again is task dependent, as errors in position probably allow for a larger reaction time compared to requirement violations in the angular rates.

In summary, the monitoring algorithm is seen to predict the evolution of the angular rates well. Therefore, it is able to detect imminent requirement violations before they appear, thus corroborating the results of the numerical simulation in section 5.

6.3 Hexacopter Application

This section details the application of the monitoring algorithm to the adaptively controlled hexacopter Firefly. The control architecture consists of a direct MRAC augmented feedback linearized rate controller as described in section 3. The plots shown in this section are down-sampled from their original frequency in a ratio of $5:1$.

The outline of this section is as follows: Section 6.3.1 details the control architecture used for the hexacopter. Section 6.3.2 presents the application of the monitoring algorithm. Finally, section 6.3.3 depicts the experimental setup and results.

6.3.1 Hexacopter: Control Architecture

The rotational dynamics of the hexacopter are given by (6-11) and (6-4). Consider the simplification $\boldsymbol{x}_P := \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B$ as well as the assumptions which were already used in section 6.2. Approximate feedback linearization (see i.a. [118], [95], [41], [114], [149]) transforms the rotational dynamics into the form (3-2). Therefore consider the linearizing feedback

$$\boldsymbol{u} = \widehat{\boldsymbol{C}}_{Mu}^{+}\left[\left(\hat{\mathbf{I}}_{BB}^{G}\right)\boldsymbol{\nu} + \boldsymbol{x}_P \times \left[\left(\hat{\mathbf{I}}_{BB}^{G}\right)\cdot\boldsymbol{x}_P\right]\right] \tag{6-33}$$

Here, $\widehat{\boldsymbol{C}}_{Mu}$, $\hat{\mathbf{I}}_{BB}^{G}$ and $\widehat{\boldsymbol{C}}_{M\omega}$ represent prior estimates of the parameters $\boldsymbol{C}_{Mu}$, $\mathbf{I}_{BB}^{G}$ and $\boldsymbol{C}_{M\omega}$. Furthermore, $\boldsymbol{\nu}$ represents the pseudo control input. Inserting (6-33) into (6-11) yields

$$\begin{aligned}\dot{\boldsymbol{x}}_P = {} & \left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{C}_{M\omega}\boldsymbol{x}_P + \left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{C}_{Mu}\widehat{\boldsymbol{C}}_{Mu}^{+}\left(\hat{\mathbf{I}}_{BB}^{G}\right)\boldsymbol{\nu} + \left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{C}_{Mu}\widehat{\boldsymbol{C}}_{Mu}^{+}\boldsymbol{x}_P \times \left[\left(\hat{\mathbf{I}}_{BB}^{G}\right)\cdot\boldsymbol{x}_P\right] \\ & -\left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{x}_P \times \left[\left(\mathbf{I}_{BB}^{G}\right)\cdot\boldsymbol{x}_P\right]\end{aligned} \tag{6-34}$$

Expanding the input term of equation (6-11) as shown in section 6.2.1 yields

$$\left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{C}_{Mu}=\begin{bmatrix}\frac{-0.5lk_T}{I_{xx}} & \frac{-lk_T}{I_{xx}} & \frac{-0.5lk_T}{I_{xx}} & \frac{0.5lk_T}{I_{xx}} & \frac{lk_T}{I_{xx}} & \frac{0.5lk_T}{I_{xx}}\\ \frac{\sqrt{3}}{2I_{yy}}lk_T & 0 & -\frac{\sqrt{3}}{2I_{yy}}lk_T & -\frac{\sqrt{3}}{2I_{yy}}k_T & 0 & \frac{\sqrt{3}}{2I_{yy}}k_T\\ \frac{k_M}{I_{zz}} & -\frac{k_M}{I_{zz}} & \frac{k_M}{I_{zz}} & -\frac{k_M}{I_{zz}} & \frac{k_M}{I_{zz}} & -\frac{k_M}{I_{zz}}\end{bmatrix}. \tag{6-35}$$

Similar to section 6.2.1 and [94], the product $\left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{C}_{Mu}$ can be decomposed into

$$\left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{C}_{Mu}=\underbrace{\begin{bmatrix}-0.5 & -1 & -0.5 & 0.5 & 1 & 0.5\\ \frac{\sqrt{3}}{2} & 0 & -\frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} & 0 & \frac{\sqrt{3}}{2}\\ 1 & -1 & 1 & -1 & 1 & -1\end{bmatrix}}_{\widehat{\boldsymbol{B}}_P}\boldsymbol{\Lambda}=\widehat{\boldsymbol{B}}_P\boldsymbol{\Lambda} \tag{6-36}$$

Note, that the right More-Penrose pseudoinverse of $\widehat{\boldsymbol{B}}_P\boldsymbol{\Lambda}$ exists, as $\left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{C}_{Mu}$ has linearly independent rows. As a consequence, $\widehat{\boldsymbol{B}}_P\boldsymbol{\Lambda}\left(\widehat{\boldsymbol{B}}_P\boldsymbol{\Lambda}\right)^{+}=\boldsymbol{I}$. Furthermore, consider the cross product $\left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{x}_P\times\left[\left(\mathbf{I}_{BB}^{G}\right)\cdot\boldsymbol{x}_P\right]$ as solved in (6-14). Adding and subtracting $\boldsymbol{\nu}$ in (6-34), as well as collecting terms results in

$$\begin{gathered}\dot{\boldsymbol{x}}_P=\boldsymbol{\nu}+\left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{C}_{M\omega}\boldsymbol{x}_P-{\boldsymbol{\Theta}_{\boldsymbol{\varphi},\boldsymbol{I}}^{*}}^{T}\boldsymbol{\varphi}(\boldsymbol{x}_P)\\ -\left[\boldsymbol{I}-\widehat{\boldsymbol{B}}_P\boldsymbol{\Lambda}\left(\widehat{\boldsymbol{B}}_P\widehat{\boldsymbol{\Lambda}}\right)^{+}\right]\boldsymbol{\nu}+\widehat{\boldsymbol{B}}_P\boldsymbol{\Lambda}\left(\widehat{\boldsymbol{B}}_P\widehat{\boldsymbol{\Lambda}}\right)^{+}{\widehat{\boldsymbol{\Theta}}_{\boldsymbol{\varphi},\boldsymbol{I}}^{*}}^{T}\boldsymbol{\varphi}(\boldsymbol{x}_P)\end{gathered} \tag{6-37}$$

Here, ${\widehat{\boldsymbol{\Theta}}_{\boldsymbol{\varphi},\boldsymbol{I}}^{*}}^{T}$ denotes the parameter matrix if (6-14) is solved with $\left(\hat{\mathbf{I}}_{BB}^{G}\right)$ instead of $\left(\mathbf{I}_{BB}^{G}\right)$. Furthermore, let $\widehat{\boldsymbol{B}}_P\widehat{\boldsymbol{\Lambda}}$ represent the result of decomposing $\left(\hat{\mathbf{I}}_{BB}^{G}\right)^{-1}\widehat{\boldsymbol{C}}_{Mu}$ instead of $\left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{C}_{Mu}$. Consider the following definitions

$$\begin{gathered}{\boldsymbol{\Theta}_{x}^{*}}^{T}=-\left(\left(\mathbf{I}_{BB}^{G}\right)^{-1}\boldsymbol{C}_{M\omega}\right)^{T}\\ {\boldsymbol{\Theta}_{\boldsymbol{\varphi}}^{*}}^{T}={\boldsymbol{\Theta}_{\boldsymbol{\varphi},\boldsymbol{I}}^{*}}^{T}-\widehat{\boldsymbol{B}}_P\boldsymbol{\Lambda}\left(\widehat{\boldsymbol{B}}_P\widehat{\boldsymbol{\Lambda}}\right)^{+}{\widehat{\boldsymbol{\Theta}}_{\boldsymbol{\varphi},\boldsymbol{I}}^{*}}^{T}\\ {\boldsymbol{\Theta}_{\nu}^{*}}^{T}=\left[\boldsymbol{I}-\widehat{\boldsymbol{B}}_P\boldsymbol{\Lambda}\left(\widehat{\boldsymbol{B}}_P\widehat{\boldsymbol{\Lambda}}\right)^{+}\right]^{T}.\end{gathered} \tag{6-38}$$

Furthermore, define $\boldsymbol{A}_P\coloneqq\boldsymbol{0}_{3x3}$ and $\boldsymbol{B}_P\coloneqq\boldsymbol{I}_{3x3}$. Then (6-37) becomes

$$\begin{aligned}\dot{\boldsymbol{x}}_P&=\boldsymbol{A}_P\boldsymbol{x}_P+\boldsymbol{B}_P\left(\boldsymbol{\nu}\underbrace{-\boldsymbol{\Theta}_{x}^{*}\boldsymbol{x}_P-\boldsymbol{\Theta}_{\boldsymbol{\varphi}}^{*}\boldsymbol{\varphi}(\boldsymbol{x}_P)-{\boldsymbol{\Theta}_{\nu}^{*}}^{T}\boldsymbol{\nu}}_{-\boldsymbol{\Delta}(\boldsymbol{x}_P,\boldsymbol{\nu})}\right)\\ &=\boldsymbol{A}_P\boldsymbol{x}_P+\boldsymbol{B}_P\left(\boldsymbol{\nu}-\boldsymbol{\Delta}(\boldsymbol{x}_P,\boldsymbol{\nu})\right),\end{aligned} \tag{6-39}$$

which in turn resembles the form (3-2). The subsequent error and adaptive controller is designed according to (3-11) and (3-12). Therefore, consider the reference model (6-12) used in the quadcopter example in section 6.2.1.

Applying (3-11) and (3-12) to (6-39) yields

$$\dot{\boldsymbol{x}}_P = \boldsymbol{A}_P\boldsymbol{x}_P + \boldsymbol{B}_P\boldsymbol{K}_e^T\boldsymbol{e} + \boldsymbol{B}_P\boldsymbol{K}_{rm,x}^T\boldsymbol{x}_{rm} + \boldsymbol{B}_P\boldsymbol{K}_{rm,r}^T\boldsymbol{r} + \boldsymbol{B}_P\big(\boldsymbol{\nu}_{ad} - \boldsymbol{\Delta}(\boldsymbol{x}_P,\boldsymbol{\nu})\big). \tag{6-40}$$

The adaptive control signal $\boldsymbol{\nu}_{ad}$ is designed to cancel the uncertain terms in the brackets (6-40). Note, that $\boldsymbol{\Delta}(\boldsymbol{x}_P,\boldsymbol{\nu})$ depends on the pseudo-control $\boldsymbol{\nu}$. In turn, the pseudo-control primarily depends on the system states $\boldsymbol{x}_P$. Hence, in order to reduce the complexity of the controller, the uncertainty is assumed to only depend on the system states $\boldsymbol{x}_P$. The adaptive control signal is constructed to be

$$\boldsymbol{\nu}_{ad} = \boldsymbol{\Theta}_x^T\boldsymbol{x}_P + \boldsymbol{\Theta}_\varphi^T\boldsymbol{\varphi}(\boldsymbol{x}_P) = \underbrace{\begin{bmatrix}\boldsymbol{\Theta}_x^T & \boldsymbol{\Theta}_\varphi^T\end{bmatrix}}_{\boldsymbol{\Theta}^T}\underbrace{\begin{bmatrix}\boldsymbol{x}_P \\ \boldsymbol{\varphi}(\boldsymbol{x}_P)\end{bmatrix}}_{\boldsymbol{\Phi}} = \boldsymbol{\Theta}^T\boldsymbol{\Phi}(\boldsymbol{x}_P). \tag{6-41}$$

The parameter update laws follow (3-19) and use a σ-modification with $k(\boldsymbol{\Theta}) = -\sigma\boldsymbol{\Theta}$, such that

$$\dot{\boldsymbol{\Theta}} = -\boldsymbol{\Gamma}\boldsymbol{\Phi}(\boldsymbol{x}_P)\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}\sigma\boldsymbol{\Theta}. \tag{6-42}$$

In the presented experiment, the adaptive controller is only added to the roll and pitch rate channel. In contrast, the yaw rate only depends on the baseline controller. In addition to the rotational dynamics, also the attitude dynamics with respect to Euler angles and the body-fixed rotational rates $\left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B$ are inverted. This allows the pilot to control the attitude instead of the rotational rates, which in turn requires more flight training. The attitude dynamics describe a purely kinematic relation and are therefore not subject to modelling uncertainties. The derivation is omitted here, but can be found i.e. in [92]. Furthermore, instead of commanding a yaw angle Ψ, the pilot commands the derivative $\dot{\Psi}$. This allows the pilot to keep the control stick in an idle position instead of having to command a fixed yaw angle throughout the experiment. The rotational dynamics and the associated controller as presented in this section are thereby unaffected.

6.3.2 Hexacopter: Application of the Monitoring Algorithm

This section applies the monitoring approach of section 4 to the hexacopter system dynamics in (6-40). In the following, first the linear surrogate model is constructed. In a second step, the setup of the GP regression algorithm is detailed.

Construction of the linear surrogate model

In order to propagate the GMM, first the linearized system dynamics need to be derived. Bringing the known dynamics of (6-40) and (6-42) in the form of (4-6) and (4-11) yields

$$\boldsymbol{f}(\boldsymbol{x}_P,\boldsymbol{r},\boldsymbol{\Theta}) = \begin{bmatrix}\boldsymbol{f}_x(\boldsymbol{x}_P,\boldsymbol{r},\boldsymbol{\Theta}) \\ \boldsymbol{f}_\Theta(\boldsymbol{x}_P,\boldsymbol{r},\boldsymbol{\Theta})\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}_e\boldsymbol{x}_P + \boldsymbol{B}_P\big(\boldsymbol{K}_{rm,x}^T - \boldsymbol{K}_e^T\big)\boldsymbol{x}_{rm} + \boldsymbol{B}_P\boldsymbol{K}_{rm,r}^T\boldsymbol{r} + \boldsymbol{B}_P\boldsymbol{\Theta}^T\boldsymbol{\Phi} \\ -\boldsymbol{\Gamma}\boldsymbol{\Phi}(\boldsymbol{x}_P)\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}_P - \boldsymbol{\Gamma}\sigma\boldsymbol{\Theta}\end{bmatrix}. \tag{6-43}$$

The exogenous reference input $\boldsymbol{r}$ is supplied by a human pilot. Since this thesis does not consider pilot models, the reference input $\boldsymbol{r}$ is considered to be constant over the prediction horizon.

Following, the steps in section 4.3, evaluating $\boldsymbol{f}(\boldsymbol{x}_{P,l},\boldsymbol{r},\boldsymbol{\Theta}_l)$ at the linearization point $\boldsymbol{x}_P=\boldsymbol{x}_{P,l}$ and $\boldsymbol{\Theta}=\boldsymbol{\Theta}_l$ gives

$$\begin{aligned}\boldsymbol{f}(\boldsymbol{x}_{P,l},\boldsymbol{r},\boldsymbol{\Theta}_l)&=\begin{bmatrix}\boldsymbol{f}_{x,l}(\boldsymbol{x}_{P,l},\boldsymbol{r},\boldsymbol{\Theta}_l)\\ \boldsymbol{f}_{\Theta_l}(\boldsymbol{x}_{P,l},\boldsymbol{r},\boldsymbol{\Theta}_{x\varphi,l})\end{bmatrix}\\ &=\begin{bmatrix}\boldsymbol{A}_e\boldsymbol{x}_{P,l}+\boldsymbol{B}_P(\boldsymbol{K}_{rm,x}^T-\boldsymbol{K}_e^T)\boldsymbol{x}_{rm,l}+\boldsymbol{B}_P\boldsymbol{K}_{rm,r}^T\boldsymbol{r}+\boldsymbol{B}_P\boldsymbol{\Theta}_l^T\boldsymbol{\Phi}(\boldsymbol{x}_{P,l})\\ -\boldsymbol{\Gamma}\boldsymbol{\Phi}(\boldsymbol{x}_{P,l})\boldsymbol{e}_l^T\boldsymbol{P}\boldsymbol{B}_P-\boldsymbol{\Gamma}\sigma\boldsymbol{\Theta}_l\end{bmatrix}.\end{aligned} \tag{6-44}$$

In order to construct the matrix $\boldsymbol{A}_{lin}$, the partial derivatives of $\boldsymbol{f}_x$,and $\boldsymbol{f}_\Theta$ with respect to the system states $\boldsymbol{x}_P$ and the controller states $\boldsymbol{\Theta}$ need to be derived. The partial derivatives of $\boldsymbol{f}_x$ with respect to the system states $\boldsymbol{x}_P$ and the controller states $\boldsymbol{\Theta}$ are

$$\begin{gathered}\left.\frac{\partial\boldsymbol{f}_x}{\partial\boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}}=\boldsymbol{A}_e+\boldsymbol{B}_P\boldsymbol{\Theta}^T\left.\frac{\partial\boldsymbol{\Phi}(\boldsymbol{x}_P)}{\partial\boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}},\\ \left.\frac{\partial\boldsymbol{f}_x}{\partial\boldsymbol{\Theta}}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}}=\boldsymbol{B}_P\boldsymbol{\Phi}^T(\boldsymbol{x}_{P,l}).\end{gathered} \tag{6-45}$$

The partial derivate $\left.\frac{\partial\boldsymbol{\Phi}(\boldsymbol{x}_P)}{\partial\boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}}$ is

$$\left.\begin{bmatrix}\frac{\partial\boldsymbol{\Phi}(\boldsymbol{x}_P)}{\partial p_B} & \frac{\partial\boldsymbol{\Phi}(\boldsymbol{x}_P)}{\partial q_B} & \frac{\partial\boldsymbol{\Phi}(\boldsymbol{x}_P)}{\partial r_B}\end{bmatrix}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}}=\begin{bmatrix}1 & 0 & 0 & 0 & r_{B,l} & q_{B,l}\\ 0 & 1 & 0 & p_{B,l} & 0 & q_{B,l}\\ 0 & 0 & 1 & p_{B,l} & r_{B,l} & 0\end{bmatrix}^T. \tag{6-46}$$

Similarly, the partial derivatives of $\boldsymbol{f}_\Theta$ with respect to the system states $\boldsymbol{x}_P$ and the controller states $\boldsymbol{\Theta}$ are

$$\begin{gathered}\left.\frac{\partial\boldsymbol{f}_\Theta}{\partial\boldsymbol{x}_P}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}}=-\boldsymbol{\Gamma}\left.\frac{\partial\boldsymbol{\Phi}(\boldsymbol{x}_P)}{\partial\boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}}\boldsymbol{x}_{P,l}^T\boldsymbol{P}\boldsymbol{B}_P-\boldsymbol{\Gamma}\boldsymbol{\Phi}(\boldsymbol{x}_{P,l})\boldsymbol{B}_P^T\boldsymbol{P}+\boldsymbol{\Gamma}\left.\frac{\partial\boldsymbol{\Phi}(\boldsymbol{x}_P)}{\partial\boldsymbol{x}_P}\right|_{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}}\boldsymbol{x}_{rm}^T\boldsymbol{P}\boldsymbol{B}_P,\\ \left.\frac{\partial\boldsymbol{f}_\Theta}{\partial\boldsymbol{\Theta}}\right|_{\substack{\boldsymbol{x}_P=\boldsymbol{x}_{P,l}\\ \boldsymbol{\Theta}=\boldsymbol{\Theta}_l}}=-\boldsymbol{\Gamma}\sigma.\end{gathered} \tag{6-47}$$

Application of GP regression to the hexacopter

The monitoring algorithm uses GP regression as described in section 4.4.3 in order to approximate the modelling uncertainty online. Therefore, the latter needs to be estimated at specific points in time. In order to reduce complexity, the uncertainty $\boldsymbol{\Delta}$ is assumed to solely depend on the states $\boldsymbol{x}_P$. This simplification is warranted, as the control input $\boldsymbol{\nu}$ directly depends on the plant states itself. Rearranging equation (6-40) according to (4-92) yields

$$\widehat{\boldsymbol{\Delta}}(\widehat{\boldsymbol{x}}_{P,k})=\boldsymbol{B}_P^+\left(\boldsymbol{A}_e\widehat{\boldsymbol{x}}_{P,k}+\boldsymbol{B}_P(\boldsymbol{K}_{rm,x}^T-\boldsymbol{K}_e^T)\boldsymbol{x}_{rm,k}+\boldsymbol{B}_P\boldsymbol{K}_{rm,r}^T\boldsymbol{r}_k+\boldsymbol{B}_P\boldsymbol{\Theta}_k^T\boldsymbol{\Phi}(\widehat{\boldsymbol{x}}_{P,k})-\dot{\widehat{\boldsymbol{x}}}_{P,k}\right). \tag{6-48}$$

Hereby, the state derivative $\dot{\boldsymbol{x}}_{P,k}$ as well as the state $\boldsymbol{x}_{P,k}$ are estimated by employing an OFPS (see Appendix E.2). The data storage algorithm checks if new data points are sufficiently different to already stored data by employing equation (4-93). Other than that, the history stack is fixed and does not discard already stored data points.

Parameter	Numerical Value
General Parameters	
dt	$0.003s$
$\boldsymbol{A}_P$	$\boldsymbol{0}_{3x3}$
$\boldsymbol{B}_P$	$\boldsymbol{I}_{3x3}$
Controller and Reference Model	
$\boldsymbol{A}_{rm}$	$-50\boldsymbol{I}_{3x3}$
$\boldsymbol{B}_{rm}$	$50\boldsymbol{I}_{3x3}$
$[p_{B,rm}(t_0) \quad q_{B,rm}(t_0) \quad r_{B,rm}(t_0)]$	$\left[0\frac{\circ}{s} \quad 0\frac{\circ}{s} \quad 0\frac{\circ}{s}\right]$
$\boldsymbol{K}_e$	$\begin{bmatrix} -30 & 0 & 0 \\ 0 & -30 & 0 \\ 0 & 0 & -10 \end{bmatrix}$
$\boldsymbol{\Gamma}_x, \boldsymbol{\Gamma}_\varphi$	$\begin{bmatrix} -2 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} -2 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
$\boldsymbol{Q} \mid \boldsymbol{P}$	$\boldsymbol{I}_{3x3} \vert \begin{bmatrix} 0.017 & 0 & 0 \\ 0 & 0.017 & 0 \\ 0 & 0 & 0.050 \end{bmatrix}$
σ	0.01
$\boldsymbol{\Theta}(t_0)$	$\boldsymbol{0}_{6x3}$

Table 6-3: List of controller parameters used in the hexacopter experiment

6.3.3 Hexacopter: Experimental Setup and Results

This section details the setup as well as the results of applying the proposed monitoring algorithm to the Hexacopter. The experiment runs for a total of $60s$. A human pilots the hexacopter with a remote control and issues roll angle, pitch angle and yaw rate commands as well as a thrust command. After approximately $120s$ after turning the system on, the pilot takes off. After take-off the pilot issues various commands to excite the system states and stimulate learning of the adaptive parameters.

The experiment uses the controller presented in section 6.3.2 onboard the hexacopter. Therefore, the hexacopter is equipped with an Overo Gumstix (1GHz, 512MB RAM). The controller runs with a timestep of $dt = 0.003s$. The reference model states as well as the adaptive weights are initialized at zero. Table 6-3 summarizes the relevant controller parameters as well as the respective initial conditions for a concise overview.

The proposed monitoring approach detailed in section 4 is applied. The monitoring algorithm runs on the Overo Gumstix board with the same frequency as the control algorithm. In particular, this means that the monitor is called at each time step. The proposed monitoring algorithm as described in section 4 is applied. The GMM used in this experiment consists of 11 mixture elements. The initial standard deviation of each mixture element is selected to be $\sigma_{i,0} = 0.03$. The mixture elements are equidistantly distributed in the range of 4σ of the initial state distribution. The latter in turn result from the state estimate of a KF, which is used on the state measurements. Note, that the forward KF is part of the OFPS and does not have to be implemented separately.

Parameter	Numerical Value
Prediction	
t_{pred}	1
dt_{pred}	$0.01s$
GMM	
m (Number of mixture elements)	11
$\sigma_{i,0}$	0.03
Initial distribution of the mixture elements μ_i	Equidistant within 4σ of the initial state distribution
GP Regression	
p_H	100
σ_n^2	0.05
γ	0.0005
Λ_i	0.25
Indicator Function	
δ_ζ	0.7
α_{ind}	1
β_{ind}	100
δ_{ind}	20
$p_{B,low}/p_{B,high}$	$-15\frac{\circ}{s}/15\frac{\circ}{s}$
$q_{B,low}/q_{B,high}$	$-15\frac{\circ}{s}/15\frac{\circ}{s}$
$r_{B,low}/r_{B,high}$	$-15\frac{\circ}{s}/15\frac{\circ}{s}$

Table 6-4: List of monitoring parameters used in the hexacopter experiment

The online GP regression algorithm of section 4.4.3 with SE kernel functions is applied in order to form a belief on the uncertainty. For the purpose of this experiment, data points were already recorded during a previous flight. For the experiment, the history stack is fixed. The bandwidth of the kernels is selected equally to be $\Lambda_i = 0.25$.

The prediction horizon is set to $t_{pred} = 1s$ with a time step of $dt_{pred} = 0.01s$. This results in $k_{end} = 100$ prediction steps. A preliminary analysis on the run-time requirements of the monitor resulted in a maximum of $k_{end,max} = 10$ prediction steps per call, which would still ensure that both the monitor and controller terminate within $\Delta t = 0.003s$ on the Overo Gumstix board.

Multiple options exist, which can extend the prediction horizon without requiring a change of hardware. First, the time step for the prediction dt_{pred} can be increased. As a result, the prediction horizon increases with the same ratio. As an example, setting $dt_{pred} = 0.1s$ results in a prediction horizon of $t_{pred} = 1s$, while still only requiring 10 prediction steps. Note, that the predictions are performed using Euler's method. Thus, increasing the step size also quickly increases the error of the prediction. This approach is applicable to a certain degree, but has to be checked a-priori. In the present example, the prediction step $dt_{pred} = 0.01s$ is already greater than the timestep of the controller with $dt = 0.003s$. The monitor was found to still perform with only minor error. Still, an increase to $dt_{pred} = 0.1s$ was found to introduce too much error. As a consequence, this approach is not pursued.

Secondly, parts of the monitor can be approximated. In this, instead of iterating 10 time steps, a look-up table for the results of the iterations can be formed. This is possible since the 'known' dynamics do not change over the scope of the experiment. Hence, the table look-up retains validity over the scope of the experiment. This second approach promises to solve the problems of limited computational resource and is used instead of increasing the step size beyond $dt_{pred} = 0.01s$.

For the construction of the look-up tables the expected state space is partitioned. Each look-up table receives the system states as an input and produces the predicted mean and variance of a mixture element. Each look-up table represents 10 prediction steps. Constructing a table lookup for the full prediction of 100 steps is also possible. However, the GMM is only analysable at the output of a table lookup. As a result, mapping 100 prediction steps only allows for a GMM evaluation at the end of the prediction horizon and not in between. As a logical consequence, from selecting a table lookup horizon of 10, the GMM is evaluated after every $10th$ prediction step within this experiment. Note that this method is applicable to the hexacopter case, as the number of states is limited. For an increasing system state dimension, also the effort in creating and running table-lookups increases.

For the sake of completion, two other options exist. First, the hardware of the hexacopter could be upgraded. Secondly, the monitoring algorithm could also be applied to a ground station with sufficient computational ressources.

A fundamental hypothesis of the proposed monitoring algorithm is that the reference input is known over the prediction horizon. This is often the case if the reference is generated by an algorithm, such as a trajectory generator. However, here the reference command originates from a human pilot. As no pilot models are considered within this thesis, the exogenous attitude reference command is assumed to be constant over the prediction horizon. The attitude reference model is then used to form an expected rate command over the prediction horizon. As a consequence, the monitor predictions exhibit an error to the real exogenous command and resulting system states. To show this effect, the experiment is also replayed while assuming the exogenouse reference command to be known throughout the replay.

For the purpose of this experiment, the state limits are arbitrarily selected. Angular rates shall not exceed a limit of $(p_B, q_B, r_B)_{high} = 15\frac{°}{s}°$ and shall not fall below $(p_B, q_B, r_B)_{high} = -15\frac{°}{s}$. As a baseline, the parameters of the wing-rock example of section 5.1 as well as the quadcopter example of section 6.2 are used for the indicator function. The parameter β_{ind} is chosen to be $\beta_{ind} = 100$. Table 6-4 summarizes all parameters of the monitoring algorithm.

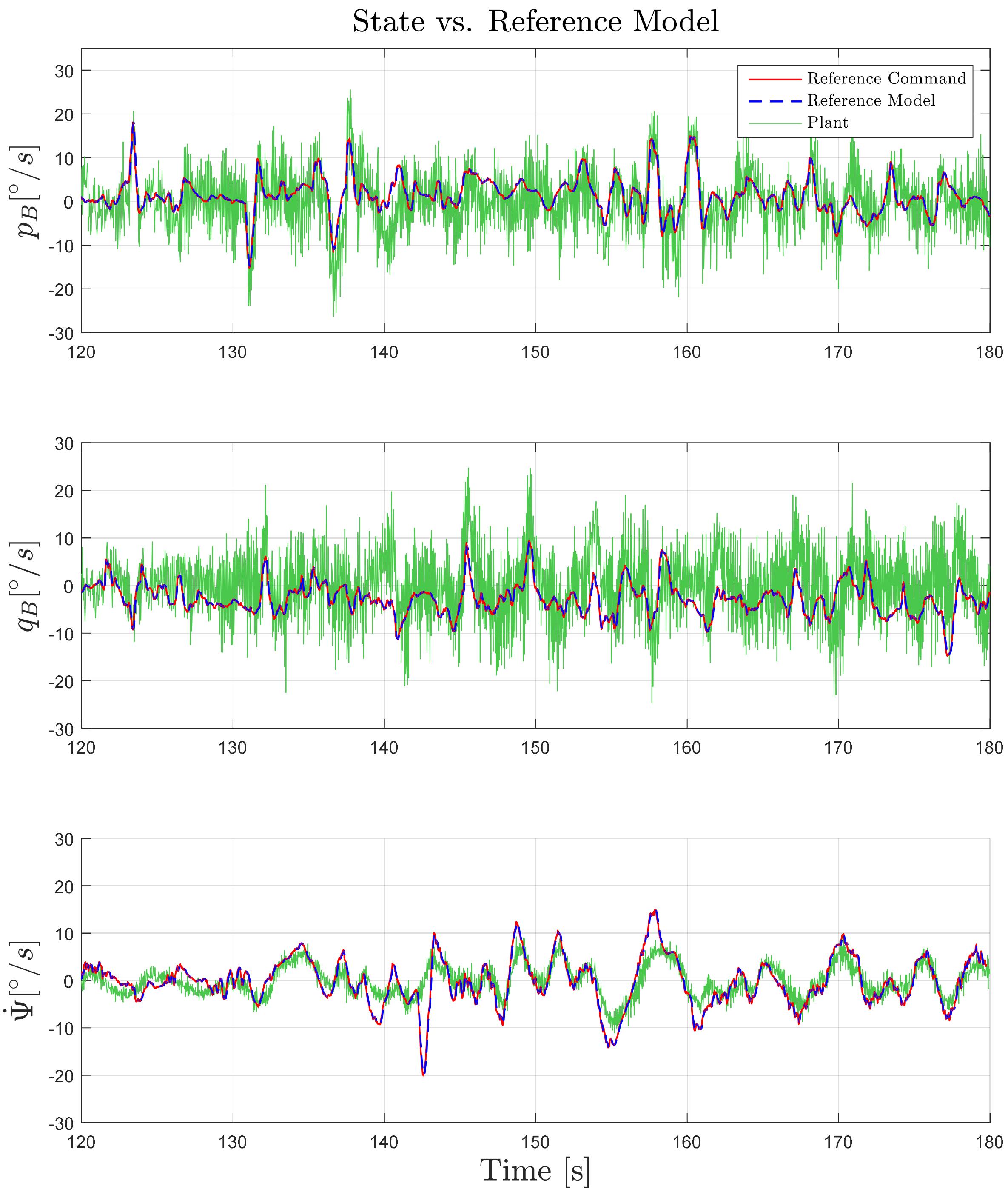


Figure 6-10: Tracking performance of the adaptive controller in the hexacopter experiment

Figure 6-10 shows the reference command, the state of the reference model as well as the evolution of the state trajectories over the course of the experiment. The pilot excites the rotational rates throughout the experiment by issuing attitude commands. The output of the attitude control loop then generates rate commands. It can be seen that the noise level of the roll and pitch rate is significantly higher than the one of the yaw rate. This can mainly be attributed to the proportional feedback gains of the baseline tracking controller in the rotational loop. In particular, the gains for roll and pitch rate are three times larger than the feedback gain of the yaw rate channel. Despite the measurement noise, the system is seen to track the reference model.

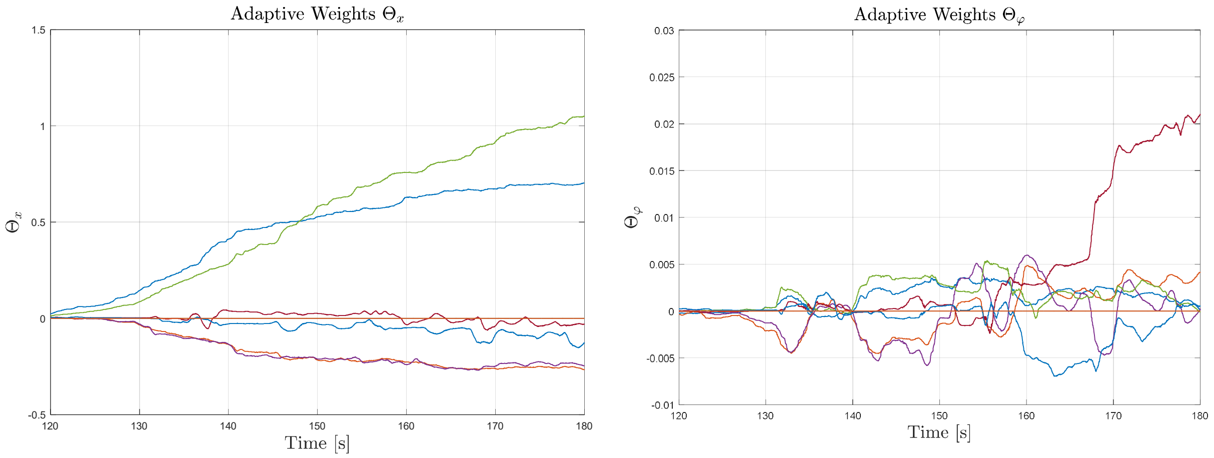


Figure 6-11: Evolution of the adaptive weights in the hexacopter experiment

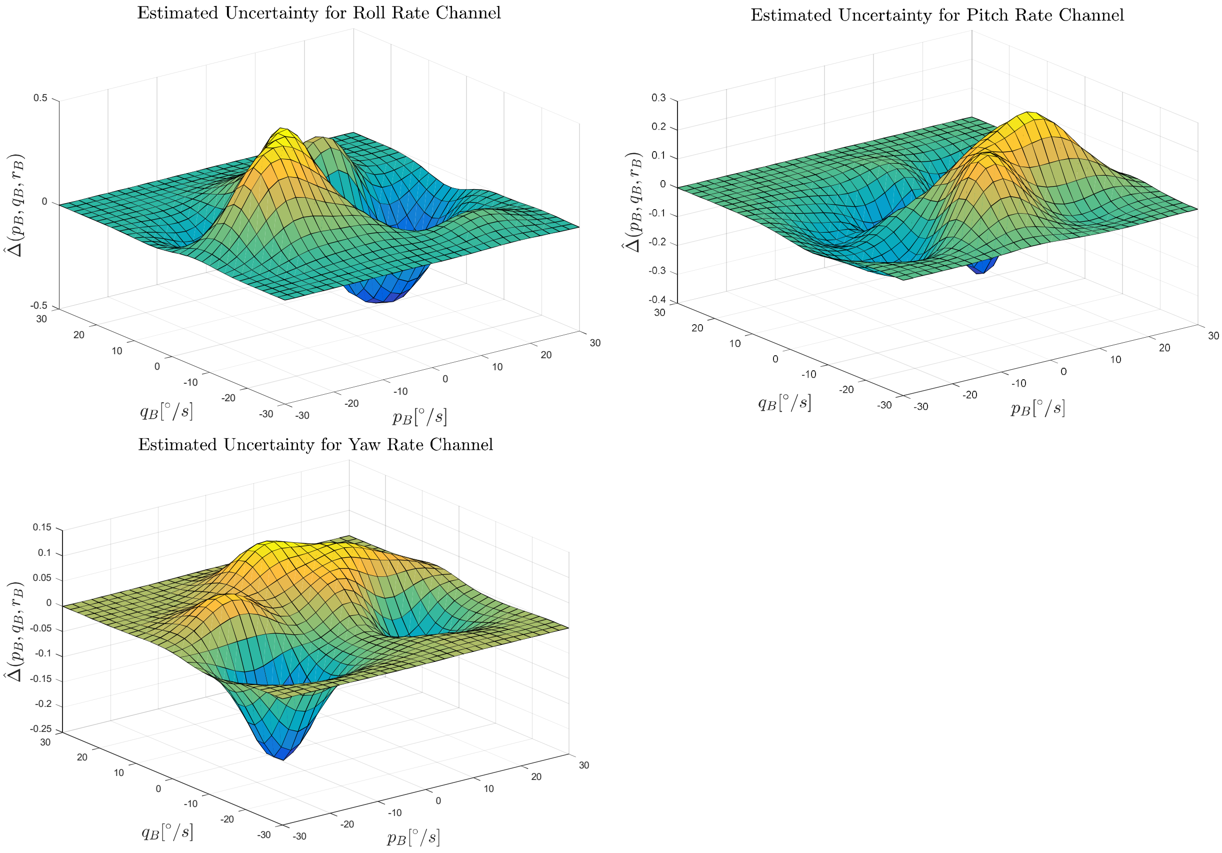


Figure 6-12: Uncertainty estimation result in the hexacopter experiment for a vanishing yaw rate

Figure 6-11 depicts the evolution of the adaptive parameters. After take-off , the weights start to evolve quickly in order to cancel the modelling uncertainty. It can be seen that some weights converge to certain numerical values within 20 seconds of the experiment. This particularly applies to the proportional gains $\boldsymbol{\Theta}_x$. This indicates that the adaptive controller is approximating the modelling uncertainty. In contrast, the nonlinear gains $\boldsymbol{\Theta}_\varphi$ fail to converge, but are continuously excited by the system rates. However, the adaptive parameters stay bounded, which can be attributed to the σ −Modification in the update laws.

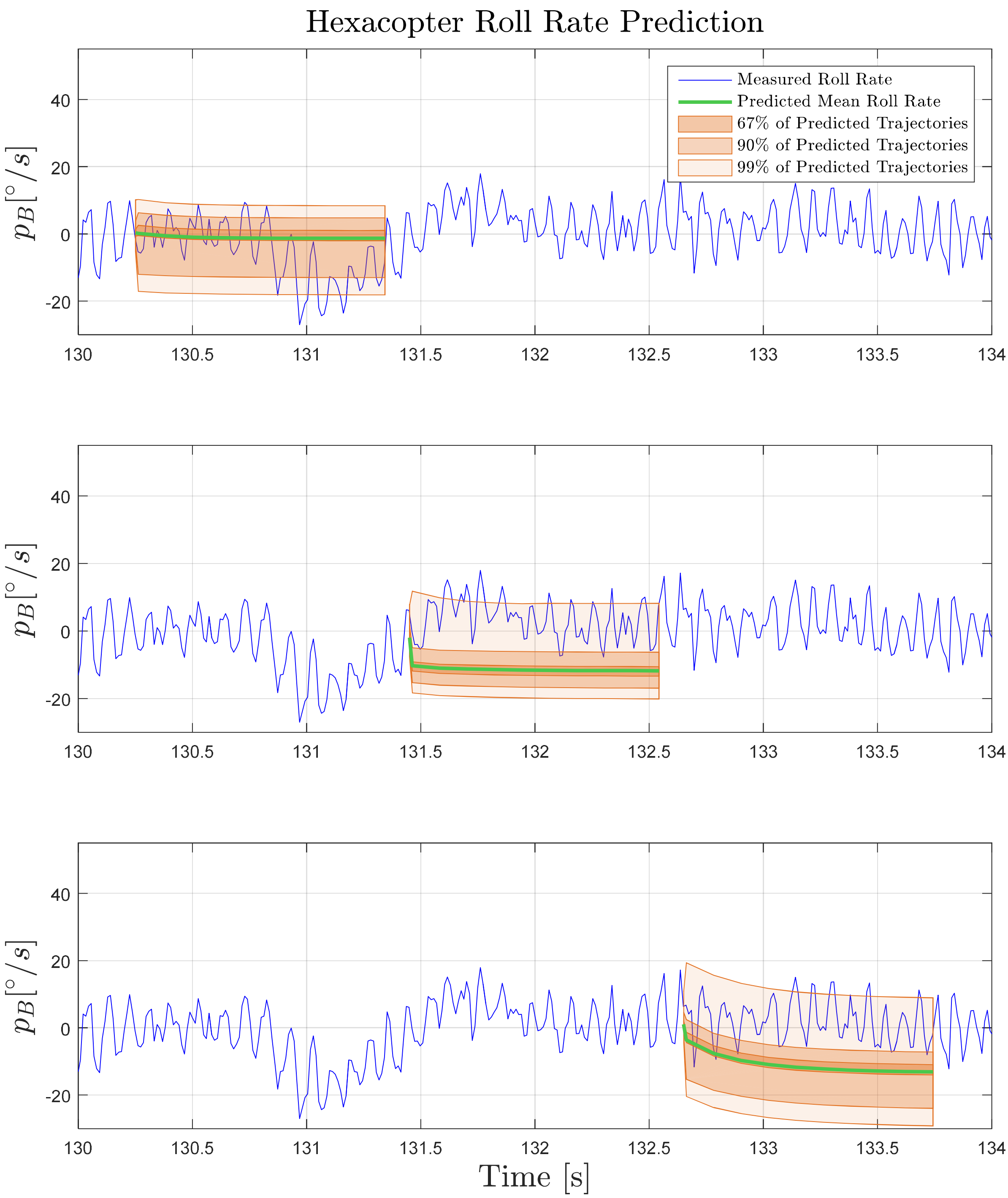


Figure 6-13: Selected predictions for the roll rate in the hexacopter experiment with unknown reference command evolution

Figure 6-12 shows the estimate of the uncertainty for each rate channel using the GP regression algorithm of section 4.4.3. The actual GP is dependent on the three rotational rates. In order to depict the estimate, here the yaw rate is kept constant at $r_B = 0°/s$. The resulting estimate is only dependent on the pitch and roll rate. It can be seen that both, the estimated roll and pitch rate uncertainties assume an axissymmetrical shape. Since a modelling uncertainty in the rate dynamics corresponds to a moment, the observed symmetry could point to a counteracting aerodynamical moment.

Exemplary for the complete experiment, Figure 6-13 shows the predicted evolution of the pitch rate at three points in time. In particular, it depicts the predicted mean trajectory as well as the area in which 67%, 90% and 99% of all state trajectories are predicted to reside in at each time instant. This example is intended to highlight the necessity of the assumption that the evolution of the exogenous reference model is known.

It can be seen that the three predicted channels are asymmetrical to the predicted mean trajectory. This indicates that the predicted pdf, which is formed by the GMM, assumes a non-Gaussian shape. For example, the predicted roll rate distribution during the second prediction is long tailed towards positive roll rates. Secondly, it can be seen that the mean trajectory converges to specific numerical values within the prediction horizon. This originates from assuming a constant attitude command. The latter drives a linear reference model for the attitude dynamics, which in turn generates a rate command. As the attitude reference model has a relative degree of one, it is comparable to the step response of a first order low-pass filter. The monitoring algorithm predicts the rate dynamics to follow this behavior.

As a direct consequence of this approach, the monitoring algorithm is seen to produce prediction errors within the prediction horizon. In particular, the mean of the predicted state trajectories does not match the mean of the state evolution. Still, the states are seen to recide within channels for most of the prediction horizon. This can be attributed to the GP as well as the considered measurement noise, which both increase the standard deviation of the mixture elements.

Figure 6-14 shows selected predictions over the complete experiment. In particular, the monitor is called in every time step. For the sake of depiction, the predictions are only shown every second. As a result, they do not overlap. In the case of the roll and pitch rate, it can be seen that even though the predicted mean does not match the mean of the state evolution, the states still often reside within the predicted channels. However, sometimes the single prediction do not match the state evolution at all. This again can be attributed to the unknown evolution of the exogenous reference command. In particular, if the monitor is called one time step before a significant change in the attitude command, the prediction produces large deviations from the state evolution. Calling the monitor again, after the change occurred adjusts the predictions as the monitor is able to incorporate the added information.

In contrast, it can be seen, that even though the predicted mean for the yaw rate does not match the state evolution, the yaw rate still resides within the three channels nearly all of the time. This can be attributed to the fact, that the belief on the uncertainty affecting the yaw dynamics, which is formed by GP regression, is highly uncertain itself. This indicates, that not enough data points were recorded. As a result, it increases the standard deviation of the mixture elements and also spreads their respective means, thus also increasing the channel size.

In summary, the experiment showed the necessity behind the assumption of a known reference command evolution over the prediction horizon. The latter can originate from e.g. a pilot model or a trajectoriy generator. Still, assuming a constant reference command is possible. In this case, the prediction horizon as well as the frequency in which the monitor is called need to be carefully selected in order to achieve usable results.

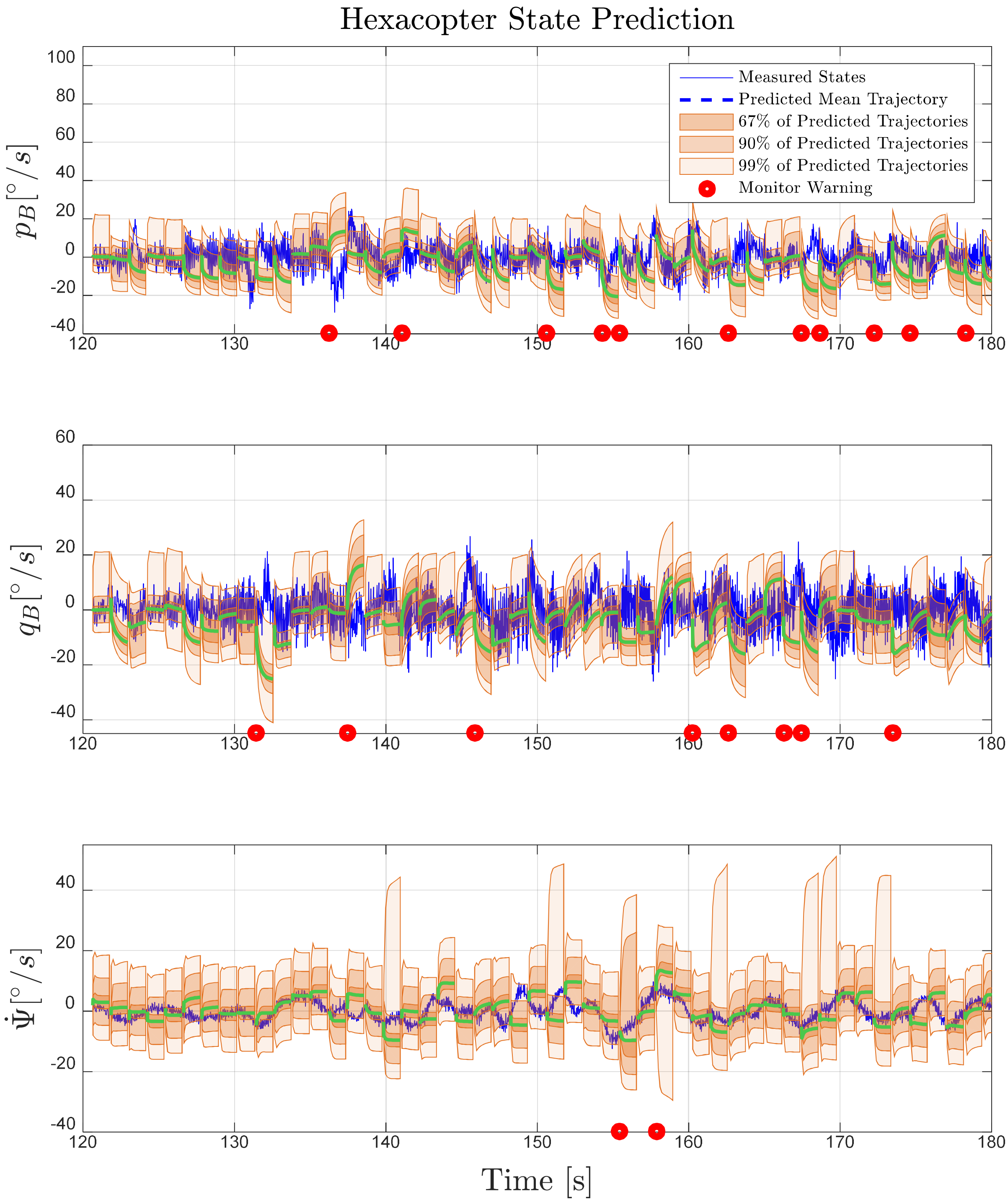


Figure 6-14: Predicted roll, pitch and yaw rate in the hexacopter experiment with unknown reference command evolution

The experiment is repeated in an offline replay. For the replay, the evolution of the exogenous reference command is assumed to be known. Figure 6-15 shows the selected predictions of Figure 6-13 for the known reference command. It can be seen that the predicted mean now qualitatively corresponds to the mean evolution of the roll rate. Furthermore, the predicted channels engulf the measured roll rate. This indicates that the monitor is able to predict the system states of the hexacopter well if the reference command is known.

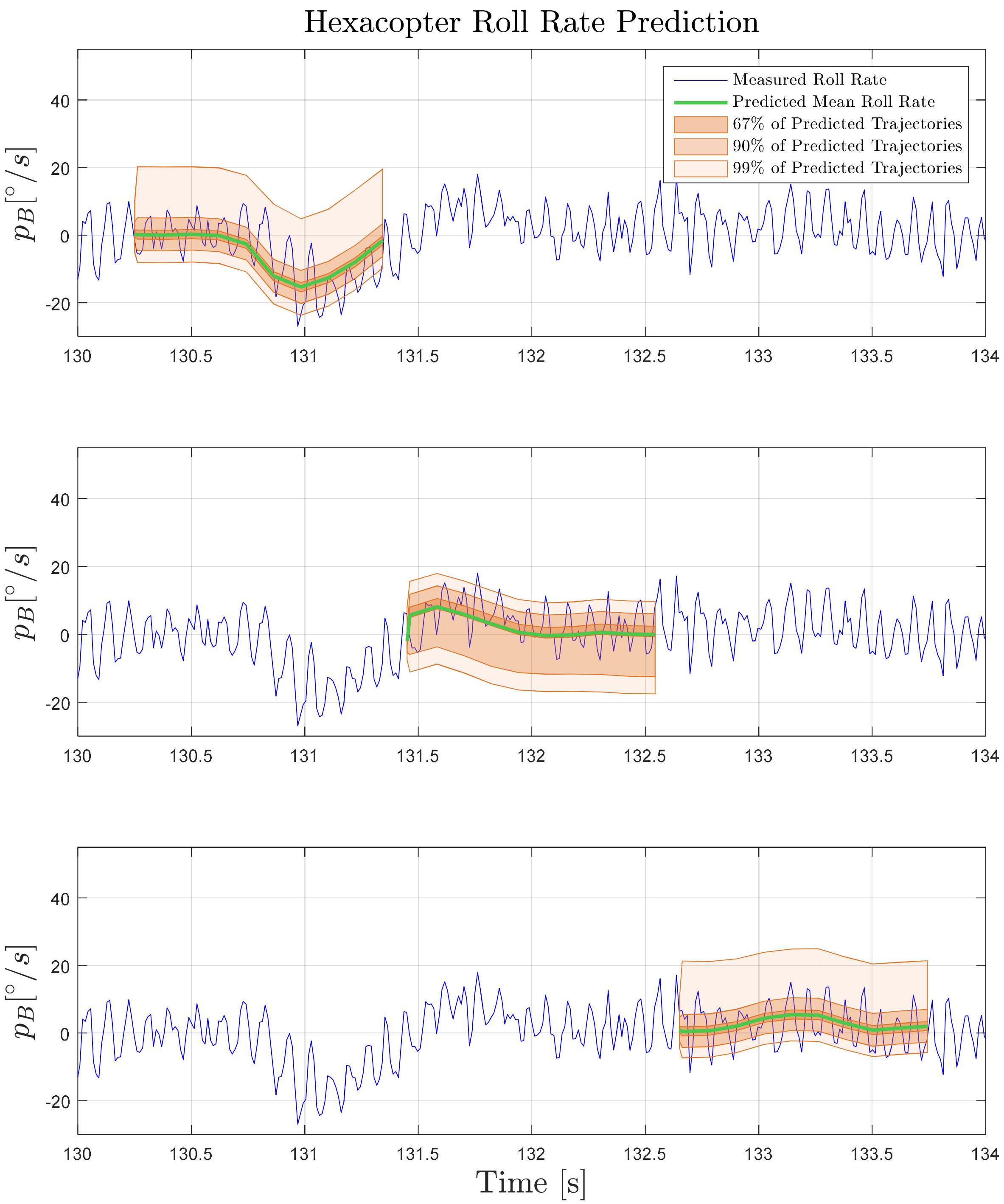


Figure 6-15: Selected predictions for the roll rate in the hexacopter experiment with known reference command evolution

Figure 6-16 shows the replay of the complete experiment. In contrast to Figure 6-14, the predicted mean is seen to qualitatively correspond to the mean evolution of the states. Furthermore, the channels engulf the system states for nearly the complete experiment. This suggests that the monitor is able to predict the system states and act as an indicator for off-nominal behavior. Note that the uncertainty in the GP of the yaw rate is independent on the knowledge of the reference command. Hence, the resulting GMM form similarly large channels as observed in Figure 6-14.

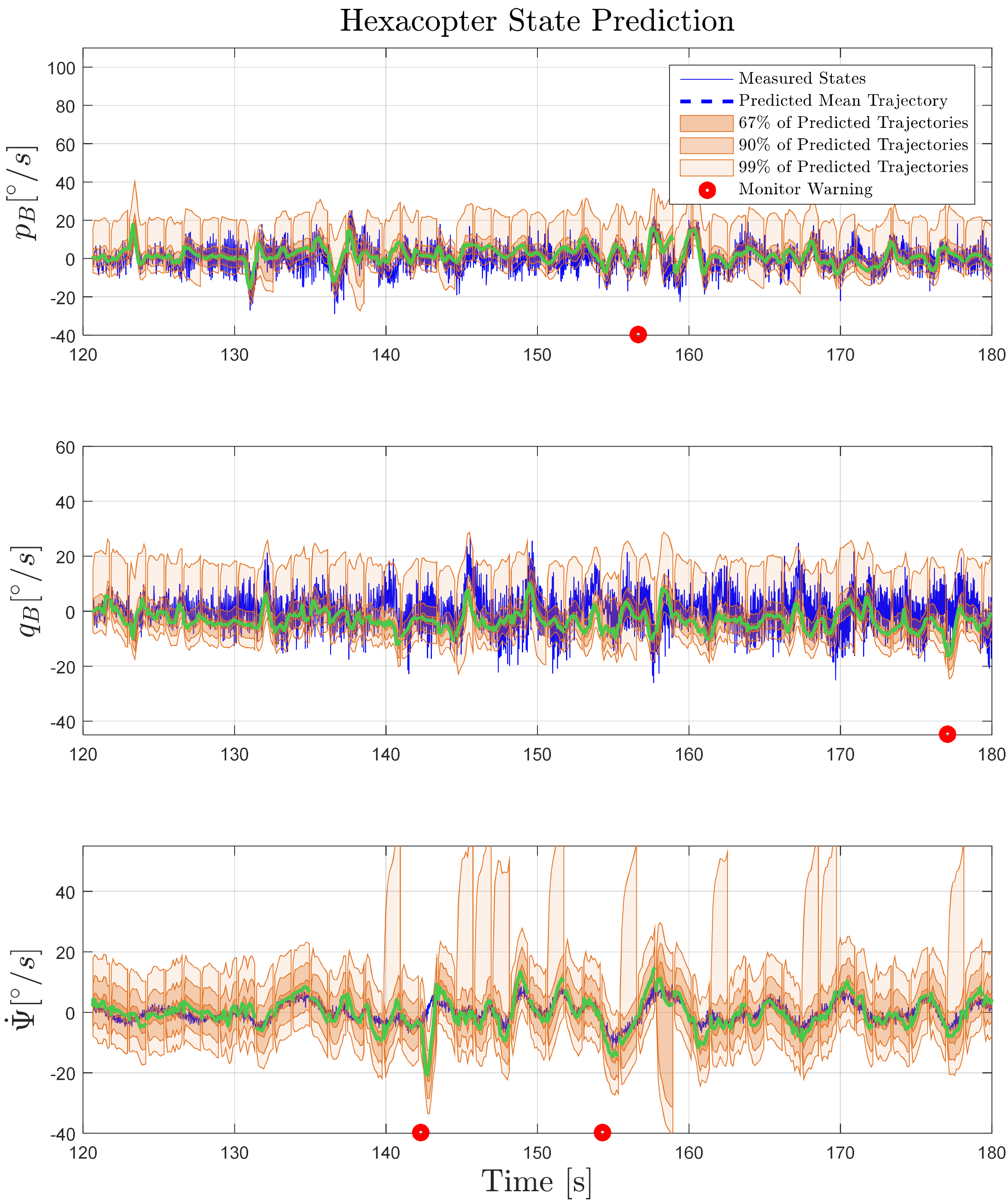


Figure 6-16: Predicted roll, pitch and yaw rate in the hexacopter experiment with known reference command evolution

Figure 6-17 depicts the predicted probability of angular rate requirement violation according to equation (4-102). It can be seen that the forecast predicts some probable requirement violations. In particular, the monitor depicts an upper requirement violation for the roll rate after approximately 124 seconds. Furthermore, the monitor detects a lower requirement violation for the yaw rate after approximately 143 seconds. As the prediction horizon is set to 1 second, the imminent requirement violation is predicted 1 second in advance. After the predicted distributions return within the state limits set by the requirements, the probability of a requirement violation reduces to zero.

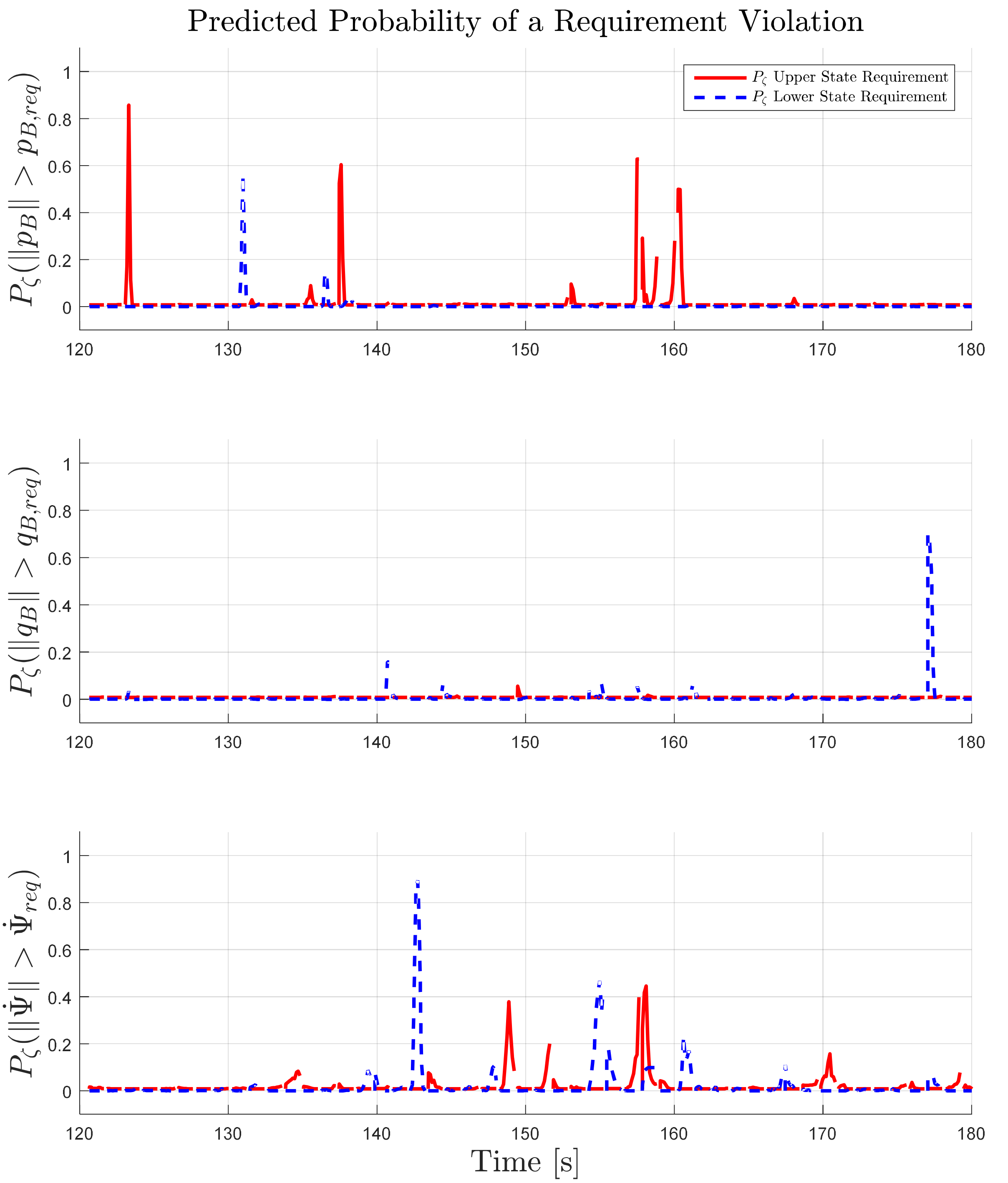


Figure 6-17: Probability of a angular rate requirement violation in the hexacopter experiment

Finally, Figure 6-18 shows the predicted probability of a requirement violation P_{mod}, which is modified by the OVL according to (4-109). In particular, the time windows, which are influenced by the OVL the most are highlighted. The resulting curves are seen to be more conservative. As the monitoring algorithm is called frequently, the OVL is reset to zero each time. This results in the spike pattern visible in Figure 6-18. Note, that in reality the monitor is called every time step, thus issuing significantly more results. With the parameter set of Table 6-4, the monitor now issues warnings for all three angular rates.

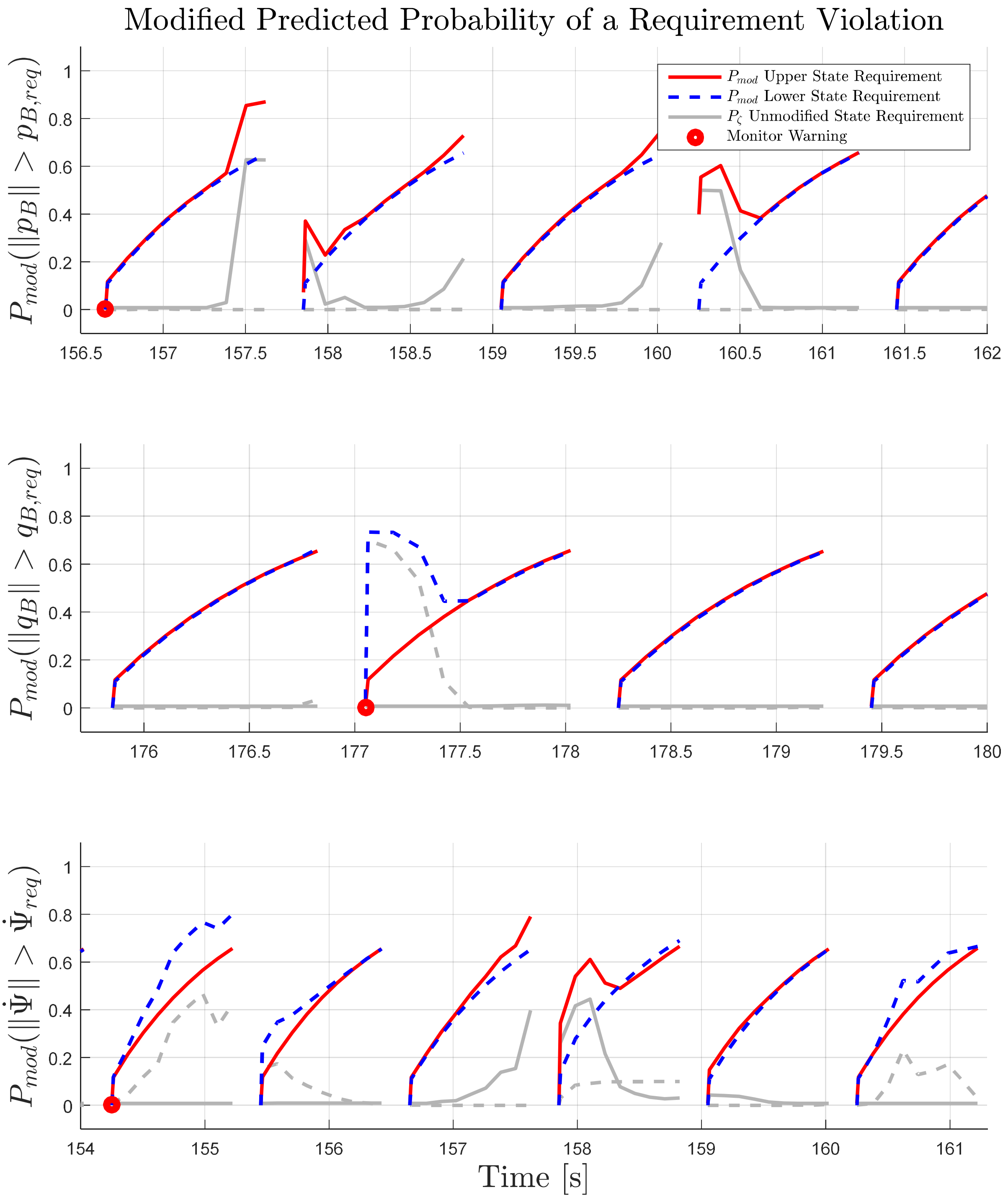


Figure 6-18: OVL modified predicted probability of an angular rate requirement violation in the hexacopter example

This example highlighted the challenges occuring if the exogenous reference command is unknown. In summary, the monitoring algorithm is seen to predict the evolution of the angular rates well if the reference command is known. In the opposite case, the monitor can still be applied, but should be called at a high frequency with only a low prediction horizon.

6.4 Conclusion

Chapter 6 applies the proposed monitoring approach of section 4 to two adaptively controlled multirotor systems. In particular, section 6.1 derives the rate dynamics of multirotor systems. Section 6.2 applies the monitoring algorithm to a quadcopter with a direct MRAC based rate controller. Section 6.3 applies the proposed monitoring approach to the rotational dynamics of a hexacopter with direct MRAC augmenting a NDI based baseline controller.

In both experiments the monitor was shown to be able to predict the evolution of the system states and thus corroborate the simulation results of section 5. Some aspects and recommendations can be discerned from the experiments. First, the frequency of the monitor calls plays a vital role. Increasing the latter directly raises the hardware requirements. On the other hand, a high frequency is important, in particular if the evolution of the exogenous reference command is unknown. If the monitor is called frequently it can quickly incorporate new information such as rapid reference command changes. The quadcopter example showed that selecting a higher frequency is reasonable even if the reference command was known. In particular, calling the monitor only once within the time frame covered by the prediction horizon, may lead to situations were only a short time remains to react to the imminent requirement violation. Hence, the frequency of monitor calls is recommend to be high enough that the prediction horizons of multiple predictions overlap sufficiently. This even more applies if the evolution of the reference command is unknown.

The prediction horizon behaves similar to the prediction frequency. Increasing the prediction horizon also raises the requirements on the computational hardware. Still, increasing the former also results in longer reaction times, which in general is desirable. However, prediction errors propagate over the horizon thus making the forecast less reliable the longer it lasts. To capture this, the OVL was introduces. The latter modifies the probability of a requirement violation relative to the expected propagation errors. For long prediction horizon, the modified probability increases drastically. If the reference command is unknown, the hexacopter example showed that the monitor frequency should be favoured above the length of the prediction horizon. This allows for a quick incorporation of new information such as a change in the reference command. All in all, setting the parameters comes down to optimizing the monitor with regards to the three goals hardware requirements, prediction horizon and prediction frequency. As a recommendation, once a minimum prediction horizon is defined, the monitoring frequency should be favoured above the horizon.

The experiments suggest three practical applications of the proposed monitoring algorithm. First, the monitor can be applied to the onboard hardware to achieve an assessment on probable requirement violations on the system itself and to be able to quickly react. Secondly, the monitor can be applied to a ground station, which usually favours stronger hardware, but doesn’t allow for immediate reaction to potential violations. Lastely, the monitor can be used to replay experiments and simulations in order to analyse them or discern the impact of changes on the closed-loop system.

All in all the proposed approach is shown to be able to predict the system states and act as a monitor for imminent requirement violation in adaptively controlled systems. Hence, it presents an alternative way of moving from an offline analysis to verification during runtime. Simultaneously, the mentioned disadvantages and short comings offer room for further development.

7 Conclusion and Outlook

7.1 Conclusion

The aim of this thesis was to introduce a concept towards runtime verification of adaptive controllers. Therefore, this thesis introduced an online monitoring concept, which aims at detecting imminent state requirement violation and undesired behavior of an adaptively controlled closed loop system. The proposed monitor predicts the distribution of the state and adaptive parameter trajectories. In turn, the predictions are analyzed at each time step by evaluating the predicted distributions with respect to a-priori formulated state requirements. If the predicted state distributions violate a requirement, the monitoring concept issues a warning.

In order to predict the distributions of the state trajectories this thesis uses GMMs. Thereby, the initial state distribution is approximated by the GMM. The resulting mixtures are then propagated through the plant model. The latter consists of two parts. First, often a significant portion of the possibly nonlinear system dynamics are known. During a prediction step, the nonlinear system equations are linearized at each mixture element. The mixture elements are then propagated through the surrogate system. Secondly, most systems are subject to modelling uncertainties. This includes amongst others, parameter uncertainties or neglected dynamics. In order to incorporate the modelling uncertainties into the predictions, the former are quantified by using GP regression. The latter uses specifically recorded data points in order to form a belief on the modelling uncertainty. The mixture elements are then propagated through the resulting GP.

At each prediction step, the system states are represented by a GMM. In order to assess, whether an imminent requirement violation occurs, the cdf of the GMM is compared against the defined state requirements. If the probability of a requirement violation is higher than a selected threshold, warning is issued.

The forecast of the system states inevitably introduces faults, which could originate from numerical errors or linearization inaccuracies. This thesis uses the OVL in order to assess the estimated prediction errors made throughout the recursive application of the forecast. Thereby, the probability of an imminent requirement violation is modified by the OVL. The result is more conservative, i.e. the monitor issues more warnings than before. However, in the view of the author, an increased false positive rate is acceptable as long as false negatives are highly unlikely. This ensures a safe operation of the system.

The monitoring concept was applied to two examples in numerical simulation. First, the monitor was applied to the roll dynamics of a fighter aircraft subject to wingrock dynamics. The controller was augmented by direct MRAC. Secondly, the monitor was applied to short-period dynamics subject to nonlinear model uncertainties. In the second example, a PMRAC augmented baseline controller was used. In both examples, the GP regression was able to accurately estimate the modelling uncertainty within the region governed by the recorded data. As a result, the monitor showed good prediction capabilities. The GoC was introduced as an indicator to assess the quality of the forecast. The GoC indicates whether or not the predicted estimate of the state distribution under- or overestimates the real state evolution. By introducing slack variables into the GoC, prediction errors and numerical errors could be incorporated into the indicator. Using the GoC, the GMM based forecast was compared to a forecast governed by a normal distribution only. The results showed, that using a simple normal distribution instead of a GMM introduces significant errors into the predicted state distributions, thus leading to wrong monitor warnings.

The results of the numerical simulation were corroborated in experiments by applying the monitoring concept to the rate dynamics of both, a quadcopter and a hexacopter system. The quadcopter is controlled by a full direct MRAC. In contrast, the baseline controller for the hexacopter is based on feedback linearization with a relative degree of one. A key assumption of the monitor is that the evolution of the exogenous reference command is known. This information can e.g. be the output of a pilot model or a trajectory generation. In the unknown case, the reference command was assumed to be constant over the prediction horizon. Naturally, the predictions are significantly wrong if the reference command indeed changes. One option to counter this effect is to increase the frequency of monitor calls. This allows for a quick introduction of new information in order to counter this effect. In the cases it was known, the proposed monitoring approach was able to predict the state evolution accurately. Furthermore, the monitor was able to accurately warn from imminent state requirement violations. Consequently, in the authors opinion the proposed monitoring approach is particularly useful for applications with a high grade of automation.

The experiments suggest three practical applications of the proposed monitoring algorithm. First, the monitor can be applied to the onboard hardware to achieve an assessment on probable requirement violations on the system itself and to be able to quickly react. Secondly, the monitor can be applied to a ground station, which usually favours stronger hardware, but doesn't allow for an immediate reaction to potential violations. Lastely, the monitor can be used to replay experiments and simulations in order to analyse them or discern the impact of changes on the closed-loop system.

An emerging consensus between various authors classifies runtime verification as a key component in the V&V process for adaptive control algorithms. The presented monitoring algorithm constitutes a possible concept of how to perform run-time monitoring for adaptive controllers. Still, the proposed approach assumes a prototype state and needs to be further refined for future application.

7.2 Summary of Contributions

This section summarizes the contributions made within this thesis.

Analytic long term prediction for monitoring of adaptively controlled systems

A central contribution of this thesis was the proposal of a stochastic monitoring system for adaptively controlled systems. The monitor relies on analytic long term forecasts of the system states in order to assess potential undesired behavior in the near future. Therefore, the monitor approximates the initial state distribution using a GMM. The GMM is propagated through the known linearized surrogate dynamics at each mixture element as well as the GP model. Recursive application of the prediction forms the predicted state evolution over the prediction horizon. To the best of the authors' knowledge, this approach has neither been employed for nonlinear flight control applications nor adaptive controllers.

To assess the quality of the forecast if the ground truth is known, the Grade of Circumvallation (GoC) is introduced by the author in section 5. The GoC indicates whether or not the predicted estimate of the state distribution under- or overestimates the real state evolution. The idea is that an overestimated distribution circumvallates the real distribution and thus yields more conservative results if absolute state requirements are used. By introducing slack variables into the GoC prediction errors and numerical errors can be incorporated into the indicator.

Combined state and control prediction for system monitoring

A central contribution of the proposed approach is the combined evaluation of controller and plant states. Existing monitoring approaches predominantly aim at assessing the learning capabilities of the adaptive element only (see amongst others [129], [133], [134], [132], [188], [222], [68], [186], [186], [82], [188]). The presented monitoring technique significantly differs from the mentioned techniques in its strategy. Thereby, the approach models the dependencies between the plant and the controller. The combined prediction of plant and controller state trajectories enables the assessment of the adaptive control impact on the system states in the future. As a consequence, the monitor is seen to also consider transients in the controller and plant state.

Contribution to a possible future certification process of adaptive flight control algorithms

The proposed monitoring algorithm demonstrates a possibility of closing the gaps in the certification process of adaptive control algorithms. An emerging consensus between various authors classifies runtime verification as a key component in the V&V process for adaptive control algorithms. The presented monitoring algorithm constitutes a possible approach of how to perform run-time monitoring for adaptive controllers. In particular, the proposed approach is able to directly relate the performance of the adaptive control algorithm to a-priory formulated state requirements and thus test for their adherence online. Still, the proposed approach assumes a prototype state and needs to be further refined for future use.

Introduction of a Confidence Measure to the prediction horizon

A further contribution of this thesis involves the introduction of a confidence measure in order to express the certainty of the algorithm in its own prediction. In section 4.5.2 the *Overlapping Coefficient* (OVL; [99], see Appendix B.9) was introduced as a way to estimate prediction errors made throughout the recursive application of the forecast step.

Novel use of Gaussian Process regression in adaptive flight control

A further contribution of the proposed technique is the use of GP regression to construct a model for long term prediction in adaptive flight control applications. GP regression has been applied to a huge variety of problems in order to model an uncertain system based on gathered data or even to use it as an augmenting controller ([38], [37]). Instead, the proposed monitoring approach uses GP regression in order to form a belief on the modelling uncertainty for long-term prediction. A similar approach has been utilized in various applications such as predictions in the field of automotive dynamics (see e.g. [85]). To the best of the authors' knowledge, an online generated GP model for multistep ahead prediction has neither been used in flight control nor adaptive control approaches yet.

7.3 Future Work

Several options for future work can be derived from this thesis. They can be loosely classified into application specific work and advancements of the monitor.

7.3.1 Application specific future work

Communicating Results

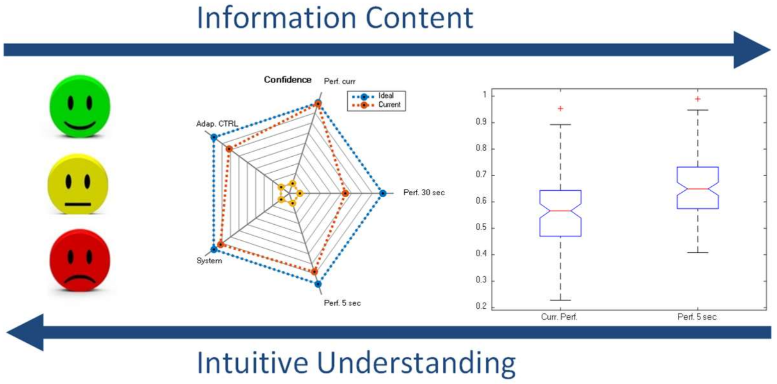


Figure 7-1: Conflicting goals regarding intuitive understanding and information contant

The proposed monitoring output is defined by a probability of an imminent requirement violation and a binary indicator whether or not one appears within the prediction horizon. Future research has to solve the question how and to which extent information is communicated to human operators. In particular, the two intrinsic conflicting goals information contant and intuitive understanding arise within this communication. Figure 7-1 shows three communication examples as well as their classification along the two conflicting goals. One possible way is to just communicate a binary variable if a requirement violation occurs. While this is intuitively understandable and doesn't require expert knowledge, a lot of information is lost and more detailed information would be needed to gauge the seriousness of the situation. A second option is using a radar chart. The information content increases. However, more time is required to grasp the situation. This is even truer when looking at the third extreme in the form of box plots. Here, the operator requires expert knowledge how to quickly assess the situation, but on the other hand this option carries the highest information content.

Requirement Formulation

In the presented approaches, the requirements are formulated as constant values. Future approaches can incorporate more advanced requirement formulations as can be found in e.g. [17, p. 163]. Furthermore, the requirements could be formulated in a relative fashion rather than as absolute values.

Confidence based reference command shaping

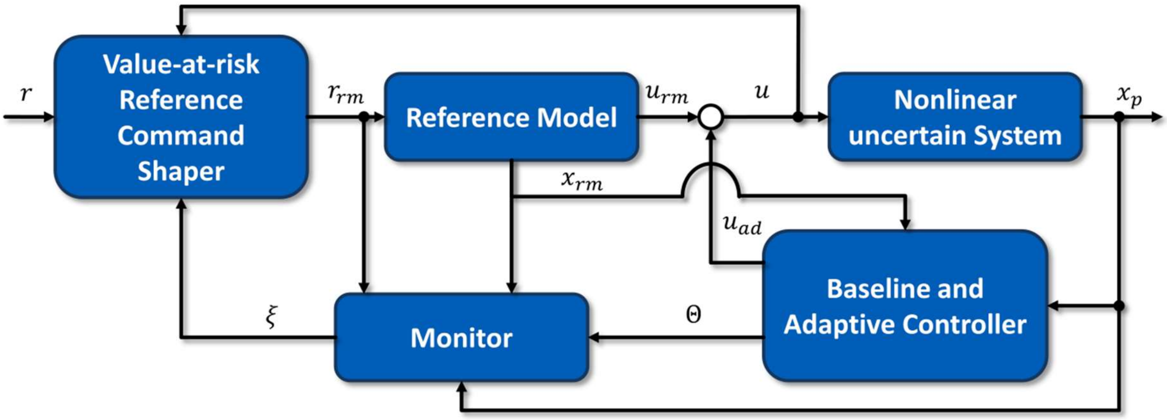

r
Value-at-risk Reference Command Shaper
r_{rm}
Reference Model
u_{rm}
u
Nonlinear uncertain System
x_p
x_{rm}
u_{ad}
Baseline and Adaptive Controller
ξ
Monitor
Θ

Figure 7-2: Value-at-risk based reference command shaping

Reference command shapers use knowledge about the reference model and plant to adjust the exogenous reference commands in order for the system to exhibit maximum performance under defined constraints. Existing reference command shapers for adaptively controlled systems (see i.a. [145], [147], [149], [40], [39]) assume that the uncertainty is cancelled pointwise in time and use the mean of measured states and other signals to adjust the exogenous reference command. With the monitoring approach detailed in this thesis, the reference shaper could use the predicted state distribution instead of only the mean. As a result the constraints are no longer deterministic but formulated in a stochastic way. In this context, optimization can be performed based on a “value-at-risk” perspective. Also, using the monitor would remove the assumption of previous approaches that the adaptive controller already estimates the modelling uncertainty pointwise in time, since predicting the evolution of the weights is an intrinsic property of the proposed monitor. Figure 7-2 shows an exemplary approach of how to integrate the monitor into the architectures of i.e. [145], [147], [149]. This approach could also be a candidate answer to the question how the system should react to an imminent requirement violation, as the shaper would change the reference command in such a way that the plant abides the stochastic state constraints.

7.3.2 Possible future advancement of the monitoring approach

Introducing pilot models for the reference command forecast

The application of the monitoring algorithm to the hexacopter system in section 6.3 showed that the prediction is heavily dependent on the evolution of the exogenous reference command. If the latter does not originate from a trajectory generator, it can be calculated from pilot models. Hence, a natural extension to the proposed monitoring approach is the introduction of pilot models.

Update of the mixture weights

The presented monitoring approach leverages a GMM forecast. The GMM is initialized at the beginning of the predictions, the number of mixtures as well as their weight within the GMM is kept constant over the prediction horizon. One future step could be to also adjust the weights of the GMM at each prediction step (see e.g. [61]). This could lead to an increase of the approximation capability of the forecast.

Evolutionary GMM

The number of mixture models as well as their initial standard deviation is fixed. Evolutionary GMM ask the question of how mixture models can be split and merged in order to make the pdf approximation better (see e.g. [46]). Even though the computational requirements increase, the approximation quality might increase, thus allowing for longer prediction horizons with higher confidence.

Parallelization of the computation

The mixture elements in the proposed monitoring algorithms are independent of each other after they are initialized. As a direct result, their evolution across the prediction horizon can be computed parllely. A natural future step could be to apply the monitoring algorithm to hardware with massive parallelization capabilities, such as GPUs. As a result, either the number of mixture elements could be increased in order to increase the approximation accuracy. On the other hand the termination time of an existing monitor could be decreased. This results in higher monitoring frequencies or longer prediction horizons.

Different machine learning concepts for uncertainty quantification

The proposed monitoring approach leverages Gaussian Process regression in order to form a belief on the uncertainty. The fields of machine learning and computational hardware are quickly evolving. A natural future step is to investigate further machine learning techniques, such as deep neural networks [78], in order to improve the belief of the uncertainty quantification.

References

[1] **Achtelik, M., Doth, K.-M., Gurdan, D., and Stumpf, J.**, “Design of a Multi Rotor MAV with regard to Efficiency, Dynamics and Redundancy,” in *AIAA Guidance, Navigation, and Control Conference*, 2012.

[2] **Akaike, H.**, “A new look at the statistical model identification,” *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, 1974.

[3] **Anderson, B.**, “Exponential stability of linear equations arising in adaptive identification,” *IEEE Transactions on Automatic Control*, vol. 22, no. 1, pp. 83–88, 1977.

[4] **Anderson, B. D. O. and Moore, J. B.**, *Optimal filtering,* 1979th ed. Mineola NY: Dover Publ, 2005.

[5] **Andrews, R., Diederich, J., and Tickle, A. B.**, “Survey and critique of techniques for extracting rules from trained artificial neural networks,” *Knowledge-Based Systems*, vol. 8, no. 6, pp. 373–389, 1995.

[6] **Andrews, R. and Geva, S.**, “RULEX & CEBP Networks As the Basis for a Rule Refinement System,” in *Hybrid Problems, Hybrid Solutions, John Hallam (Ed)*: IOS Press, 1995, pp. 1–12.

[7] Ascending Technologies, *AscTec Hummingbird.* Available: http://www.asctec.de/en/uav-uas-drones-rpas-roav/asctec-hummingbird/. Accessed on: Oct. 06 2018.

[8] **Aström, K. J. and Wittenmark, B.**, *Adaptive control,* 2nd ed. Mineola, N.Y: Dover Publications, 2008.

[9] **Azzalini, A. and Bowman, A. W.**, “A Look at Some Data on the Old Faithful Geyser,” *Applied Statistics*, vol. 39, no. 3, p. 357, 1990.

[10] **Barber, D.**, *Bayesian reasoning and machine learning*, Cambridge, New York: Cambridge University Press, 2012.

[11] **Bayes, T. and Price, R.**, “An Essay towards Solving a Problem in the Doctrine of Chances.: By the Late Rev. Mr. Bayes, F. R. S. Communicated by Mr. Price, in a Letter to John Canton, A. M. F. R. S,” *Philosophical Transactions of the Royal Society of London*, vol. 53, no. 0, pp. 370–418, 1763.

[12] **Bemporad, A., Casavola, A., and Mosca, E.**, “Nonlinear control of constrained linear systems via predictive reference management,” *IEEE Trans. Automat. Contr.*, vol. 42, no. 3, pp. 340–349, 1997.

[13] **Bemporad, A. and Morari, M.**, “Robust model predictive control: A survey,” in *Lecture Notes in Control and Information Sciences, Robustness in identification and control*, London: Springer London, 1999, pp. 207–226.

[14] **Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N.**, “The explicit linear quadratic regulator for constrained systems,” *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[15] **Ben-Israel, A. and** Greville, T. N. E, *Generalized inverses: Theory and applications,* 2nd ed. New York: Springer, 2003.

[16] **Bierlaire, M., Toint, P.L., and Tuyttens, D.**, “On iterative algorithms for linear least squares problems with bound constraints,” *Linear Algebra and its Applications*, vol. 143, pp. 111–143, 1991.

[17] **Bierling, T.**, “Comparative Analysis of Adaptive Control Techniques for Improved Robust Performance,” Dissertation, Lehrstuhl für Flugsystemdynamik, Technische Universität München, München, 2014.

[18] **Bierling, T., Höcht, L., Holzapfel, F., Maier, R., and Wildschek, A.**, “Comparative Analysis of MRAC Architectures in a Unified Framework,” in *AIAA Guidance, Navigation, and Control Conference*, 2010.

[19] **Bierling, T., Höcht, L., Merkl, C., Holzapfel, F., and Maier, R.**, “Similarities of Hedging and L1 Adaptive Control,” in *Advances in Aerospace Guidance, Navigation and Control*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 535–554.

[20] **Billingsley, P.**, *Probability and measure,* 3rd ed. New York: Wiley, 1995.

[21] **Bishop, C. M.**, *Pattern recognition and machine learning*, New York: Springer, 2006.

[22] **Boser, B. E., Guyon, I. M., and Vapnik, V. N.**, “A training algorithm for optimal margin classifiers,” in *Fifth annual workshop*, pp. 144–152.

[23] **Boyd, S. and Sastry, S. S.**, “Necessary and sufficient conditions for parameter convergence in adaptive control,” *Automatica*, vol. 22, no. 6, pp. 629–639, 1986.

[24] **Brandimarte, P.**, *Handbook in Monte Carlo simulation: Applications in financial engineering, risk management, and economics*. Hoboken, NJ: Wiley, 2014.

[25] **Brockhaus, R., Alles, W., and Luckner, R.**, *Flugregelung*, Dordrecht: Springer, 2011.

[26] **Burnham, K. P.**, “Multimodel Inference: Understanding AIC and BIC in Model Selection,” *Sociological Methods & Research*, vol. 33, no. 2, pp. 261–304, 2004.

[27] **Calise, A. J. and Yucelen, T.**, “Adaptive Loop Transfer Recovery,” *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 3, pp. 807–815, 2012.

[28] **Candela, J. Q., Girard, A., Larsen, J., and Rasmussen, C. E.**, “Propagation of uncertainty in Bayesian kernel models - application to multiple-step ahead forecasting,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP'03)*, Apr. 2003, II-701-4.

[29] **Cao, C.** *et al.*, “Are Phase and Time-Delay Margins Always Adversely Affected by High-Gain?,” in *AIAA Guidance, Navigation, and Control Conference*, 2006.

[30] **Carlton, M. A. and Devore, J. L.**, *Probability with Applications in Engineering, Science, and Technology*, New York, NY, s.l.: Springer New York, 2014.

[31] **Carreira-Perpinan, M. A.**, “Mode-finding for mixtures of Gaussian distributions,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1318–1323, 2000.

[32] **Cavanaugh, J. E.**, “Unifying the derivations for the Akaike and corrected Akaike information criteria,” *Statistics & Probability Letters*, vol. 33, no. 2, pp. 201–208, 1997.

[33] **Chan, Y.-H., Correa, C. D., and Ma, K.-L.**, “The generalized sensitivity scatterplot,” *Transactions on visualization and computer graphics*, vol. 19, no. 10, pp. 1768–1781, 2013.

[34] **Chen, Z.**, “Bayesian filtering: From Kalman filters to particle filters, and beyond,” *Statistics*, vol. 182, no. 1, pp. 1–69, 2003.

[35] **Chowdhary, G.**, “Concurrent learning for convergence in adaptive control without persistency of excitation,” Dissertation, Georgia Institute of Technology, Atlanta, GA, 2010.

[36] **Chowdhary, G. and Jategaonkar, R.**, “Aerodynamic parameter estimation from flight data applying extended and unscented Kalman filter,” *Aerospace Science and Technology*, vol. 14, no. 2, pp. 106–117, http://www.sciencedirect.com/science/article/pii/S1270963809000650, 2010.

[37] **Chowdhary, G., Kingravi, H. A., How, J. P., and Vela, P. A.**, “A Bayesian nonparametric approach to adaptive control using Gaussian Processes,” in *IEEE Conference on Decision and Control*, 2013, pp. 874–879.

[38] **Chowdhary, G., Kingravi, H. A., How, J. P., and Vela, P. A.**, “Bayesian Nonparametric Adaptive Control Using Gaussian Processes,” *Neural Networks and Learning Systems, IEEE Transactions on*, vol. PP, no. 99, 2014.

[39] **Chowdhary, G., Mühlegg, M., How, J. P., and Holzapfel, F.**, “A concurrent learning adaptive-optimal control architecture for nonlinear systems,” in *IEEE Conference on Decision and Control*, 2013, pp. 868–873.

[40] **Chowdhary, G., Mühlegg, M., How, J. P., and Holzapfel, F.**, “Concurrent Learning Adaptive Model Predictive Control,” in *Advances in Aerospace Guidance, Navigation and Control*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 29–47.

[41] **Chowdhary, G., Mühlegg, M., and Johnson, E.**, “Exponential parameter and tracking error convergence guarantees for adaptive controllers without persistency of excitation,” *International Journal of Control*, vol. 87, no. 8, pp. 1583–1603, 2014.

[42] **Chowdhary, G., Yucelen, T., Mühlegg, M., and Johnson, E. N.**, “Concurrent learning adaptive control of linear systems with exponentially convergent bounds,” *Int. J. Adapt. Control Signal Process.*, vol. 27, no. 4, pp. 280–301, 2013.

[43] **Chowdhary, G. V. and Johnson, E. N.**, “Theory and Flight-Test Validation of a Concurrent-Learning Adaptive Controller,” *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 2, pp. 592–607, 2011.

[44] **Chowdhary, G. V., Srinivasan, S., and Johnson, E. N.**, “Frequency Domain Method for Real-Time Detection of Oscillations,” *Journal of Aerospace Computing, Information, and Communication*, vol. 8, no. 2, pp. 42–52, 2011.

[45] **Comaniciu, D. and Ramesh, V.**, “Mean shift and optimal prediction for efficient object tracking,” in *International Conference on Image Processing*, 2000, 70-73 vol.3.

[46] **Covões, T. F., Hruschka, E. R., and Ghosh, J.**, “Evolving Gaussian Mixture Models with Splitting and Merging Mutation Operators,” (eng), *Evolutionary computation*, vol. 24, no. 2, pp. 293–317, 2016.

[47] CS23, “Certification Specifications and Aceeptable Means of Compliance for Normal, Utility, Aerobatic, and Commuter Category Aeroplanes: Amendment 4,” European Aviation Safety Agency, Jul. 2015. [Online] Available: https://www.easa.europa.eu/system/files/dfu/CS-23%20Amendment%204.pdf. Accessed on: Oct. 06 2016.

[47b] CS23, “Certification Specifications for Normal-Category Aeroplanes: Amendment 5,” European Aviation Safety Agency, Mar. 2017. [Online] Available: https://www.easa.europa.eu/sites/default/files/dfu/CS-23%20Amendment%205.pdf. Accessed on: Feb. 28 2019.

[48] **Csató, L. and Opper, M.**, “Sparse On-Line Gaussian Processes,” *Neural Computation*, vol. 14, no. 3, pp. 641–668, 2002.

[49] **Dahm, W.**, “Report on Technology Horizons: Building Future Air Force Capabilities,” United States Air Force, Sep. 2010. [Online] Available: http://www.au.af.mil/au/awc/awcgate/af/tech_horizons_dahm_comments_13sep2010.pdf. Accessed on: Sep. 28 2016.

[50] **Darrah, M., Taylor, B., and Skias, S.**, “Rule Extraction From Dynamic Cell Structure Neural Networks Used in a Safety Critical Application,” in *Florida Artificial Intelligence Research Society Conference*, 2004, pp. 629–634.

[51] **DasGupta, A.**, *Fundamentals of Probability: A First Course*, New York, NY: Springer-Verlag New York, 2010.

[52] **DasGupta, A.**, *Probability for statistics and machine learning: Fundamentals and advanced topics*. New York: Springer, 2011.

[53] **Deisenroth, M. P., Huber, M. F., and Hanebeck, U. D.**, “Analytic moment-based Gaussian process filtering,” in *International Conference on Machine Learning*, 2009, pp. 225–232.

[54] **Deisenroth, M. P., Turner, R. D., Huber, M. F., Hanebeck, U. D., and Rasmussen, C. E.**, “Robust Filtering and Smoothing with Gaussian Processes,” *IEEE Trans. Automat. Contr.*, vol. 57, no. 7, pp. 1865–1871, 2012.

[55] **Dempster, A. P., Laird, N. M., and Rubin, D. B.**, “Maximum likelihood from incomplete data via the EM algorithm,” *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, vol. 39, no. 1, pp. 1–38, 1977.

[56] **Di Cairano, S. and Kolmanovsky, I. V.**, “Further developments and applications of network reference governor for constrained systems,” in *American Control Conference*, 2012, pp. 3907–3912.

[57] **Dorobantu, A., Seiler, P., and Balas, G. J.**, “Time-Delay Margin Analysis for an Adaptive Controller,” *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 5, pp. 1418–1425, 2012.

[58] **Duarte, M. A. and Narendra, K. S.**, “Combined direct and indirect approach to adaptive control,” *IEEE Transactions on Automatic Control*, vol. 34, no. 10, pp. 1071–1075, 1989.

[59] **Duvenaud, D. K.**, “Automatic Model Construction with Gaussian Processes,” Dissertation, University of Cambridge, 2014.

[60] EASA CM-SWCEH-002, “Software Aspects of Certification: Certification Memorandum,” Mar. 2012. [Online] Available: http://acg-solutions.fr/acg/wp-content/uploads/2014/09/EASA-CM-SWCEH-002-Issue-01-Rev-01-Software-Aspects-of-Certification.pdf. Accessed on: Oct. 06 2016.

[61] **Eirola, E. and Lendasse, A.**, “Gaussian Mixture Models for Time Series Modelling, Forecasting, and Interpolation,” *IDA,* https://pdfs.semanticscholar.org/a3cd/8693cd5601d22a2e2a8ec76e494c3e08c533.pdf, 2013.

[62] **Eschenbach, T. G. and Gimpel, R. J.**, “Stochastic Sensitivity Analysis,” *The Engineering Economist*, vol. 35, no. 4, pp. 305–321, 1990.

[63] **Flam, J. T.**, “The Linear Model under Gaussian Mixture Inputs,” Phd Thesis, Faculty of Information Technology, Mathematics and Electrical Engineering, Norwegian University of Science and Technology, Trondheim, Norway, 2013.

[64] **Flam, J. T., Chatterjee, S., Kansanen, K., and Ekman, T.**, “On MMSE Estimation: A Linear Model Under Gaussian Mixture Statistics,” *IEEE Transactions on Signal Processing*, vol. 60, no. 7, pp. 3840–3845, 2012.

[65] **Flam, J. T., Zachariah, D., Vehkapera, M., and Chatterjee, S.**, “The Linear Model Under Mixed Gaussian Inputs: Designing the Transfer Matrix,” *IEEE Transactions on Signal Processing*, vol. 61, no. 21, pp. 5247–5259, 2013.

[66] **Fleischmann, M. and Funke, O.**, *Valles Marineris Explorer – using a robotic swarm to explore Mars.* Available: http://www.dlr.de/dlr/en/desktopdefault.aspx/tabid-10081/151_read-3574/year-all/#/gallery/5799. Accessed on: Apr. 15 2017, 2012.

[67] **Fox, D., Hightower, J., Liao, L., Schulz, D., and Borriello, G.**, “Bayesian filtering for location estimation,” *IEEE Pervasive Comput.*, vol. 2, no. 3, pp. 24–33, 2003.

[68] **Fuller, E., Yerramalla, S., Cukic, B., and Gururajan, S.**, “An approach to predicting non-deterministic neural network behavior,” in *IEEE International Joint Conference on Neural Networks*, Jul. 2005, pp. 2921–2926.

[69] **Funahashi, K.-I.**, “On the approximate realization of continuous mappings by neural networks,” *Neural Networks*, vol. 2, no. 3, pp. 183–192, 1989.

[70] **Gelb, A.**, *Applied optimal estimation*, Cambridge, Mass: M.I.T. Press, 1974.

[71] **Ghassemi, N. H. and Deisenroth, M. P.**, “Analytic Long-Term Forecasting with Periodic Gaussian Processes,” *Journal of Machine Learning Research W&CP (Proceedings of AISTATS)*, no. vol. 33, pp. 303–311, 2014.

[72] **Gibson, T. E., Annaswamy, A. M., and Lavretsky, E.**, “Adaptive systems with closed-loop reference-models, part I: Transient performance,” in *American Control Conference*, 2013, pp. 3376–3383.

[73] **Gibson, T. E., Annaswamy, A. M., and Lavretsky, E.**, “On Adaptive Control With Closed-Loop Reference Models: Transients, Oscillations, and Peaking,” *IEEE Access*, vol. 1, pp. 703–717, 2013.

[74] **Gibson, T. E., Matsutani, M., Annaswamy, A. M., and Crespo, L. G.**, “Internal Algorithm Monitors for Adaptive Systems,” Active Adaptive Controls Laboratory, Cambridge, MA, 2010. [Online] Available: http://www.mit.edu/~tgibson/aac2.pdf. Accessed on: Mar. 21 2016.

[75] **Gilbert, E. G. and Kolmanovsky, I.**, “A generalized reference governor for nonlinear systems,” in *Conference on Decision and Control*, 2001, pp. 4222–4227.

[76] **Gilbert, E. G. and Tan, K. T.**, “Linear systems with state and control constraints: the theory and application of maximal output admissible sets,” *IEEE Trans. Automat. Contr.*, vol. 36, no. 9, pp. 1008–1020, 1991.

[77] **Gilks, W. R., Richardson, S., and Spiegelhalter, D. J.**, *Markov chain Monte Carlo in practice*, Boca Raton, Fla.: Chapman & Hall, 1998.

[78] **Goodfellow, I., Bengio, Y., and Courville, A.**, *Deep learning*, Cambridge, Massachusetts, London, England: MIT Press, 2016.

[79] **Grande, R. C., Chowdhary, G., and How, J. P.**, “Experimental Validation of Bayesian Nonparametric Adaptive Control Using Gaussian Processes,” *Journal of Aerospace Information Systems*, vol. 11, no. 9, pp. 565–578, http://dx.doi.org/10.2514/1.I010190, 2014.

[80] **Grieder, P., Borrelli, F., Torrisi, F., and Morari, M.**, “Computation of the constrained infinite time linear quadratic regulator,” *Automatica*, vol. 40, no. 4, pp. 701–708, 2004.

[81] **Grimmett, G. and** Welsh, D. J. A, *Probability: An introduction*.

[82] **Gupta, P. and Schumann, J.**, “A tool for verification and validation of neural network based adaptive controllers for high assurance systems,” in Los Alamitos, Calif.: IEEE Computer Society, 2004, pp. 277–278.

[83] **Gustafsson, F.**, “Particle filter theory and practice with positioning applications,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 25, no. 7, pp. 53–82, 2010.

[84] **Haddad, W. M. and Chellaboina, V.**, *Nonlinear dynamical systems and control: A Lyapunov-based approach*. Princeton: Princeton University Press, 2008.

[85] **Hardy, J., Havlak, F., and Campbell, M.**, “Multi-step prediction of nonlinear Gaussian Process dynamics models with adaptive Gaussian mixtures,” *The International Journal of Robotics Research*, vol. 34, no. 9, pp. 1211–1227, 2015.

[86] **Harris, T. J., Seppala, C. T., and Desborough, L. D.**, “A review of performance monitoring and assessment techniques for univariate and multivariate control systems,” *Journal of Process Control*, vol. 9, no. 1, pp. 1–17, 1999.

[87] **Heise, C. D.**, “Adaptive Flight Control with Guaranteed Stability and Performance Characteristics,” Dissertation, Institute of Flight System Dynamics, Technische Universität München, Munich, Germany, 2017 expected.

[88] **Heise, C. D., Falconi, G. P., and Holzapfel, F.**, “Hexacopter outdoor flight test results of an Extended State Observer based controller,” in *IEEE International Conference on Aerospace Electronics and Remote Sensing Technology*, 2014, pp. 26–33.

[89] **Helton, J. C., Johnson, J. D., Sallaberry, C. J., and Storlie, C. B.**, “Survey of Sampling-Based Methods for Uncertainty and Sensitivity Analysis,” Sandia National Laboratories, Albuquerque, New Mexico, Jun. 2006. [Online] Available: http://prod.sandia.gov/techlib/access-control.cgi/2006/062901.pdf. Accessed on: Jun. 19 2017.

[90] **Henderson, H. V. and Searle, S. R.**, “On Deriving the Inverse of a Sum of Matrices,” *SIAM Rev.*, vol. 23, no. 1, pp. 53–60, 1981.

[91] **Hennig, P.**, “Animating Samples from Gaussian Distributions,” Max Planck Institute for Intelligent Systems, Spemannstraße, 72076 Tübingen, Germany 8, 2013.

[92] **Holzapfel, F.**, “Nichtlineare adaptive Regelung eines unbemannten Fluggerätes,” Dissertation, Lehrstuhl für Flugmechanik und Flugregelung, Technische Universität München, München, 2004.

[93] **Holzapfel, F.**, *Flight System Dynamics 1&2 - Lecture Slides*, Munich, Germany, 2016.

[94] **Holzapfel, F. and Raffler, T.**, *Model Reference Adaptive Control - Lecture Slides*, Munich, Germany, 2016.

[95] **Holzapfel, F. and Sachs, G.**, “Dynamic Inversion Based Control Concept with Application to an Unmanned Aerial Vehicle,” in *AIAA Guidance, Navigation, and Control Conference*, 2004.

[96] **Hörmander, L.**, *The Analysis of Linear Partial Differential Operators I: Distribution Theory and Fourier Analysis*. Berlin, New York: Springer, 2003.

[97] **Hovakimyan, N. and Cao, C.**, *L 1 adaptive control theory: Guaranteed robustness with fast adaptation*. Philadelphia, Pa.: SIAM, Soc. for Industrial and Applied Mathematics, 2010.

[98] **Iaccarino, G.**, “Introduction to Uncertainty Quantification in Computational Science,” Jun. 1 2009.

[99] **Inman, H. F. and Bradley, E. L.**, “The overlapping coefficient as a measure of agreement between probability distributions and point estimation of the overlap of two normal densities,” *Communications in Statistics - Theory and Methods*, vol. 18, no. 10, pp. 3851–3874, 1989.

[100] **Ioannou, P. A. and Fidan, B.**, *Adaptive control tutorial*, Philadelphia, PA: Society for Industrial and Applied Mathematics, 2006.

[101] **Ioannou, P. A. and Kokotovic, P. V.**, “Instability analysis and improvement of robustness of adaptive control,” *Automatica*, vol. 20, no. 5, pp. 583–594, 1984.

[102] **Ioannou, P. A. and Sun, J.**, *Robust adaptive control*, Upper Saddle River, NJ: PTR Prentice-Hall, 1996.

[103] **Jablonka, A.**, “Stochastic sensitivity analysis for structural dynamics systems via the second-order perturbation,” *Archive of Applied Mechanics*, vol. 86, no. 11, pp. 1913–1926, 2016.

[104] **Jacklin, S.** *et al.*, “Verification, Validation, and Certification Challenges for Adaptive Flight-Critical Control System Software,” in *AIAA Guidance, Navigation, and Control Conference*, 2004.

[105] **Jacklin, S.**, “Closing the Certification Gaps in Adaptive Flight Control Software,” in *AIAA Guidance, Navigation and Control Conference*, 2008.

[106] **Jazwinski, A. H.**, *Stochastic Processes and Filtering Theory*: Academic Press, 1970.

[107] **Jelali, M.**, “An overview of control performance assessment technology and industrial applications,” *Control Engineering Practice*, vol. 14, no. 5, pp. 441–466, 2006.

[108] **Jian-Xin, X., Qing-Wei, J., and Tong, H. L.**, “On the design of a nonlinear adaptive variable structure derivative estimator,” *IEEE Trans. Automat. Contr.*, vol. 45, no. 5, pp. 1028–1033, 2000.

[109] **Joe Qin, S.**, “Control performance monitoring — a review and assessment,” *Computers & Chemical Engineering*, vol. 23, no. 2, pp. 173–186, 1998.

[110] **Johnson, E., Calise, A., El-Shirbiny, H., and Eysdyk, R.**, “Feedback linearization with Neural Network augmentation applied to X-33 attitude control,” in *AIAA Guidance, Navigation, and Control Conference*, 2000.

[111] **Johnson, E. N.**, “Limited Authority Adaptive Flight Control,” PhD Thesis, Georgia Institute of Technology, Atlanta, GA, 2000.

[112] **Johnson, E. N. and Calise, A. J.**, “Pseudo-Control Hedging: A New Method for Adaptive Control,” Advances in Navigation and Control Technology Workshop, Redstone Arsenal, Alabama, Nov. 2000. [Online] Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.16.4719&rep=rep1&type=pdf . Accessed on: Mar. 22 2016.

[113] **Johnson, E. N. and Calise, A. J.**, “Neural network adaptive control of systems with input saturation,” in *American Control Conference*, 2001, 3527-3532 vol.5.

[114] **Johnson, E. N. and Kannan, S. K.**, “Adaptive Trajectory Control for Autonomous Helicopters,” *Journal of Guidance, Control, and Dynamics*, vol. 28, no. 3, pp. 524–538, 2005.

[115] **Kalman, R. E.**, “A New Approach to Linear Filtering and Prediction Problems,” *Journal of Basic Engineering*, vol. 82, no. 1, p. 35, 1960.

[116] **Katzfuss, M., Stroud, J. R., and Wikle, C. K.**, “Understanding the Ensemble Kalman Filter,” *The American Statistician*, vol. 70, no. 4, pp. 350–357, 2016.

[117] **Kerrigan, E. C.**, “Robust Constraint Satisfaction: Invariant Sets and Predictive Control,” Ph.D. Dissertation, Department of Engineering, University of Cambridge, Cambridge, UK, 2000.

[118] **Khalil, H. K.**, *Nonlinear systems,* 3rd ed. Upper Saddle River, N.J: Prentice Hall, 2002.

[119] **Kharisov, E., Gregory, I., Cao, C., and Hovakimyan, N.**, “L1 Adaptive Control for Flexible Space Launch Vehicle and Proposed Plan for Flight Validation,” in *AIAA Guidance, Navigation and Control Conference*, 2008.

[120] **Kim, B.** *et al.*, “Probabilistic Vehicle Trajectory Prediction over Occupancy Grid Map via Recurrent Neural Network,” [Online] Available: http://arxiv.org/pdf/1704.07049v1.

[121] **Kim, K.**, “K-Modification and a Novel Approach to Output Feedback Adaptive Control,” PhD Thesis, Georgia Institute of Technology, Atlanta, GA, 2011.

[122] **Kim, K., Yucelen, T., and Calise, A.**, “K- Modification in Adaptive Control,” in *AIAA Infotech Conference*, 2010.

[123] **Kim, N.**, “Improved Methods in Neural Network-Based Adaptive Output Feedback Control, with Applications to Flight Control,” Ph.D. Dissertation, Georgia Institute of Technology, Atlanta, GA, 2003.

[124] **Kingravi, H. A., Chowdhary, G., Vela, P. A., and Johnson, E. N.**, “A reproducing Kernel Hilbert Space approach for the online update of Radial Bases in neuro-adaptive control,” in *IEEE Conference on Decision and Control and European Control Conference*, 2011.

[125] **Krishnakumar, K.**, “Stability, Maneuverability, and Safe Landing in the Presence of Adverse Conditions,” National Aeronautics and Space Administration, May. 2009. [Online] Available: http://www.aeronautics.nasa.gov/nra_pdf/irac_tech_plan_c1.pdf. Accessed on: Sep. 28 2016.

[126] **Kullback, S.**, *Information Theory and Statistics*: John Wiley & Sons, Inc., 1959.

[127] **Kullback, S. and Leibler, R. A.**, “On Information and Sufficiency,” *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[128] **Lavretsky, E., Gadient, R., and Gregory, I. M.**, “Predictor-Based Model Reference Adaptive Control,” *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 4, pp. 1195–1201, 2010.

[129] **Leonard, J. A., Kramer, M. A., and Ungar, L. H.**, “Using radial basis functions to approximate a function and its error bounds,” *IEEE Transactions on Neural Networks*, vol. 3, no. 4, pp. 624–627, 1992.

[130] **Liao, Y., Fang, S.-C., and Nuttle, H. L.W.**, “Relaxed conditions for radial-basis function networks to be universal approximators,” *Neural Networks*, vol. 16, no. 7, pp. 1019–1028, 2003.

[131] **Limanond, S. and Tsakllis, K. S.**, “Model reference adaptive and nonadaptive control of linear time-varying plants,” *IEEE Transactions on Automatic Control*, vol. 45, no. 7, pp. 1290–1300, 2000.

[132] **Liu, Y., Cukic, B., Fuller, E., Yerramalla, S., and Gururajan, S.**, “Monitoring techniques for an online neuro-adaptive controller,” *Journal of Systems and Software*, vol. 79, no. 11, pp. 1527–1540, 2006.

[133] **Liu, Y., Cukic, B., and Gururajan, S.**, “Validating neural network-based online adaptive systems: a case study,” *Software Quality Journal*, vol. 15, no. 3, pp. 309–326, 2007.

[134] **Liu, Y., Cukic, B., Jiang, M., and Xu, Z.**, “Predicting with Confidence – An Improved Dynamic Cell Structure,” in *Lecture Notes in Computer Science, Advances in Natural Computation*, Berlin, Heidelberg: Springer, 2005, pp. 750–759.

[135] **Löbl, D., Mumm, N., and Holzapfel, F.**, “A Total Capability Approach for Development of Safety-Critical Functions,” in *13th International Conference on Probabilistic Safety Assessment and Management*, Seoul, Korea, 2016.

[135b] **Löbl, D.**, “A Total Capability Approach for the Development of Safety-Critical Functions,” Dissertation, Lehrstuhl für Flugsystemdynamik, Technische Universität München, München, 2018.

[136] **Lorenc, A. C.**, “Analysis methods for numerical weather prediction,” *Q.J Royal Met. Soc.*, vol. 112, no. 474, pp. 1177–1194, 1986.

[137] **Lötstedt, P.**, “Solving the minimal least squares problem subject to bounds on the variables,” *BIT*, vol. 24, no. 2, pp. 205–224, 1984.

[138] **Lu, F., Morzfeld, M., Tu, X., and Chorin, A. J.**, “Limitations of polynomial chaos expansions in the Bayesian solution of inverse problems,” *Journal of Computational Physics*, vol. 282, pp. 138–147, http://arxiv.org/pdf/1404.7188v2, 2015.

[139] **McLachlan, G. J. and Peel, D.**, *Finite mixture models*, New York: Wiley, 2005.

[140] **Middleton, R. H. and Goodwin, G. C.**, “Adaptive control of time-varying linear systems,” *IEEE Transactions on Automatic Control*, vol. 33, no. 2, pp. 150–155, 1988.

[141] MIL-HDBK-1797, “Flying Qualities of Piloted Aircraft,” Department of Defense, Dec. 1997.

[142] **Monahemi, M. M. and Krstic, M.**, “Control of wing rock motion using adaptive feedback linearization,” *Journal of Guidance, Control, and Dynamics*, vol. 19, no. 4, pp. 905–912, 1996.

[143] **Mourikis, A. I. and Roumeliotis, S. I.**, “A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation,” in *IEEE International Conference on Robotics and Automation*, 2007, pp. 3565–3572.

[144] **Movellan, J. R.**, *Discrete Time Kalman Filters and Smoothers*, 2011.

[145] **Mühlegg, M., Chowdhary, G., and Holzapfel, F.**, “Optimizing Reference Commands for Concurrent Learning Adaptive-Optimal Control of Uncertain Dynamical Systems,” in *Guidance, Navigation, and Control and Co-located Conferences, AIAA Guidance, Navigation, and Control (GNC) Conference*: American Institute of Aeronautics and Astronautics, 2013.

[146] **Mühlegg, M., Chowdhary, G., and Holzapfel, F.**, “State monitoring of systems augmented by model reference adaptive control using analytic time-series forecasting,” in *24th Mediterranean Conference on Control and Automation*, 2016, pp. 437–442.

[147] **Mühlegg, M., Chowdhary, G., How, J. P., and Holzapfel, F.**, “Adaptive-Optimal Control of Constrained Nonlinear Uncertain Dynamical Systems using Concurrent Learning Model Predictive Control,” in *Guidance, Navigation, and Control and Co-located Conferences, AIAA Guidance, Navigation, and Control (GNC) Conference*: American Institute of Aeronautics and Astronautics, 2013.

[148] **Mühlegg, M., Niermeyer, P., Falconi, G. P., and Holzapfel, F.**, “L1 fault tolerant adaptive control of a hexacopter with control degradation,” in *Conference on Control Applications*, 2015, pp. 750–755.

[149] **Mühlegg, M., Niermeyer, P., and Holzapfel, F.**, “Reference command shaping for approximate dynamic inversion based model Reference Adaptive Control,” in *IEEE International Conference on Aerospace Electronics and Remote Sensing Technology*, 2014, pp. 179–184.

[150] **Mühlegg, M., Raffler, T., and Holzapfel, F.**, “Quadcopter Flight Test Results of a Consistency Monitoring Algorithm for Adaptive Controllers,” in *International Conference on Control, Automation, Robotics and Vision*, 2016.

[151] Munoz De La Pena, D.**, Bemporad, A., and Filippi, C.**, “Robust Explicit MPC Based on Approximate Multiparametric Convex Programming,” *IEEE Trans. Automat. Contr.*, vol. 51, no. 8, pp. 1399–1403, 2006.

[152] **Murphy, K. P.**, *Machine learning: A probabilistic perspective*. Cambridge, Mass.: MIT Press, 2012.

[153] **Nagy, Z. K. and Braatz, R. D.**, “Distributional uncertainty analysis using polynomial chaos expansions,” in *International Symposium on Computer-Aided Control System Design*: IEEE, 2010, pp. 1103–1108.

[154] **Narendra, K. S. and Annaswamy, A. M.**, “A new adaptive law for robust adaptation without persistent excitation,” *IEEE Transactions on Automatic Control*, vol. 32, no. 2, pp. 134–145, 1987.

[155] **Narendra, K. S. and Annaswamy, A. M.**, *Stable adaptive systems*, Mineola, N.Y: Dover Publications, 2005.

[156] **Narendra, K. S. and Valavani, L. S.**, “Direct and indirect model reference adaptive control,” *Automatica*, vol. 15, no. 6, pp. 653–664, 1979.

[157] National Aeronautics and Space Administration, Ed., “NASA Technology Roadmaps: TA4: Robotics and Autonomous Systems,” Jul. 2015. [Online] Available: http://www.nasa.gov/sites/default/files/atoms/files/2015_nasa_technology_roadmaps_ta_4_robotics_and_autonomous_systems_final.pdf. Accessed on: Oct. 13 2016.

[158] **Nguyen, N., Ishihara, A., Krishnakumar, K., and Bakhtiari-Nejad, M.**, “Bounded Linear Stability Analysis - A Time Delay Margin Estimation Approach for Adaptive Control,” in *AIAA Guidance, Navigation, and Control Conference*, 2009.

[159] **Nguyen, N., Krishnakumar, K., and Boskovic, J.**, “An Optimal Control Modification to Model-Reference Adaptive Control for Fast Adaptation,” in *AIAA Guidance, Navigation and Control Conference*, 2008.

[160] **Nguyen, N. and Summers, E.**, “On Time Delay Margin Estimation for Adaptive Control and Robust Modification Adaptive Laws,” in *AIAA Guidance, Navigation, and Control Conference*, 2011.

[161] **O'Hagan, A.**, “Polynomial Chaos: A Tutorial and Critique from a Statistician's Perspective,” University of Sheffield, UK, 2013. [Online] Available: http://tonyohagan.co.uk/academic/pdf/Polynomial-chaos.pdf. Accessed on: Jun. 27 2017.

[162] **Okoh, C., Roy, R., Mehnen, J., and Redding, L.**, “Overview of Remaining Useful Life Prediction Techniques in Through-life Engineering Services,” *Procedia CIRP*, vol. 16, pp. 158–163, 2014.

[163] **Orchard, M. E. and Vachtsevanos, G. J.**, “A particle-filtering approach for on-line fault diagnosis and failure prognosis,” *Transactions of the Institute of Measurement and Control*, vol. 31, no. 3-4, pp. 221–246, 2009.

[164] **Panayirci, H. M.**, “Efficient solution for Galerkin-based polynomial chaos expansion systems,” *Advances in Engineering Software*, vol. 41, no. 12, pp. 1277–1286, 2010.

[165] **Park, J. and Sandberg, I. W.**, “Universal Approximation Using Radial-Basis-Function Networks,” *Neural Computation*, vol. 3, no. 2, pp. 246–257, 1991.

[166] **Patel, V., Cao, C., Hovakimyan, N., Wise, K., and Lavretsky, E.**, “L1 Adaptive Controller for Tailless Unstable Aircraft in the Presence of Unknown Actuator Failures,” in *AIAA Guidance, Navigation and Control Conference*, 2007.

[167] **Petersen, K. B. and Pedersen, M. S.**, *The Matrix Cookbook.* Available: https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf. Accessed on: Dec. 04 2018, 2012.

[168] **Pfeiffer, P. E.**, *Probability for Applications*, New York, NY: Springer, 1990.

[169] **Phillips, B. and Blackburn, M.**, “Verification Points for Self-adaptive Systems,” *Procedia Computer Science*, vol. 36, pp. 118–123, 2014.

[170] **Pullum, L.** *et al.*, “Verification of Adaptive Systems,” in *Infotech, Infotech@Aerospace 2012*: American Institute of Aeronautics and Astronautics, 2012.

[171] **Quiñonero-Candela, J.**, “Prediction at an Uncertain Input for Gaussian Prediction at an Uncertain Input for Gaussian Processes and Relevance Vector Machines Application to Multiple-Step Ahead Time-Series Forecasting,” Technical Report, Technical University of Denmark, 2003.

[172] **Quiñonero-Candela, J. and Rasmussen, C. E.**, “A Unifying View of Sparse Approximate Gaussian Process Regression,” *J. Mach. Learn. Res.*, vol. 6, pp. 1939–1959, 2005.

[173] **Rasmussen, C. E. and** Williams, Christopher K. I, *Gaussian processes for machine learning*, Cambridge, Mass: MIT Press, 2006.

[174] **Robert, C. P.**, “Monte Carlo Methods,”

[175] **Rogers, D. W. O.**, “Fifty years of Monte Carlo simulations for medical physics,” (eng), *Physics in medicine and biology*, vol. 51, no. 13, R287-301, 2006.

[176] RTCA/DO-178C, “Software Considerations in Airborne Systems and Equipment Certification,” Dec. 2011.

[177] **Rushby, J.**, “How Do we Certify for the Unexpected?,” in *AIAA Guidance, Navigation and Control Conference*, 2008.

[178] **Rushby, J.**, “Runtime Certification,” in *Runtime Verification: 8th International Workshop, RV 2008, Budapest, Hungary, March 30, 2008.*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 21–35.

[179] **Saad, A. A.**, “Simulation and Analysis of Wing Rock Physics for a Generic Fighter Model with three Degrees-of-Freedom,” PhD Thesis, Air Force Institute of Technology, Air University, Wright-Patterson Air Force Base, Dayton, Ohio, 2000.

[180] SAE ARP94910, “Aerospace - Vehicle Management Systems - Flight Control Design, Installation and Test of, Military Unmanned Aircraft, Specification Guide For: Aerospace Recommended Practice,” SAE International, Dec. 2012.

[181] SAE AS94900, “Aerospace - Flight Control Systems - Design, Installation and Test of Piloted Military Aircraft, General Specification For: Aerospace Standard,” SAE International, Jul. 2007.

[182] **Saltelli, A. and Annoni, P.**, “How to avoid a perfunctory sensitivity analysis,” *Environmental Modelling & Software*, vol. 25, no. 12, pp. 1508–1517, 2010.

[183] **Sanner, R. M. and Slotine, J.-J.E.**, “Gaussian networks for direct adaptive control,” *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 837–863, 1992.

[184] **Särkkä, S.**, *Bayesian filtering and smoothing*, Cambridge u.a.: Cambridge Univ. Press, 2013.

[185] **Sastry, S. and Bodson, M.**, *Adaptive control: Stability, convergence, and robustness*. Mineola, N.Y: Dover Publications, 2011.

[186] **Schumann, J. and Gupta, P.**, “Monitoring the Performance of a neuro-adaptive Controller,” in *24th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering*, Jul. 2004, pp. 289–296.

[187] **Schumann, J., Gupta, P., and Jacklin, S.**, “Toward Verification and Validation of Adaptive Aircraft Controllers,” in *2005 IEEE Aerospace Conference*, pp. 1–6.

[188] **Schumann, J. and Liu, Y.**, “Performance Estimation of a Neural Network-Based Controller,” in *Lecture Notes in Computer Science, Advances in Neural Networks - ISNN 2006*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 981–990.

[189] **Schumann, J. and** Yan Liu, “Tools and Methods for the Verification and Validation of Adaptive Aircraft Control Systems,” in *IEEE Aerospace Conference*, 2007, pp. 1–8.

[190] **Schwarz, H.-R. and Köckler, N.**, *Numerische Mathematik,* 5th ed. Wiesbaden: Vieweg+Teubner Verlag, 2004.

[191] **Seanor, B. A.**, “Flight testing of a remotely piloted vehicle for aircraft parameter estimation purposes,” PhD Thesis, West Virginia University, Morgantown, WV, 2001.

[192] **Searle, S. R.**, *Matrix algebra useful for statistics*, New York: Wiley, 1982.

[193] **Seber, G. A. F. and Lee, A. J.**, *Linear Regression Analysis*, Hoboken, NJ, USA: John Wiley & Sons, Inc, 2003.

[194] **Seiferth, D., Chowdhary, G., Mühlegg, M., and Holzapfel, F.**, “Online Gaussian Process regression with non-Gaussian likelihood,” in *American Control Conference*, 2017, pp. 3134–3140.

[195] **Sharma, M., Lavretsky, E., and Wise, K.**, “Application and Flight Testing of an Adaptive Autopilot on Precision Guided Munitions,” in *AIAA Guidance, Navigation, and Control Conference*, 2006.

[196] **Singh, S. N., Yirn, W., and Wells, W. R.**, “Direct Adaptive and Neural Control of Wing-Rock Motion of Slender Delta Wings,” *Journal of Guidance, Control, and Dynamics*, vol. 18, no. 1, pp. 25–30, 1995.

[197] **Slotine, J.-J. E. and Li, W.**, *Applied nonlinear control*, Englewood Cliffs, N.J.: Prentice Hall, 1991.

[198] **Solak, E., Murray-smith, R., Leithead, W. E., Leith, D. J., and Rasmussen, C. E.**, “Derivative Observations in Gaussian Process Models of Dynamic Systems,” in *Advances in Neural Information Processing Systems 15*: MIT Press, 2003, pp. 1057–1064.

[199] **Sorenson, H. W. and Alspach, D. L.**, “Recursive bayesian estimation using gaussian sums,” *Automatica*, vol. 7, no. 4, pp. 465–479, 1971.

[200] **Stevens, B. L. and Lewis, F. L.**, *Aircraft control and simulation,* 2nd ed. Hoboken, NJ: Wiley, 2003.

[201] **Stuttle, M. N.**, “A Gaussian Mixture Model Spectral Representation for Speech Recognition,” PhD Thesis, Cambridge University Engineering Department, University of Cambridge, Cambridge, England, 2003.

[202] **Szukalski, R., Mühlegg, M., and Johnson, E. N.**, “Consistency monitoring of Adaptive Controllers using Bayesian linear regression,” in *American Control Conference*, 2015, pp. 177–182.

[203] **Taboga, M.**, *Lectures on probability theory and mathematical statistics,* 2nd ed. s.l., 2012.

[204] **Tao, G.**, *Adaptive control design and analysis*, Hoboken, N.J: Wiley-Interscience, 2003.

[205] **Taylor, B. J., Darrah, M. A., and Moats, C. D.**, “Verification and validation of neural networks: a sampling of research in progress,” in *AeroSense 2003*: SPIE, 2003, p. 8.

[206] **Terejanu, G., Singla, P., Singh, T., and Scott, P. D.**, “Uncertainty Propagation for Nonlinear Dynamic Systems Using Gaussian Mixture Models,” *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 6, pp. 1623–1633, 2008.

[207] **Thorarinsdottir, T. L., Gneiting, T., and Gissibl, N.**, “Using Proper Divergence Functions to Evaluate Climate Models,” *SIAM/ASA J. Uncertainty Quantification*, vol. 1, no. 1, pp. 522–534, 2013.

[208] **Tøndel, P., Johansen, T. A., and Bemporad, A.**, “An algorithm for multi-parametric quadratic programming and explicit MPC solutions,” *Automatica*, vol. 39, no. 3, pp. 489–497, 2003.

[209] **Tsakalis, K. S. and Ioannou, P. A.**, “Adaptive control of linear time-varying plants: a new model reference controller structure,” *IEEE Transactions on Automatic Control*, vol. 34, no. 10, pp. 1038–1046, 1989.

[210] **Tylavsky, D. J. and Sohie, G.R.L.**, “Generalization of the matrix inversion lemma,” *Proc. IEEE*, vol. 74, no. 7, pp. 1050–1052, 1986.

[211] **van der Merwe, R.**, “Sigma-point Kalman Filters for Probabilistic Inference in Dynamic State-space Models,” Oregon Health & Science University, Portland, OR, 2004.

[212] **Volyanskyy, K. Y., Calise, A. J., and** Bong-Jun Yang, “A novel Q-modification term for adaptive control,” in *American Control Conference*, 2006, pp. 4072–4076.

[213] **Volyanskyy, K. Y., Haddad, W. M., and Calise, A. J.**, “A new neuroadaptive control architecture for nonlinear uncertain dynamical systems: Beyond σ- and e-modifications,” in *IEEE Conference on Decision and Control*, 2008, pp. 80–85.

[214] **Volyanskyy, K. Y., Haddad, W. M., and Calise, A. J.**, “A new neuroadaptive control architecture for nonlinear uncertain dynamical systems: beyond sigma- and e-modifications,” (eng), *IEEE Transactions on Neural Networks*, vol. 20, no. 11, pp. 1707–1723, 2009.

[215] **Wang, J., Hertzmann, A., and Blei, D. M.**, “Gaussian Process Dynamical Models,” in *Advances in Neural Information Processing Systems*: MIT Press, 2006, pp. 1441–1448.

[216] **Wang, L.**, *Model Predictive Control System Design and Implementation Using MATLAB®*, London, Heidelberg: Springer, 2009.

[217] **Wells, C.**, *The Kalman filter in finance*, Dordrecht: Kluwer, 2010.

[218] **West, M. and Harrison, J.**, *Bayesian forecasting and dynamic models,* 2nd ed. New York: Springer, 1997.

[219] **Wiener, N.**, “The Homogeneous Chaos,” *American Journal of Mathematics*, vol. 60, no. 4, p. 897, 1938.

[220] **Wilkinson, C., Lynch, J., and Bharadwaj, R.**, “Regulatroy Considerations for Adaptive Systems,” Technical Report NASA/CR-2013-218010, Jun. 2013. [Online] Available: http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20140000469.pdf. Accessed on: Aug. 18 2015.

[221] **Wise, K., Lavretsky, E., Hovakimyan, N., Cao, C., and Wang, J.**, “Verifiable Adaptive Control: UCAV and Aerial Refueling,” in *AIAA Guidance, Navigation and Control Conference*, 2008.

[222] **Yerramalla, S., Cukic, B., Campa, G., Napolitano, M., and Fuller, E.**, “Stability Monitoring and Analysis of Learning in Adaptive Systems,” in *14th Mediterranean Conference on Control and Automation*, 2006, pp. 1–7.

[223] **Yucelen, T.**, “Advances in Adaptive Control Theory: Gradient- and Derivative-Free Approaches,” PhD Thesis, Georgia Institute of Technology, Atlanta, GA, 2012.

[224] **Yucelen, T., Calise, A., Haddad, W., and Volyanskyy, K.**, “A Comparison of a New Neuroadaptive Controller Architecture with the sigma- and e- Modification Architectures,” in *AIAA Guidance, Navigation, and Control Conference*, 2009.

[225] **Zollitsch, A. W., Holzapfel, F., and Annaswamy, A. M.**, “Application of adaptive control with closed-loop reference models to a model aircraft with actuator dynamics and input uncertainty,” in *American Control Conference*, 2015, pp. 3848–3853.

Appendix

A Mathematical Preliminaries

This section provides a general overview of the most important mathematical concepts used within this thesis.

A.1 Properties of the Trace Operator

Let $\boldsymbol{M}$ denote a square matrix and let $c \neq 0$ denote a scalar value. The application of the trace operator $Tr(\cdot)$ yields

$$Tr(\boldsymbol{M}) = Tr(\boldsymbol{M}^T) = \frac{1}{c} Tr(c\boldsymbol{M}). \tag{A-1}$$

Applying the trace operator to the multiplication of two vector signals $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$ yields

$$\boldsymbol{a}^T \boldsymbol{b} = Tr(\boldsymbol{b}\boldsymbol{a}^T). \tag{A-2}$$

A.2 LaSalle’s Invariance Principle

Consider the autonomous system

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) \tag{A-3}$$

where $f: D \to \mathbb{R}^n$ is a locally Lipschitz map from a domain $D \subset \mathbb{R}^n$ into $\mathbb{R}^n$.

Theorem A.1 LaSalle’s Invariance Principle (see [118, p. 128], Theorem 4.4)

Let $\Omega \subset D$ be a compact set that is positively invariant with respect to (A-3). Let $V: D \to \mathbb{R}$ be a continuously differentiable function such that $\dot{V}(\boldsymbol{x}) \leq 0$ in Ω. Let E be the set of all points in Ω where $\dot{V}(\boldsymbol{x}) = 0$. Let M be the largest invariant set in E. Then every solution starting in Ω approaches M as $t \to \infty$.

Proof:

From [118, p. 128], Theorem 4.4: Let $\boldsymbol{x}(t)$ be a solution of (A-3) starting in Ω. Since $\dot{V}(\boldsymbol{x}) \leq 0$ in Ω, $V\big(\boldsymbol{x}(t)\big)$ is a decreasing function of t. Since $V(\boldsymbol{x})$ is continuous on the compact set Ω, it is bounded from below on Ω. Therefore, $V\big(\boldsymbol{x}(t)\big)$ has a limit a as $t \to \infty$. Note also that the positive limit set L^+ (see [118, p. 127], Lemma 4.1) is in Ω because Ω is a closed set. For any $p \in L^+$, there is a sequence t_n with $t_n \to \infty$ and $\boldsymbol{x}(t_n) \to p$ as $n \to \infty$. By continuity of $V(\boldsymbol{x})$, $V(p) = \lim_{n\to\infty} V\big(\boldsymbol{x}(t_n)\big) = a$. Hence, $V(\boldsymbol{x}) = a$ on L^+. Since L^+ is an invariant set, $\dot{V}(\boldsymbol{x}) = 0$ on L^+. Thus

$$L^+ \subset M \subset E \subset \Omega. \tag{A-4}$$

Since $\boldsymbol{x}(t)$ is bounded, $\boldsymbol{x}(t)$ approaches L^+ as $t \to \infty$ (see [118, p. 127], Lemma 4.1). Hence, $\boldsymbol{x}(t)$ approaches M as $t \to \infty$.

□

A.3 Persistency of Excitation

In the following the concept of excited and persistently excited vector signals is introduced. Equivalent definitions for discrete signals can be found in [204, p. 120].

Definition A.1 Excited and Persistently Excited Signals ([204, p. 108])

A bounded vector signal $\boldsymbol{\Phi}(\mathrm{t}) \in \mathbb{R}^l, l \geq 1$, is *exciting* over the finite time interval $[\sigma_0, \sigma_0 + \Delta\mathrm{t}]$, $\Delta\mathrm{t} > 0, \sigma_0 \geq \mathrm{t}_0$, if for some $\alpha_0 > 0$

$$\int_{\sigma_0}^{\sigma_0+\Delta\mathrm{t}} \boldsymbol{\Phi}(\tau)\boldsymbol{\Phi}^T(\tau)\, d\tau \geq \alpha_0 \boldsymbol{I}. \tag{A-5}$$

A bounded vector signal $\boldsymbol{\Phi}(\mathrm{t}) \in \mathbb{R}^l, l \geq 1$, is *persistently exciting* if there exist $\delta > 0$ and $\alpha_0 > 0$ such that

$$\int_{\sigma}^{\sigma+\delta} \boldsymbol{\Phi}(\tau)\boldsymbol{\Phi}^T(\tau)\, d\tau \geq \alpha_0 \boldsymbol{I}, \forall \sigma \geq \mathrm{t}_0 \tag{A-6}$$

A.4 Special Matrix Inverses

Throughout the thesis various special matrix inverses are used. Refer to [192] for a concise overview of common matrix inverses. Fundamentally, for any invertible matrices $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{n\times n}$

$$(\boldsymbol{AB})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1} \tag{A-7}$$

If a matrix can be partitioned into blocks, it can be inverted such that

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{bmatrix}^{-1} = \begin{bmatrix} (\boldsymbol{A} - \boldsymbol{BD}^{-1}\boldsymbol{C})^{-1} & (\boldsymbol{A} - \boldsymbol{BD}^{-1}\boldsymbol{C})^{-1}\boldsymbol{BD}^{-1} \\ -\boldsymbol{D}^{-1}\boldsymbol{C}(\boldsymbol{A} - \boldsymbol{BD}^{-1}\boldsymbol{C})^{-1} & \boldsymbol{D}^{-1} + \boldsymbol{D}^{-1}\boldsymbol{C}(\boldsymbol{A} - \boldsymbol{BD}^{-1}\boldsymbol{C})^{-1}\boldsymbol{BD}^{-1} \end{bmatrix} \tag{A-8}$$

Here, $\boldsymbol{A}$ as well as $\boldsymbol{A} - \boldsymbol{BD}^{-1}\boldsymbol{C}$ are assumed to be nonsingular. The inverse $(\boldsymbol{A} - \boldsymbol{BD}^{-1}\boldsymbol{C})^{-1}$ is known as the Woodbury matrix inverse or matrix inversion lemma and is given by

$$(\boldsymbol{A} - \boldsymbol{BD}^{-1}\boldsymbol{C})^{-1} = \boldsymbol{A}^{-1} + \boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{D} - \boldsymbol{CA}^{-1}\boldsymbol{B})^{-1}\boldsymbol{CA}^{-1}. \tag{A-9}$$

A derivation and proof can be found amongst others in [90], [210]. The afore mentioned derivations are particularly important with calculations including covariance matrices.

A typical inversion problem is

$$\begin{bmatrix} \boldsymbol{P}^{-1}+\boldsymbol{H}^T\boldsymbol{R}^{-1}\boldsymbol{H} & -\boldsymbol{H}^T\boldsymbol{R}^{-1} \\ -\boldsymbol{R}^{-1}\boldsymbol{H} & \boldsymbol{R}^{-1} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{P} & \boldsymbol{P}\boldsymbol{H}^T \\ \boldsymbol{H}\boldsymbol{P} & \boldsymbol{H}\boldsymbol{P}\boldsymbol{H}^T+\boldsymbol{R} \end{bmatrix}. \tag{A-10}$$

A.5 Cholesky factorization

The Cholesky factorization, also called Cholesky decomposition, decomposes matrices with special properties into the product of two triangular matrices. In particular, assume $\boldsymbol{A}$ to be a symmetric positive definite square matrix. Following e.g. [190], the Cholesky factorization of $\boldsymbol{A}$ yields

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{U}^T\boldsymbol{U}. \tag{A-11}$$

Here, $\boldsymbol{L}$ represents a lower triangular matrix, while $\boldsymbol{U}$ represents an upper triangular matrix. The Cholesky factorization can be used to compute the inverse of $\boldsymbol{A}$ efficiently. The reason for this is that the inverse of a triangular matrix is easy to compute. The inverse of $\boldsymbol{A}$ using Cholesky factorization is

$$\boldsymbol{A}^{-1} = (\boldsymbol{L}^{-1})^T\boldsymbol{L}^{-1} = \boldsymbol{U}^{-1}(\boldsymbol{U}^{-1})^T. \tag{A-12}$$

A.6 Taylor Series Expansion

The Taylor series expansion (see e.g. [96]) is a series expansion of a function $f(x)$ about a point x_0. In general the Taylor series expansion of $f(x)$ at $x = x_0$ is

$$f(x) = \sum_{n=1}^{\infty} \frac{f^{(n)}(x_0)}{n!}(x - x_0), \tag{A-13}$$

where $f^{(n)}(x_0)$ represents the n –th derivative of $f(x)$ evaluated at $x = x_0$. As an example, the Taylor series expansion of $f(x_1, \dots, x_d)$ about $x_j = x_{j,0}, j \in (1, d)$ is

$$\begin{aligned} f(x_1, \dots, x_d) &= f(x_{1,0}, \dots, x_{d,0}) + \frac{1}{1!}\sum_{j=1}^{d} \frac{\partial f(x_1, \dots, x_d)}{\partial x_j}(x_j - x_{j,0}) \\ &+ \frac{1}{2!}\sum_{j=1}^{d}\sum_{k=1}^{d} \frac{\partial^2 f(x_1, \dots, x_d)}{\partial x_j \partial x_k}(x_j - x_{j,0})(x_k - x_{k,0}) + \cdots \end{aligned} \tag{A-14}$$

A.7 Matrix and vector derivatives

This section details the three most important matrix derivatives, which are used throughout the thesis. They are particularly important for the linearization of the nonlinear system dynamics. For this purpose let $\boldsymbol{x}$ denote a state vector, let $\boldsymbol{a}$ denote a vector of fixed parameters and let $\boldsymbol{B}$ denote a matrix of fixed parameters with appropriate dimensions. Following e.g. [167], the most important derivatives used within this thesis are

$$\frac{\partial \boldsymbol{x}^T \boldsymbol{a}}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{a}^T \boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{a}$$

$$\frac{\partial \boldsymbol{x}^T \boldsymbol{B}}{\partial \boldsymbol{x}} = \frac{\partial (\boldsymbol{B}^T \boldsymbol{x})^T}{\partial \boldsymbol{x}} = \boldsymbol{B}^T \tag{A-15}$$

$$\left.\frac{\partial \boldsymbol{f}(\boldsymbol{x})\boldsymbol{x}^T \boldsymbol{B}}{\partial \boldsymbol{x}}\right|_{x=x_l} = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|_{x=x_l} \boldsymbol{x}_l^T \boldsymbol{B} + \boldsymbol{f}(\boldsymbol{x}_l)\boldsymbol{B}^T.$$

B Basics of Probability Theory

This section introduces the basics of probability theory required throughout this thesis. Most of the theory can be found in books about probability theory, such as [168], [20, p. 445], [81, p. 5], [52], [51]. The notions are derived for continuous random variables, but are similary applicable to the discrete case.

B.1 Probability Space

Let Ω denote the *sample space* of a particular experiment. The *sample space* describes all possible outcomes of an experiment ξ. Each member of Ω is an elementary event and denoted by ω. Let A denote an event, which may be a set of elementary events ω. Define $\mathcal{F} = \{A_i | i \in \mathbb{N}^*\}$ to be a collection of subsets of Ω. Then $\mathcal{F}$ is called *event space* and defined as follows.

Definition B.1 – Event Space ([81, p. 5], Definition 1.1)

The collection $\mathcal{F}$ of subsets of the sample space Ω is called an *event space* if

a) $\mathcal{F}$ is non-empty,
b) $A \in \mathcal{F} \Rightarrow \Omega \backslash A \in \mathcal{F}$,
c) $A_1, A_2, \ldots \in \mathcal{F} \Rightarrow \bigcup_{i=1}^{\infty} A_i \in \mathcal{F}$.

With the sample space and event space defined, the notion of probability can be introduced.

Definition B.2 – Probability ([81, p. 6], Definition 1.13)

A mapping $\mathbb{P}: \mathcal{F} \rightarrow \mathbb{R}$ is called a *probability (measure)* on $(\Omega, \mathcal{F})$ if

a) $\mathbb{P}(A) \geq 0 \; \forall \; A \in \mathcal{F}$,
b) $\mathbb{P}(\Omega) = 1$ and $\mathbb{P}(\emptyset)$,
c) $A_1, A_2, \ldots$ are disjoint events in $\mathcal{F}$ (in that $A_i \cap A_j = \emptyset$ whenever $i \neq j$) then

$$\mathbb{P}\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mathbb{P}(A_i). \quad \textbf{(B-1)}$$

In essence, the function $\mathbb{P}(A)$ describes with which probability the event A occurs. Definition B.2 lists conditions that the probability function $\mathbb{P}(A)$ has to satisfy. The sample space, the event space and the notion of probability are combined to form the probability space.

Definition B.3 – Probability Space ([81, p. 7], Definition 1.18)

A *probability space* is a triple $(\Omega, \mathcal{F}, \mathbb{P})$ of objects such that

a) Ω is a non-empty set,
b) $\mathcal{F}$ is an event space of subsets of Ω,
c) $\mathbb{P}$ is a probability measure on $(\Omega, \mathcal{F})$.

B.2 Random Variable, Cumulative Distribution Function and Probability Density Function

This section introduces the notion of a random variables, the cumulative distribution and the probability density function. First, in order to apply stochastic uncertainty to physical states, the notion of a random variable is introduced.

Definition B.4 – Random Variable ([52, p. 8], Definition 1.6)

Let Ω be a sample space corresponding to some experiment ξ and let $X\colon \Omega \rightarrow \mathbb{R}$ be a function from the sample space to the real line. Then X is called a *random variable* (rv).

Instead of a single event, it is often more interesting to find the probability of the random variable lying in a certain area. Therefore the cumulative distribution function is introduced.

Definition B.5 – Cummulative Distribution Function ([81, p. 62], Definition 5.2)

If X is a random variable on $(\Omega, \mathcal{F}, \mathbb{P})$, the *cumulative distribution function* (cdf) of X is the function $F_X(x)\colon \mathbb{R} \rightarrow [0,1]$ defined by

$$F_X(x) = \mathbb{P}(X \leq x). \tag{B-2}$$

Note, that the cdf is often simply referred to as *distribution function*. Also note, that the notation x corresponds to an event of X on $(\Omega, \mathcal{F}, \mathbb{P})$. The following definition introduces the probability density function, which completely describes a distribution.

Definition B.6 – Probability Density Function ([52, p. 36], Definition 1.22)

Let X be a real-valued random variable taking values in $\mathbb{R}$. A function $p(x)$ is called the *probability density function* (pdf) of X if

$$\mathbb{P}(a \leq X \leq b) = \int_a^b p(x) dx\,, -\infty < a \leq b < \infty. \tag{B-3}$$

Note, that the pdf is often simply referred to as *density function*. The equivalent to the pdf for discrete rvs is the probability mass function (see e.g. [52, p. 9], Definition 1.7). Note, that if $p(x)$ is the density function of the rv X, then the following properties hold:

$$p(x) \geq 0 \forall x; \quad \int_{-\infty}^{\infty} p(x) dx = 1. \tag{B-4}$$

Depending on the application a huge variety of pdfs is available. This includes amongst others the normal distribution (see Appendix C), the binomial distribution, the Poisson distribution, the Gamma distribution etc. This thesis predominantly employs the normal distribution, as it possesses mathematical properties, which allow for an easy and fast manipulation.

B.3 Multivariate Distributions

Many problems require the introduction of more than one rv. This section introduces the concepts required to deal with more than one rv. Therefore, let $X \in \mathbb{R}$ and $Y \in \mathbb{R}$ be rvs on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$. The following deal with the vector $(X, Y) \in \mathbb{R}^2$. Note, that the concept can be extended to arbitrary dimension.

Definition B.7 – Joint Distribution ([81, p. 83], Definition 6.1)

The *joint distribution* of the pair X, Y of random variables is the mapping $F_{X,Y}: \mathbb{R}^2 \rightarrow [0,1]$ given by

$$F_{X,Y}(x, y) = \mathbb{P}(X \leq x, Y \leq y). \tag{B-5}$$

Consequently, the *joint density function* of the pair (X, Y) can be defined as follows.

Definition B.8 – Joint Density Function (based on [81, p. 86], Definition 6.16)

Let (X, Y) be random variables on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$. If their joint distribution function is expressible in the form

$$F_{X,Y}(x, y) = \mathbb{P}(X \leq x, Y \leq y) = \int\limits_{u=-\infty}^{x} \int\limits_{v=-\infty}^{y} p(u, v) du dv \tag{B-6}$$

for $x, y \in \mathbb{R}$ and some function $p: \mathbb{R}^2 \rightarrow [0, \infty)$, then $p(x, y)$ is the *joint density function* of (X, Y).

While the joint density function contains all relevant information, often information about a single random variable is sought. Therefore, the following definitions introduce the marginal and conditional density functions. The former seeks the density function of one variable only, e.g. $p(x)$. This is achieved by incorporating all possible outcomes for Y.

Definition B.9 – Marginal Density Function (based on [203])

Let (X, Y) be continuous rvs with joint probability density $p(x, y)$. Then the probability density function of the random variable X is called *marginal density function*.

The marginal density function can be obtained from the joint by integrating $p(x, y)$ with respect to all variables except x:

$$p(x) = \int_{-\infty}^{\infty} p(x, y) dy. \tag{B-7}$$

This also applies for $n, n > 2$, rvs. Consider n rvs $X_1, \dots, X_n$ with joint density function $p(x_1, \dots, x_n)$. Hence, the marginal density function for X_1 is

$$p(x_1) = \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} p(x_1, \dots, x_n) dx_n \dots dx_2. \tag{B-8}$$

Contrary to the marginal density, which obtains $p(x)$ from $p(x,y)$ by considering all possible events for Y, the conditional density function seeks the density for X given that a certain event for Y occurred.

Definition B.10 – Conditional Density Function (based on [203])

Let (X,Y) be rvs. The function $p(x|y)\colon \mathbb{R} \to [0,\infty)$ is the conditional density function of X given $Y = y$ if, for any interval $[a,b] \subseteq \mathbb{R}$,

$$\mathbb{P}(X \in [a,b] | Y = y) = \int_a^b p(x|y) dx \tag{B-9}$$

and $p(x|y)$ is such that the above integral is well defined.

The conditional density function is obtained by

$$p(x|y) = p(x|Y = y) = \frac{p(x,y)}{p(y)}, p(y) > 0. \tag{B-10}$$

The conditional density function insinuates the idea that knowing a certain event $Y = y$ occurred, leads to a change in the probability of $X = x$ occurring. While this is often true, for many problems conveying information about Y doesn't change the probability of X. This leads to the notion of independence of two random variables.

Definition B.11 – **Independence** (based on [203])

Two random variables X and Y are said to be *independent*, iff

$$\mathbb{P}([X \in A] \cap [Y \in B]) = \mathbb{P}([X \in A])\mathbb{P}([Y \in B]) \tag{B-11}$$

For any couple of events $[X \in A]$ and $[Y \in B]$, where $A \subseteq \mathbb{R}$ and $B \subseteq \mathbb{R}$**.**

A criterion for independence of the two rvs X and Y is (see e.g. [203]):

$$p(x,y) = p(x)p(y), \forall x,y \in \mathbb{R}. \tag{B-12}$$

B.4 Moments, Expected Value and (Co-)Variance

It is often difficult or even impossible to use the complete distribution in calculations. Rather, supplementary features of the distribution are sought, which are easily interpreted and can be used to simplify calculations. One such family of quantitative measures is the statistical moments.

Definition B.12 – Moment ([81, p. 7], Definition 4.20)

Let $k \geq 1$. The kth *moment* of the random variable X is $\mathbb{E}_x[X^k]$.

Probably the single most important moment of a probability distribution is its expected value, or mean, which corresponds to its first moment.

Definition B.13 – Expected Value (or mean) ([51, p. 147], Definition 7.9)

Let X be a continuous random variable with a pdf $p(x)$. The *expected value (*or *mean)* μ of X is defined as

$$\mathbb{E}_x[X] = \int_{-\infty}^{\infty} xp(x)dx, \tag{B-13}$$

If

$$\int_{-\infty}^{\infty} |x|p(x)dx < \infty. \tag{B-14}$$

The mean essentially describes the position of the center of gravity of the distribution. It is not to be confused with the median, which divides the distribution into two equally large parts, and the mode, which describes the extrema of a distribution. The expected value alone is not enough to describe a distribution. In fact, an infinite number of moments are required in order to fully capture an arbitrary distribution. A special case is the normal distribution (introduced in Appendix C), since it is fully described by its first moment and second *central* moment. In contrast to the moments of Definition B.12, the central moments characterize distributions relative to their mean μ.

Definition B.14 – Central Moments ([51, p. 88], Definition 5.4)

Let X be a random variable with finite kth moment for some $k \geq 1$. Then the kth *central moment* of X is defined as $\mu_k = \mathbb{E}_x[(X-\mu)^k]$, where $\mathbb{E}_x[X] = \mu$.

Note that the first central moment is zero, since $\mathbb{E}_x[(X-\mu)] = 0$. The second central moment is called variance and is key to the description of a normal distribution.

Definition B.15 – Variance ([51, p. 63], Definition 4.9)

Let X be a random variable with finite mean μ. The *variance* of X is defined as

$$\mathbb{V}_x[X] = \mathbb{E}_x[(X-\mu)^2]. \tag{B-15}$$

The previous case only considers scalar rvs X. However, in the presence of a random vector, Definition B.15 does not account for dependencies between different rvs within the random vector. Hence, the concept of variance needs to be extended. The following definition introduces the concept of covariance based on a random vector consisting of two random variables X, Y.

Definition B.16 – Covariance (based on [52, p. 107], Definition 2.7)

Let X, Y be two random variables defined on a common sample space Ω. The *covariance* of X and Y is defined as

$$\mathbb{C}_{x,y}[X,Y] = \mathbb{E}_{x,y}\left[\left(X - \mathbb{E}_x(X)\right)\left(Y - \mathbb{E}_y(Y)\right)\right]. \tag{B-16}$$

The "[c]ovariance is a measure of whether two random variables X, Y tend to increase or decrease together" [52, p. 107]. The second central moment of a random vector is described by the covariance matrix. The latter combines the variance and covariance and is defined as follows.

Definition B.17 – Covariance Matrix

Let $\boldsymbol{Z} = [X \quad Y]$ be a random vector consisting of two random variables X, Y, which are defined on a common sample space Ω. The *covariance matrix* $\boldsymbol{\Sigma}$ of X and Y is defined as

$$\boldsymbol{\Sigma} = \mathbb{V}_Z[\boldsymbol{Z}] = \begin{bmatrix} \mathbb{V}_x[X] & \mathbb{C}_{x,y}[X,Y] \\ \mathbb{C}_{y,x}[Y,X] & \mathbb{V}_y[Y] \end{bmatrix}. \tag{B-17}$$

Note that the covariance matrix contains the variance of each rv on its main diagonal. This thesis often calls the covariance matrix *variance matrix* or simply *covariance*. It should be clear from the context that using the operator $\mathbb{V}_{\cdot}[\cdot]$ on a random vector leads to a covariance matrix.

B.5 Conditional Expectation and Variance

Section B.3 introduced multivariate distributions and the notion of conditional distributions. Furthermore, section B.4 introduced the concept of moments. This section combines both concepts to introduce the conditional expectation and the conditional variance, which will play an integral role in derivations throughout the thesis.

Let $\boldsymbol{Z} = [X \quad Y]$ be a random vector with a multivariate probability distribution. The governing distribution can be fully described by its moments. However, the occurrence of an event $X = x$ inevitably affects the probability distribution of Y. The conditional expectation describes this effect and is defined as follows.

Definition B.18 – Conditional Expectation ([81, p. 99], Definition 6.68)

The *conditional expectation of Y given $X = x$,* written $\mathbb{E}_y[Y|X = x]$, is the mean of the conditional density function

$$\mathbb{E}_y[Y|X = x] = \int_{-\infty}^{\infty} yp(y|x)dy = \int_{-\infty}^{\infty} y\frac{p(x,y)}{p(x)}dy, \tag{B-18}$$

Valid for any value of x for which $p(x) > 0$.

This thesis predominantly relies on the normal distribution, which is introduced in Appendix C. Hence, Similarly, s

Definition B.19 – Conditional Variance ([52, p. 295], Definition 12.7)

Let X, Y have a joint density $p(x,y)$. The *conditional variance* of Y given $X = x$ is defined as

$$\mathbb{V}_y[Y|X = x] = \int_{-\infty}^{\infty} (y - \mu_y)^2 p(y|x)dy = \int_{-\infty}^{\infty} (y - \mu_y)^2 \frac{p(x,y)}{p(x)}dy, \tag{B-19}$$

$\forall x$ such that $p(x) > 0$, where μ_y denotes $\mathbb{E}_y[Y|X = x]$.

B.6 Law of Iterated Expectation and Total Variance

The law of total probability is a fundamental law in probability theory, which relates marginal and conditional probabilities. For this thesis, two paradigms falling under this concept are important, namely the law of iterated expectations and the law of total variance. Consider two jointly distributed random variables X, Y. In the case that the conditional expectation $\mathbb{E}[Y|X]$ is known, one way of finding $\mathbb{E}[Y]$ is to employ the law of iterated expectations.

Theorem B.1 – Law of iterated expectations (based on [81, p. 99], Theorem 6.69)

If X and Y are jointly continuous random variables, then

$$\mathbb{E}[Y] = \mathbb{E}_X\big[\mathbb{E}[Y|X]\big]. \tag{B-20}$$

Proof:

The following identities hold:

$$\begin{aligned}\mathbb{E}_X\big[\mathbb{E}[Y|X]\big] &= \int_{-\infty}^{\infty}\left(\int_{-\infty}^{\infty} yp(y|x)dy\right)p(x)dx \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} yp(y|x)p(x)dydx \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} yp(x,y)dydx \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} yp(x|y)p(y)dydx \\ &= \int_{-\infty}^{\infty} y\left(\int_{-\infty}^{\infty} p(x|y)dx\right)p(y)dy \\ &= \mathbb{E}[Y]\end{aligned} \tag{B-21}$$

□

Similarly, in the case that the conditional expectation $\mathbb{E}[Y|X]$ and variance $\mathbb{V}[Y|X]$ are known, one way of finding $\mathbb{V}[Y]$ is to employ the law of total variance.

Theorem B.2 – Law of total variance (based on [30, p. 344])

If X and Y are jointly continuous random variables, then

$$\mathbb{V}[Y] = \mathbb{E}_x\big[\mathbb{V}[Y|X]\big] + \mathbb{V}_x\big[\mathbb{E}[Y|X]\big]. \tag{B-22}$$

Proof:

Recall the definition of variance:

$$\mathbb{V}[Y] = \mathbb{E}[Y^2] - (\mathbb{E}[Y])^2. \tag{B-23}$$

Similarly, the conditional variance $\mathbb{V}[Y|X]$ is

$$\mathbb{V}[Y|X] = \mathbb{E}[Y^2|X] - (\mathbb{E}[y|x])^2. \tag{B-24}$$

Taking the expectation and applying the law of total expectation yields

$$\mathbb{E}_x\big[\mathbb{V}[Y|X]\big] = \mathbb{E}[Y^2] - \mathbb{E}_x[(\mathbb{E}[Y|X])^2]. \tag{B-25}$$

Similarly, taking the variance of the conditional expectation yields

$$\mathbb{V}_x\big[\mathbb{E}[Y|X]\big] = \mathbb{E}_x[(\mathbb{E}[Y|X])^2] - (\mathbb{E}[Y])^2. \tag{B-26}$$

Summing up (B-25) and (B-26) yields

$$\begin{aligned}\mathbb{E}_x\big[\mathbb{V}[Y|X]\big] + \mathbb{V}_x\big[\mathbb{E}[Y|X]\big] &= \mathbb{E}[Y^2] - \mathbb{E}_x[(\mathbb{E}[Y|X])^2] + \mathbb{E}_x[(\mathbb{E}[Y|X])^2] - (\mathbb{E}[Y])^2 \\ &= \mathbb{V}[Y].\end{aligned} \tag{B-16}$$

□

B.7 Sum of two random variables

Often the probability density function of the sum of two random variables needs to be calculated.

Theorem B.3 – Sum of two dependent random variables (based on [81, p. 91])

Suppose, that X, Y are two jointly continuous random variables with the joint probability density function $p(x, y)$. The density function $p(z)$ of the sum $Z = X + Y$ is

$$p(z) = \int_{-\infty}^{\infty} p(x, z - x)dx. \tag{B-27}$$

Proof:

The proof is omitted here and can be found in e.g. [81, p. 91].

□

In the special case that the two random variables X, Y are independent, equation (B-27) reduces to

$$p(z) = \int_{-\infty}^{\infty} p(x)p(z - x)dx, \tag{B-28}$$

which resembles a convolution $p(x) * p(y)$ and is also known as the Convolution formula. The corresponding proof can be found in e.g. [81, p. 91].

B.8 Moment Generating Function

If a rv has a *moment generating function* (mgf), the latter can be used to directly calculate any required moment of the underlying distribution. However, the existence of the moments of a rv not necessarily implies the existence of a mgf. A mgf is defined as follows.

Definition B.20 – Moment Generating Function ([51, p. 85], Definition 5.3)

Let X be a real-valued random variable. The *Moment Generating Function* of X is defined as

$$M_x(s) = \mathbb{E}_x[e^{sX}] \tag{B-29}$$

whenever the expectation is finite.

Mgfs are not only useful to calculate the moments of a rv, they also often simplify the calculation of the sum of several random variables. This thesis predominantly uses them to simplify an affine transformation of a GMM distributed rv.

B.9 Kullback-Leibler Divergence and Overlapping Coefficient

In order to compare results analytically, a metric needs to be used. Several measures exist, which help to to assess the difference between two distributions. This work uses the Kullback-Leibler divergence ([127], [126]) and the Overlapping Coefficient ([99]). Alternative measures include i.a. the negative log likelihood ([21]), the integrated quadratic distance ([207]) or the Akaike information criterion ([2], [32], [26]).

Definition B.21 – Kullback-Leibler Divergence ([127], [126])

Let X be a random variable and let $p(x)$ and $q(x)$ denote two pdfs of X. Then the Kullback-Leibler Divergence is defined as

$$D_{KL}(p||q) = \int_X p(x) log \frac{p(x)}{q(x)} dx \tag{B-30}$$

The Kullback-Leibler Divergence evaluates to zero if and only if $p(x) = q(x)$. Otherwise, it satisfies $D_{KL}(p||q) > 0$. Note that the Kullback-Leibler divergence is not symmetric. This means $D_{KL}(p||q) \neq D_{KL}(q||p)$.

Definition B.22 – Overlapping Coefficient (based on [99])

Let X be a random variable and let $p(x)$ and $q(x)$ denote two pdfs of X defined on $\mathbb{R}$. In the continuous case, the Overlapping Coefficient is defined as

$$\mathrm{OVL}(p, q) = \int_{\mathbb{R}} \min[p(x), q(x)]\, dx. \tag{B-31}$$

Similarly, in the discrete case, the Overlapping Coefficient is defined as

$$\mathrm{OVL}(p, q) = \sum_X \min[p(x), q(x)]. \tag{B-32}$$

B.10 Further important properties

Mean of quadratic forms

In order to propagate a normally distributed rv through a GP, the mean of a special quadratic form needs to be calculated. The calculation rule is presented in the following.

Theorem B.4 – Mean of Quadratic Forms ([193, p. 9], Theorem 1.5)

Let $\boldsymbol{X} \in \mathbb{R}^n$ be a vector of random variables and let $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ be a symmetric matrix. If $\mathbb{E}_x[\boldsymbol{X}] = \boldsymbol{\mu}$ and $\mathbb{V}_x[\boldsymbol{X}] = \boldsymbol{\Sigma}$ then

$$\mathbb{E}_x[\boldsymbol{X}^T \boldsymbol{A} \boldsymbol{X}] = \mathrm{Tr}(\boldsymbol{A}\boldsymbol{\Sigma}) + \boldsymbol{\mu}^T \boldsymbol{A} \boldsymbol{\mu}. \tag{B-33}$$

Proof:

Making use of the properties of the trace operator from Appendix A.1, the following expressions are equal:

$$\begin{aligned}\mathbb{E}_x[\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X}] &= \mathrm{Tr}(\mathbb{E}_x[\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X}]) = \mathbb{E}_x[\mathrm{Tr}(\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X})] = \mathbb{E}_x[\mathrm{Tr}(\boldsymbol{A}\boldsymbol{X}\boldsymbol{X}^T)] = \mathrm{Tr}(\mathbb{E}_x[\boldsymbol{A}\boldsymbol{X}\boldsymbol{X}^T]) \\ &= \mathrm{Tr}(\boldsymbol{A}\mathbb{E}_x[\boldsymbol{X}\boldsymbol{X}^T]) = \mathrm{Tr}\big(\boldsymbol{A}(\mathbb{V}_x[\boldsymbol{X}] + \boldsymbol{\mu}\boldsymbol{\mu}^T)\big) = \mathrm{Tr}(\boldsymbol{A}\boldsymbol{\Sigma}) + \mathrm{Tr}(\boldsymbol{A}\boldsymbol{\mu}\boldsymbol{\mu}^T) = \mathrm{Tr}(\boldsymbol{A}\boldsymbol{\Sigma}) + \boldsymbol{\mu}^T\boldsymbol{A}\boldsymbol{\mu}.\end{aligned} \tag{B-34}$$

□

C The Gaussian probability distribution

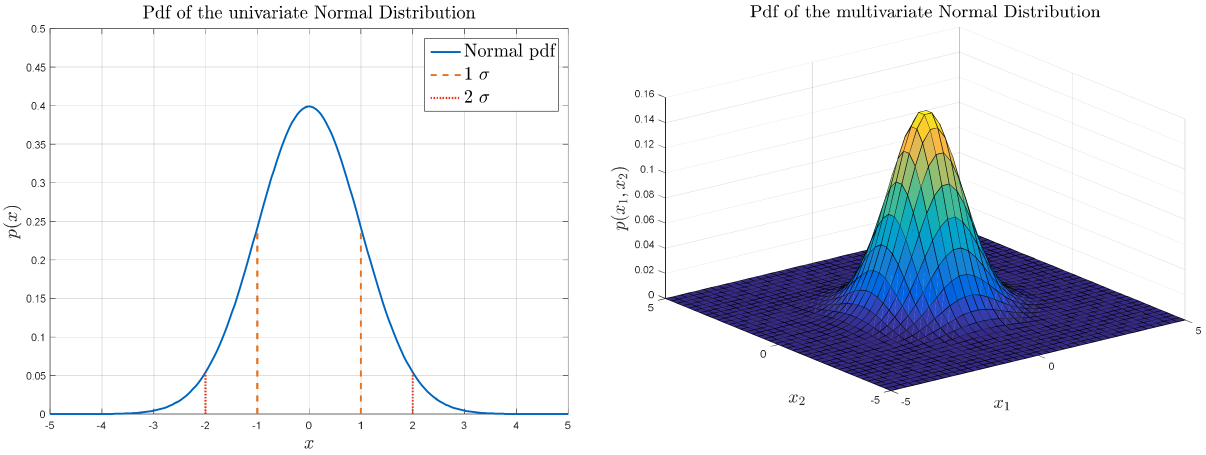


Figure C-1: Pdf of the Univariate and Multivariate normal distribution

The probably most famous and most widely used distribution is the so called normal distribution, also often called Gaussian distribution or simply Gaussian. The latter plays a vital role in this thesis. It's key properties are introduced in the following.

Definition C.1 – Normal Distribution ([52, p. 62], Definition 1.35)

A random variable X is said to have a *normal distribution* with parameters μ and σ^2 if it has the density

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, -\infty \leq x \leq \infty, \tag{C-1}$$

where μ can be any real number and $\sigma > 0$. We write $X \sim N(x|\mu, \sigma^2)$. If $X \sim N(x|0,1)$, we call it a *standard normal variable.*

The normal distribution is. Often, the rv X is not a scalar but a vector. The direct extension

Definition C.2 – Multivariate Normal Distribution (based on [52, p. 62], Definition 1.35)

A random vector $\boldsymbol{X} = [X_1 \quad \dots \quad X_k], \boldsymbol{X} \in \mathbb{R}^k$, is said to have a *multivariate normal distribution* with parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ if it has the density

$$p(\boldsymbol{x}) = (2\pi)^{-\frac{k}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}, \tag{C-2}$$

where $\boldsymbol{\mu}$ can be any real number and $\boldsymbol{\Sigma}$ is positive definite. We $\boldsymbol{X} \sim N(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Figure C-1 shows the univariate (left) and the multivariate (right) normal distribution.

C.1 Expectation and Variances

Expected Value of a random variable with Gaussian pdf

The expected value of a random variable x with a Gaussian pdf such that $x \sim N(\mu, \sigma^2)$ is defined as

$$\mathbb{E}_x[X] = \int_{-\infty}^{\infty} x p(x) dx = \int_{-\infty}^{\infty} x \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx. \tag{C-3}$$

Substituting $z = x - \mu$ and noting that $dz = dx$ yields

$$\mathbb{E}_x[X] = \int_{-\infty}^{\infty} (z+\mu) \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz = \underbrace{\int_{-\infty}^{\infty} z \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz}_{I_1} + \underbrace{\int_{-\infty}^{\infty} \mu \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz.}_{I_2} \tag{C-4}$$

Expanding the first integral yields

$$I_1 = \int_{-\infty}^{\infty} z \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz = \int_{-\infty}^{0} z \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz + \int_{0}^{\infty} z \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz. \tag{C-5}$$

Swapping integration limits and using integral rules yields

$$\begin{aligned} I_1 &= -\int_{0}^{-\infty} z \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz + \int_{0}^{\infty} z \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz \\ &= \int_{0}^{\infty} (-z) \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(-z)^2}{2\sigma^2}} dz + \int_{0}^{\infty} z \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz \\ &= -\int_{0}^{\infty} z \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz + \int_{0}^{\infty} z \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz = 0. \end{aligned} \tag{C-6}$$

Solving the second integral yields

$$I_2 = \int_{-\infty}^{\infty} \mu \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz = \mu \underbrace{\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{z^2}{2\sigma^2}} dz}_{1} = \mu. \tag{C-7}$$

Hence, the expected value is given by

$$\mathbb{E}_x[X] = \mu. \tag{C-8}$$

Variance of a random variable with Gaussian pdf

The variance of the random variable x is defined as

$$\mathbb{V}_x[X] = \mathbb{E}_x\left[\left(X - \mathbb{E}_x(X)\right)^2\right]. \tag{C-9}$$

Inserting the definition of the expected value yields

$$\mathbb{V}_x[X] = \int_{-\infty}^{\infty} (x-\mu)^2 \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx. \tag{C-10}$$

Similar to the derviation of the expected value, define $z = \frac{x-\mu}{\sigma\sqrt{2\pi}}$. Note that $\sigma\sqrt{2\pi}dz = dx$. Hence,

$$\mathbb{V}_x[X] = \int_{-\infty}^{\infty} \sigma^2 2z^2 \frac{1}{\sqrt{2\pi\sigma^2}} e^{-z^2} \sigma\sqrt{2\pi} dz = \frac{2\sigma^2}{\sqrt{\pi}} \int_{-\infty}^{\infty} z^2 e^{-z^2} dz. \tag{C-11}$$

Integration by parts with $u = z$ and $v = -\frac{1}{2}e^{-z^2}$ and noting that $v' = ze^{-z^2}$ yields

$$\mathbb{V}_x[X] = \frac{2\sigma^2}{\sqrt{\pi}}\left(-\frac{1}{2}ze^{-z^2}\Big|_{-\infty}^{\infty} - \int_{-\infty}^{\infty} -\frac{1}{2}e^{-z^2} dz\right). \tag{C-12}$$

Using L'Hopital's rule, the first term becomes zero and the equation reduces to

$$\mathbb{V}_x[X] = \frac{2\sigma^2}{\sqrt{\pi}} \int_{-\infty}^{\infty} \frac{1}{2} e^{-z^2} dz. \tag{C-13}$$

By using the Euler-Poisson integral $\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}$, the variance is

$$\mathbb{V}_x[X] = \frac{2\sigma^2}{\sqrt{\pi}} \frac{1}{2} \sqrt{\pi} = \sigma^2. \tag{C-14}$$

C.2 Moment Generating Function of a Gaussian random variable

Let X denote a rv with a Gaussian probability distribution such that $X \sim N(x|\mu, \sigma^2)$. The mgf of X is

$$M_x(s) = e^{s\mu + \frac{1}{2}\sigma^2 s^2}. \tag{C-15}$$

Similarly, for a random vector $\boldsymbol{X} \in \mathbb{R}^n$ with multivariate normal probability distribution $\boldsymbol{X} \sim N(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ the corresponding mgf is

$$M_x(s) = e^{\boldsymbol{s}^T\left(\boldsymbol{\mu} + \frac{1}{2}\boldsymbol{\Sigma}\boldsymbol{s}\right)} \tag{C-16}$$

In order to extract the n-th moment, M_x needs to be differentiated n times with respect to s and evaluated at $s = 0$. Using its mgf, the mean of the rv $\boldsymbol{X}$ is

$$\mathbb{E}_x[\boldsymbol{X}] = \left.\frac{dM_x}{d\boldsymbol{s}}\right|_{\boldsymbol{s}=0} = \left[\left(\boldsymbol{\mu} + \frac{1}{2}\boldsymbol{\Sigma s} + \frac{1}{2}\boldsymbol{s}^T\boldsymbol{\Sigma}\right) e^{\boldsymbol{s}^T\left(\boldsymbol{\mu}+\frac{1}{2}\boldsymbol{\Sigma s}\right)}\right]_{\boldsymbol{s}=0} = \boldsymbol{\mu} \tag{C-17}$$

Similarly, its covariance is

$$\begin{aligned}\mathbb{V}_x[\boldsymbol{X}] = \mathbb{E}_x[(\boldsymbol{X}\boldsymbol{X}^T)] - \mathbb{E}_x[\boldsymbol{X}]\mathbb{E}_x[\boldsymbol{X}^T] &= \left.\frac{d^2M_x}{d\boldsymbol{s}^2}\right|_{\boldsymbol{s}=0} - \left.\frac{dM_x}{d\boldsymbol{s}}\right|_{\boldsymbol{s}=0}\left(\left.\frac{dM_x}{d\boldsymbol{s}}\right|_{\boldsymbol{s}=0}\right)^T \\ &= \Bigg[\Bigg(\boldsymbol{\mu}\left(\boldsymbol{\mu}^T + \frac{1}{2}\boldsymbol{\Sigma s} + \boldsymbol{s}^T\boldsymbol{\Sigma}\right) + \frac{1}{2}\boldsymbol{\Sigma} + \frac{1}{2}\boldsymbol{\Sigma} + \frac{1}{2}\boldsymbol{\Sigma s}\left(\boldsymbol{\mu} + \frac{1}{2}\boldsymbol{\Sigma s} + \frac{1}{2}\boldsymbol{s}^T\boldsymbol{\Sigma}\right) \\ &\quad + \frac{1}{2}\boldsymbol{s}^T\boldsymbol{\Sigma}\left(\boldsymbol{\mu} + \frac{1}{2}\boldsymbol{\Sigma s} + \frac{1}{2}\boldsymbol{s}^T\boldsymbol{\Sigma}\right)\Bigg) e^{\boldsymbol{s}^T\left(\boldsymbol{\mu}+\frac{1}{2}\boldsymbol{\Sigma s}\right)}\Bigg]_{\boldsymbol{s}=0} - \boldsymbol{\mu}\boldsymbol{\mu}^T = \boldsymbol{\Sigma}.\end{aligned} \tag{C-18}$$

C.3 Calculation rules for joint, conditional and marginal normal pdfs

This section summarizes required calculation rules for multivariate normal distributions.

Lemma C.1 – Joint Distribution of Gaussian variables (based on [184, 209, Lemma A.1]

If two random variables $\boldsymbol{X} \in \mathbb{R}^n, \boldsymbol{Y} \in \mathbb{R}^m$ have the Gaussian probability distributions $p(\boldsymbol{x}) = N(\boldsymbol{m}, \boldsymbol{P})$ and $p(\boldsymbol{y}|\boldsymbol{x}) = N(\boldsymbol{Hx}, \boldsymbol{R})$, then the joint distribution $p(\boldsymbol{x}, \boldsymbol{y})$ is

$$\begin{pmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{pmatrix} \sim N\left(\begin{pmatrix}\boldsymbol{m}\\ \boldsymbol{Hm}\end{pmatrix}, \begin{pmatrix}\boldsymbol{P} & \boldsymbol{PH}^T\\ \boldsymbol{HP} & \boldsymbol{HPH}^T + \boldsymbol{R}\end{pmatrix}\right). \tag{C-19}$$

Proof:

The joint probability density function is defined as

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x}) = N(\boldsymbol{Hx}, \boldsymbol{R})N(\boldsymbol{m}, \boldsymbol{P}). \tag{C-20}$$

Note that the Gaussian pdf only depends on the two parameters mean and variance. As a consequence, the equality sign can be switched for a proportionality operator, which leads to

$$\begin{aligned}p(\boldsymbol{x}, \boldsymbol{y}) &\propto e^{-\left[(\boldsymbol{y}-\boldsymbol{Hx})^T\boldsymbol{R}^{-1}(\boldsymbol{y}-\boldsymbol{Hx})\right]}e^{-\left[(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{P}^{-1}(\boldsymbol{x}-\boldsymbol{m})\right]} = \\ &\propto e^{-\left[(\boldsymbol{y}-\boldsymbol{Hx})^T\boldsymbol{R}^{-1}(\boldsymbol{y}-\boldsymbol{Hx})+(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{P}^{-1}(\boldsymbol{x}-\boldsymbol{m})\right]}.\end{aligned} \tag{C-21}$$

Transferring the result into the notation for matrix-vector multiplication yields

$$\begin{aligned}p(\boldsymbol{x}, \boldsymbol{y}) &\propto e^{-\left[(\boldsymbol{y}-\boldsymbol{Hx})^T\boldsymbol{R}^{-1}(\boldsymbol{y}-\boldsymbol{Hx})+(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{P}^{-1}(\boldsymbol{x}-\boldsymbol{m})\right]} = \\ &= \exp\left(-\begin{bmatrix}(\boldsymbol{x}-\boldsymbol{m})\\ (\boldsymbol{y}-\boldsymbol{Hm})^T\end{bmatrix}^T \underbrace{\begin{bmatrix}\boldsymbol{P}^{-1} + \boldsymbol{H}^T\boldsymbol{R}^{-1}\boldsymbol{H} & -\boldsymbol{H}^T\boldsymbol{R}^{-1}\\ -\boldsymbol{R}^{-1}\boldsymbol{H} & \boldsymbol{R}^{-1}\end{bmatrix}}_{\boldsymbol{\Sigma}^{-1}} \begin{bmatrix}(\boldsymbol{x}-\boldsymbol{m})\\ (\boldsymbol{y}-\boldsymbol{Hm})^T\end{bmatrix}\right).\end{aligned} \tag{C-22}$$

Taking the inverse of the matrix $\boldsymbol{\Sigma}^{-1}$ yields

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{P}^{-1} + \boldsymbol{H}^T\boldsymbol{R}^{-1}\boldsymbol{H} & -\boldsymbol{H}^T\boldsymbol{R}^{-1} \\ -\boldsymbol{R}^{-1}\boldsymbol{H} & \boldsymbol{R}^{-1} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{P} & \boldsymbol{P}\boldsymbol{H}^T \\ \boldsymbol{H}\boldsymbol{P} & \boldsymbol{H}\boldsymbol{P}\boldsymbol{H}^T + \boldsymbol{R} \end{bmatrix}. \qquad \textbf{(C-23)}$$

□

Lemma C.2 – Calculation of conditional and marginal pdfs from a joint pdf (based on [184, 209, Lemma A.2]

Consider two random variables $\boldsymbol{X} \in \mathbb{R}^n, \boldsymbol{Y} \in \mathbb{R}^m$ with the joint Gaussian probability distribution

$$\begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{pmatrix} \sim N\left(\begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix}\right). \qquad \textbf{(C-24)}$$

Then the conditional and marginal pdfs are

$$\begin{gathered} p(\boldsymbol{x}) = N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}) \\ p(\boldsymbol{y}) \sim N(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22}) \\ p(\boldsymbol{x}|\boldsymbol{y}) \sim N(\boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{y} - \boldsymbol{\mu}_2),\ \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}) \\ p(\boldsymbol{y}|\boldsymbol{x}) \sim N(\boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_1),\ \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}). \end{gathered} \qquad \textbf{(C-25)}$$

Proof:

The results follow directly from applying the calculation rules in Appendix B.3 and solving the matrix-vector multiplications. The lengthy derivation is omitted here.

□

C.4 Affine transform of a normally distributed random variable

The affine transformation of a normally distributed rv. is a common procedure found in a multitude of application such as the Kalman Filter. It forms the basis for the GMM propagation used within this thesis.

Lemma C.3 – Affine transformation of a normally distributed random variable

Consider a continuous random vector $\boldsymbol{X} \in \mathbb{R}^n$ with a multivariate normal pdf $p_x(\boldsymbol{x}) = N(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Under the affine transformation $\boldsymbol{Z} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{B}$ the continuous rv. $\boldsymbol{Z}$ has a multivariate Gaussian pdf with $p_z(\boldsymbol{z}) = N(\boldsymbol{z}|\boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{B}, \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T)$.

Proof:

Note, that the definition of a pdf requires it to integrate to one:

$$\int p(\boldsymbol{x}) d\boldsymbol{x} = 1 \int p\big(\boldsymbol{A}^{-1}(\boldsymbol{z} - \boldsymbol{B})\big)\, d\boldsymbol{x} \qquad \textbf{(C-26)}$$

The affine transformation of the rv. is shown by substitution. Therefore, note that $\boldsymbol{X} = \boldsymbol{A}^{-1}(\boldsymbol{Z} - \boldsymbol{B})$ and $d\boldsymbol{X} = \boldsymbol{A}^{-1}d\boldsymbol{Z}$:

$$\int p(\boldsymbol{x})d\boldsymbol{x} = \int p\big(\boldsymbol{A}^{-1}(\boldsymbol{z} - \boldsymbol{B})\big)\boldsymbol{A}^{-1}\, d\boldsymbol{z}$$

$$\begin{aligned}
&= \int \frac{1}{(2\pi)^{\frac{n}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{A}^{-1}(\boldsymbol{z} - \boldsymbol{B}) - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{A}^{-1}(\boldsymbol{z} - \boldsymbol{B}) - \boldsymbol{\mu})\right] \boldsymbol{A}^{-1}d\boldsymbol{z} \\
&= \int \frac{1}{(2\pi)^{\frac{n}{2}}|\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{A}^{-1}(\boldsymbol{z} - \boldsymbol{B}) - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{A}^{-1}(\boldsymbol{z} - \boldsymbol{B}) - \boldsymbol{\mu})\right] d\boldsymbol{z} \\
&= \int \frac{1}{(2\pi)^{\frac{n}{2}}|\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{z} - \boldsymbol{B} - \boldsymbol{A}\boldsymbol{\mu})^T {\boldsymbol{A}^{-1}}^{T} \boldsymbol{\Sigma}^{-1}\boldsymbol{A}^{-1}(\boldsymbol{z} - \boldsymbol{B} - \boldsymbol{A}\boldsymbol{\mu})\right] d\boldsymbol{z} \\
&= \int \underbrace{\frac{1}{(2\pi)^{\frac{n}{2}}|\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{z} - \boldsymbol{B} - \boldsymbol{A}\boldsymbol{\mu})^T (\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T)^{-1}(\boldsymbol{z} - \boldsymbol{B} - \boldsymbol{A}\boldsymbol{\mu})\right]}_{p_Z(\boldsymbol{z}) = N(\boldsymbol{z}|\boldsymbol{A}\boldsymbol{\mu}+\boldsymbol{B},\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^T)} d\boldsymbol{z}.
\end{aligned} \tag{C-27}$$

□

The proof shown above only holds if $\boldsymbol{A}$ is invertible. A more general proof can be obtained by using moment generating functions.

C.5 Summation and Difference of two independent and normally distributed random variables

Finding the sum or difference of two multivariate, independent normally distributed random variables is a common task and explained in many standard references (see e.g. [10]).

Lemma C.4 - Sum and difference of two independent normal random variables

Consider two independent and continuous random variables $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^n$ with probability distributions $\boldsymbol{X} \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $\boldsymbol{Y} \sim N(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$. Then the pdf of the sum $\boldsymbol{Z}_1 = \boldsymbol{X} + \boldsymbol{Y}$ is $\boldsymbol{Z}_1 \sim N(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2)$. Similarly, the pdf of the difference $\boldsymbol{Z}_2 = \boldsymbol{X} - \boldsymbol{Y}$ is $\boldsymbol{Z}_2 \sim N(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2)$.

Proof:

The proof is omitted here. It follows by solving the convolution in (B-28) in Appendix B.7.

□

C.6 Sum of two squared forms

Within this thesis, often the multiplication of two normal pdfs $N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $N(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ is performed. In particular, the goal is to find the mean $\boldsymbol{\mu}_3$ and variance $\boldsymbol{\Sigma}_3$ in

$$N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) N(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2) = cN(\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3). \tag{C-28}$$

Inserting the equations for multivariate normal pdfs yields

$$cN(\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3) = (2\pi)^{-\frac{k}{2}} |\boldsymbol{\Sigma}_1|^{-\frac{1}{2}} e^{-\frac{1}{2}(x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1)} (2\pi)^{-\frac{k}{2}} |\boldsymbol{\Sigma}_2|^{-\frac{1}{2}} e^{-\frac{1}{2}(x-\mu_2)^T \Sigma_2^{-1}(x-\mu_2)} \tag{C-29}$$

Solving this multiplication requires the solution of a sum of the two squared forms

$$\begin{aligned} -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}_1^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_2)^T \boldsymbol{\Sigma}_2^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2) \\ = -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_3)^T \boldsymbol{\Sigma}_3^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_3) + C. \end{aligned} \tag{C-30}$$

The solution of the sum of two squared forms is omitted here. It involves the expansion of the squared forms, the rearrangement of terms and rearranging the required squared forms at the end. The solution is ([167])

$$\begin{aligned} \boldsymbol{\mu}_3 &= (\boldsymbol{\Sigma}_1^{-1} + \boldsymbol{\Sigma}_2^{-1})^{-1} (\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_2^{-1}\boldsymbol{\mu}_2) \\ \boldsymbol{\Sigma}_3 &= (\boldsymbol{\Sigma}_1^{-1} + \boldsymbol{\Sigma}_2^{-1})^{-1} \\ c &= \frac{1}{|(2\pi)(\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2)|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mu_1-\mu_2)^T (\Sigma_1+\Sigma_2)^{-1}(\mu_1-\mu_2)}. \end{aligned} \tag{C-31}$$

D Fundamentals of Lyapunov Stability Theory

D.1 Stability of Nonlinear Systems

Consider the following class of nonlinear, autonomous dynamical systems

$$\dot{\boldsymbol{x}} = \boldsymbol{f}\big(\boldsymbol{x}(t)\big). \tag{D-1}$$

In the following various stability properties for the nonlinear system in (D-1) are defined.

Definition D.1 – Stability of Nonlinear Systems ([84, p. 136], Definition 3.1)

a) The zero solution $\boldsymbol{x}(t) \equiv \boldsymbol{0}$ to equation (D-1) is *Lyapunov stable* if, for all $\varepsilon > 0$, there exists $\delta = \delta(\varepsilon) > 0$ such that if $\|\boldsymbol{x}(0)\| < \delta$, then $\|\boldsymbol{x}(t)\| < \varepsilon$, $t \geq 0$.

b) The zero solution $\boldsymbol{x}(t) \equiv \boldsymbol{0}$ to equation (D-1) is *(locally) asymptotically stable* if it is Lyapunov stable and there exists $\delta > 0$ such that if $\|\boldsymbol{x}(0)\| < \delta$, then $\lim_{t\to\infty} \boldsymbol{x}(t) = \boldsymbol{0}$.

c) The zero solution $\boldsymbol{x}(t) \equiv \boldsymbol{0}$ to equation (D-1) is *(locally) exponentially stable* if there exist positive constants α, β and δ such that if $\|\boldsymbol{x}(0)\| < \delta$, then $\|\boldsymbol{x}(t)\| \leq \alpha\|\boldsymbol{x}(0)\|e^{-\beta t}, t \geq 0$.

d) The zero solution $\boldsymbol{x}(t) \equiv \boldsymbol{0}$ to equation (D-1) is *globally asymptotically stable* if it is Lyapunov stable and for all $\boldsymbol{x}(0) \in \mathbb{R}^n$, $\lim_{t\to\infty} \boldsymbol{x}(t) = \boldsymbol{0}$.

e) The zero solution $\boldsymbol{x}(t) \equiv \boldsymbol{0}$ to equation (D-1) is *globally exponentially stable* if there exist positive constants α and β such that $\|\boldsymbol{x}(t)\| \leq \alpha\|\boldsymbol{x}(0)\|e^{-\beta t}, t \geq 0$, for all $\boldsymbol{x}(0) \in \mathbb{R}^n$.

The zero solution $\boldsymbol{x}(t) \equiv \boldsymbol{0}$ to equation (D-1) is *unstable* if it is not Lyapunov stable.

D.2 Lyapunov's Direct Method

Lyapunov's direct method provides sufficient conditions for Lyapunov, asymptotic and exponential stability of a nonlinear system.

Theorem D.1 – Lyapunov's Direct Method ([84, p. 137], Theorem 3.1)

Consider the nonlinear dynamical system (D-1) and assume that there exists a continuously differentiable function $V\colon D \to \mathbb{R}$ such that

$$\begin{gathered} V(0) = 0, \\ V(\boldsymbol{x}) > 0, \quad \boldsymbol{x} \in D, \quad \boldsymbol{x} \neq \boldsymbol{0}, \\ V'(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}) \leq 0, \quad \boldsymbol{x} \in D. \end{gathered} \tag{D-2}$$

Then the zero solution $\boldsymbol{x}(t) \equiv \boldsymbol{0}$ to equation (D-1) is Lyapunov stable. If, in addition,

$$V'(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}) < 0, \quad \boldsymbol{x} \in D, \quad \boldsymbol{x} \neq \boldsymbol{0}, \tag{D-3}$$

then the zero solution $\boldsymbol{x}(t) \equiv \boldsymbol{0}$ to equation (D-1) is asymptotically stable.

Finally, if there exist scalars $\alpha, \beta, \varepsilon > 0$, and $p \geq 1$, such that $V: D \to \mathbb{R}$ satisfies

$$\begin{aligned} \alpha\|\boldsymbol{x}\|^p \leq V(\boldsymbol{x}) \leq \beta\|\boldsymbol{x}\|^p, \quad \boldsymbol{x} \in D, \\ V'(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}) \leq -\varepsilon V(\boldsymbol{x}), \quad \boldsymbol{x} \in D, \end{aligned} \tag{D-4}$$

then the zero solution $\boldsymbol{x}(t) \equiv \boldsymbol{0}$ to equation (D-1) is exponentially stable.

Proof:

The proof is omitted here. For a full derivation, see i.a. [84, pp. 137-141].

□

E Filter Equations

This section introduces the equations for the OFPS, which is used in section 4.4.3.1. The OFPS consists of a forward and a backward Kalman Filter. In the following, first the equations for the Kalman Filter are derived, followed by the OFPS.

E.1 Kalman Filter

This section introduces the well-known Kalman Filter ([115], [144]) and derives the propagation and update step from a Bayesian perspective. Consider a Markovian model of the form

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{q}_k, \quad \boldsymbol{q}_k \sim N(0,\boldsymbol{Q}) \\ \boldsymbol{y}_k &= \boldsymbol{H}_k\boldsymbol{x}_k + \boldsymbol{r}_k, \quad \boldsymbol{r}_k \sim N(0,\boldsymbol{R}) \end{aligned} \tag{E-1}$$

Assume $\boldsymbol{x}_k$ to be modeled by a prior distribution $p(\boldsymbol{x}_k|\boldsymbol{y}_k) = N_{\boldsymbol{x}_k}(\boldsymbol{m}_k, \boldsymbol{P}_k)$. The Kalman-Filter is always seen to be comprised of two steps: the prediction step and the update step. The former predicts the state distribution $p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_k)$ based on the model in (E-1). The latter corrects the prediction by incoming measurements. Employing the Chapman-Kolmogorov Equation in order to predict the state $\boldsymbol{x}_{k+1}$ yields

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_k) = \int p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{y}_k)d\boldsymbol{x}_k. \tag{E-2}$$

The distribution $p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)$ is described by the system dynamics in (E-1). Furthermore, the prior $p(\boldsymbol{x}_k|\boldsymbol{y}_k)$ is assumed to be known. Inserting both into (E-2) yields

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_k) = \int N_{\boldsymbol{x}_{k+1}}(\boldsymbol{A}_k\boldsymbol{x}_k, \boldsymbol{Q})N_{\boldsymbol{x}_k}(\boldsymbol{m}_k, \boldsymbol{P}_k)d\boldsymbol{x}_k \tag{E-3}$$

Building the joint distribution by using the calculation rules in Appendix C yields

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_k) = \int N_{\binom{x_k}{x_{k+1}}}\left(\begin{bmatrix}\boldsymbol{m}_k \\ \boldsymbol{A}_k\boldsymbol{m}_k\end{bmatrix}, \begin{bmatrix}\boldsymbol{P}_k & \boldsymbol{P}_k\boldsymbol{A}_k^T \\ \boldsymbol{A}_k\boldsymbol{P}_k & \boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T + \boldsymbol{Q}\end{bmatrix}\right)d\boldsymbol{x}_k. \tag{E-4}$$

Note that $p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_k)$ does not depend on $\boldsymbol{x}_k$. Hence, it can be moved out of the integral. Using the calculation rules in Appendix C, the predicted distribution is

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_k) = N_{\boldsymbol{x}_{k+1}}\left(\underbrace{\boldsymbol{A}_k\boldsymbol{m}_k}_{\boldsymbol{m}_{k+1}^-}, \underbrace{\boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T + \boldsymbol{Q}}_{\boldsymbol{P}_{k+1}^-}\right). \tag{E-5}$$

The explicit prediction equations are

$$\begin{aligned} \boldsymbol{m}_{k+1}^- &= \boldsymbol{A}_k\boldsymbol{m}_k \\ \boldsymbol{P}_{k+1}^- &= \boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T + \boldsymbol{Q}. \end{aligned} \tag{E-6}$$

The update step aims at calculating $p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k+1})$. Therefore, consider the definition of conditional probability:

$$p(\boldsymbol{x}_{k+1}, \boldsymbol{y}_{k+1}|\boldsymbol{y}_k) = p(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1})p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_k). \tag{E-7}$$

The dependency on $\boldsymbol{y}_k$ is neglected since it is fully described by the dependence on $\boldsymbol{x}_{k+1}$. Inserting the result of the update step in (E-5) as well as the model of (E-1) into (E-7) and again using the calculation rules in Appendix C yields

$$p(\boldsymbol{x}_{k+1}, \boldsymbol{y}_{k+1}|\boldsymbol{y}_k) = N_{\boldsymbol{y}_{k+1}}(\boldsymbol{H}_{k+1}\boldsymbol{x}_{k+1}, \boldsymbol{R})N_{\boldsymbol{x}_{k+1}}(\boldsymbol{m}_{k+1}^-, \boldsymbol{P}_{k+1}^-) =$$
$$N_{\binom{x_{k+1}}{y_{k+1}}}\left(\begin{bmatrix}\boldsymbol{m}_{k+1}^- \\ \boldsymbol{H}_{k+1}\boldsymbol{m}_{k+1}^-\end{bmatrix}, \begin{bmatrix}\boldsymbol{P}_{k+1}^- & \boldsymbol{P}_{k+1}^-\boldsymbol{H}_{k+1}^T \\ \boldsymbol{H}_{k+1}\boldsymbol{P}_{k+1}^- & \boldsymbol{H}_{k+1}\boldsymbol{P}_{k+1}^-\boldsymbol{H}_{k+1}^T + \boldsymbol{R}\end{bmatrix}\right). \tag{E-8}$$

Employing the calculation rules for conditional probabilities from Appendix C results in

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k+1}) =$$
$$N_{\boldsymbol{x}_{k+1}}\begin{pmatrix}\boldsymbol{m}_{k+1}^- + \boldsymbol{P}_{k+1}^-\boldsymbol{H}_{k+1}^T\left[\boldsymbol{H}_{k+1}\boldsymbol{P}_{k+1}^-\boldsymbol{H}_{k+1}^T + \boldsymbol{R}\right]^{-1}(\boldsymbol{y}_{k+1} - \boldsymbol{H}_{k+1}\boldsymbol{m}_{k+1}^-), \\ \boldsymbol{P}_{k+1}^- - \boldsymbol{P}_{k+1}^-\boldsymbol{H}_{k+1}^T\left[\boldsymbol{H}_{k+1}\boldsymbol{P}_{k+1}^-\boldsymbol{H}_{k+1}^T + \boldsymbol{R}\right]^{-1}\boldsymbol{H}_{k+1}\boldsymbol{P}_{k+1}^-\end{pmatrix}. \tag{E-9}$$

Using the simplifications

$$\begin{aligned}\boldsymbol{S}_{k+1} &= \boldsymbol{H}_{k+1}\boldsymbol{P}_{k+1}^-\boldsymbol{H}_{k+1}^T + \boldsymbol{R} \\ \boldsymbol{K}_{k+1} &= \boldsymbol{P}_{k+1}^-\boldsymbol{H}_{k+1}^T\boldsymbol{S}_{k+1}^{-1},\end{aligned} \tag{E-10}$$

equation (E-9) becomes

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k+1}) =$$
$$N_{\boldsymbol{x}_{k+1}}\left(\underbrace{\boldsymbol{m}_{k+1}^- + \boldsymbol{K}_{k+1}(\boldsymbol{y}_{k+1} - \boldsymbol{H}_{k+1}\boldsymbol{m}_{k+1}^-)}_{\boldsymbol{m}_{k+1}}, \underbrace{\boldsymbol{P}_{k+1}^- - \boldsymbol{K}_{k+1}\boldsymbol{S}_{k+1}\boldsymbol{K}_{k+1}^T}_{\boldsymbol{P}_{k+1}}\right). \tag{E-11}$$

The explicit update equations are

$$\begin{aligned}\boldsymbol{m}_{k+1} &= \boldsymbol{m}_{k+1}^- + \boldsymbol{K}_{k+1}(\boldsymbol{y}_{k+1} - \boldsymbol{H}_{k+1}\boldsymbol{m}_{k+1}^-) \\ \boldsymbol{P}_{k+1} &= \boldsymbol{P}_{k+1}^- - \boldsymbol{K}_{k+1}\boldsymbol{S}_{k+1}\boldsymbol{K}_{k+1}^T.\end{aligned} \tag{E-12}$$

E.2 Optimal Fixed Point Smoother

This section introduces the Optimal Fixed Point Smoother, also called Rauch –Tung-Striebel smoother ([70], [144]). The following derivation is based on [184]. Assume that the state estimate $\boldsymbol{x}_k$ at the time instant k is sought. The general idea of a fixed point smoother is to use the KF approach in section E.1 together with future measurements in order to arrive at an improved state estimate. In practice, the OFPS consists of a Forward KF up until time step k and a Backward Kalman Filter from a future time instant T, with $T > k$, to k. Naturally, the estimate of the OFPS is delayed, as $\boldsymbol{x}_T$ needs to be available in order to arrive at a smoothed estimate of $\boldsymbol{x}_k$. Figure E- shows an exemplary timeline of the optimal fixed point smoother. Here, the Backward KF operates on four time steps. Also note, that the forward KF is operational up to T. Its estimate of $\boldsymbol{x}_T$ is used in order to initialize the backward KF.

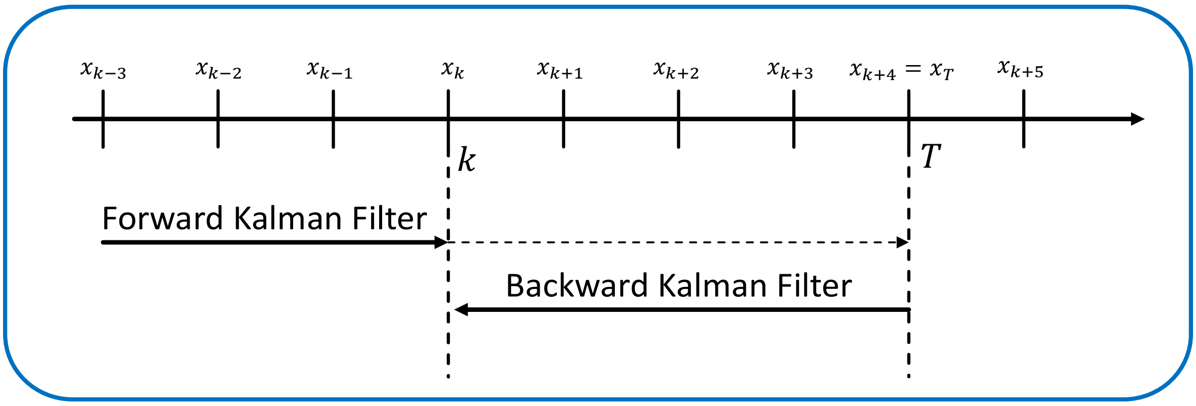


Figure E-1: Timeline of an Optimal Fixed Point Smoother

In order to derive the solution of the fixed point smoother, consider the system dynamics in (E-1) and the estimate of the forward KF in (E-11). In essence, the backward KF seeks the distribution of $\boldsymbol{x}_k$ given the data from k to T, i.e. $p(\boldsymbol{x}_k|\boldsymbol{y}_{k:T})$. In order to derive the backward iterated calculation laws, consider the joint probability distribution $p(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{y}_k)$, given by

$$p(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{y}_k) = p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{y}_k). \tag{E-13}$$

The latter is already calculated in (E-4) and given by

$$p(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{y}_k) = N_{\left(\begin{smallmatrix}\boldsymbol{x}_k \\ \boldsymbol{x}_{k+1}\end{smallmatrix}\right)}\left(\begin{bmatrix}\boldsymbol{m}_k \\ \boldsymbol{A}_k\boldsymbol{m}_k\end{bmatrix}, \begin{bmatrix}\boldsymbol{P}_k & \boldsymbol{P}_k\boldsymbol{A}_k^T \\ \boldsymbol{A}_k\boldsymbol{P}_k & \boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T + \boldsymbol{Q}\end{bmatrix}\right) \tag{E-14}$$

In order to yield a backward iteration, the distribution $p(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{y}_k)$ is conditioned on $\boldsymbol{x}_{k+1}$:

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k+1}, \boldsymbol{y}_k) = N_{\boldsymbol{x}_k}\left(\boldsymbol{m}_k^{k+1}, \boldsymbol{P}_k^{k+1}\right) \tag{E-15}$$

Using the calculation rules for conditional distributions in Appendix C.3 the conditional mean is

$$\boldsymbol{m}_k^{k+1} = \boldsymbol{m}_k + \boldsymbol{P}_k\boldsymbol{A}_k^T\left(\boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T + \boldsymbol{Q}\right)^{-1}(\boldsymbol{x}_{k+1} - \boldsymbol{A}_k\boldsymbol{m}_k). \tag{E-16}$$

Similarly, the conditional covariance is

$$\boldsymbol{P}_k^{k+1} = \boldsymbol{P}_k - \boldsymbol{P}_k\boldsymbol{A}_k^T\left(\boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T + \boldsymbol{Q}\right)^{-1}\boldsymbol{A}_k\boldsymbol{P}_k. \tag{E-17}$$

For simplicity, define

$$\boldsymbol{G}_k = \boldsymbol{P}_k\boldsymbol{A}_k^T\left(\boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T + \boldsymbol{Q}\right)^{-1}. \tag{E-18}$$

Then (E-15) becomes

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k+1}, \boldsymbol{y}_k) = N_{\boldsymbol{x}_k}\left(\boldsymbol{m}_k + \boldsymbol{G}_k(\boldsymbol{x}_{k+1} - \boldsymbol{A}_k\boldsymbol{m}_k), \boldsymbol{P}_k - \boldsymbol{G}_k\left(\boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T + \boldsymbol{Q}\right)\boldsymbol{G}_k^T\right). \tag{E-19}$$

The system in (E-1) represents a Markov chain and therefore its states exhibit the Markov property. As a result, $p(\boldsymbol{x}_k|\boldsymbol{x}_{k+1}, \boldsymbol{y}_k)$ is only dependent on $\boldsymbol{y}_k$ and not on future measurements. Hence,

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k+1}, \boldsymbol{y}_{k:T}) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k+1}, \boldsymbol{y}_k). \tag{E-20}$$

To interpret the latter, note that the conditional distribution assumes $\boldsymbol{x}_{k+1}$ to be known. Since a Markov chain is only dependent on the current information and not a batch of states, all future information, such as $\boldsymbol{x}_{k+2}$ or $\boldsymbol{y}_{k+1}$ is already contained in $\boldsymbol{x}_{k+1}$. In the next step, the joint distribution $p(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{y}_k)$ in (E-13) is conditioned on $\boldsymbol{y}_{k:T}$ instead of $\boldsymbol{y}_k$ only:

$$p(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{y}_{k:T}) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k+1}, \boldsymbol{y}_{k:T})p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k:T}). \tag{E-21}$$

The first term on the right hand side is given by (E-19). Let the second term be given by $p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k:T}) = N_{\boldsymbol{x}_{k+1}}\left(\boldsymbol{m}_{s,k+1}, \boldsymbol{P}_{s,k+1}\right)$. Building the joint distribution by using the calculation rules in Appendix C yields

$$p(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{y}_{k:T}) =$$

$$N_{\left(\begin{smallmatrix}\boldsymbol{x}_{k+1}\\ \boldsymbol{x}_k\end{smallmatrix}\right)}\left(\begin{bmatrix}\boldsymbol{m}_{s,k+1}\\ \boldsymbol{m}_k + \boldsymbol{G}_k\left(\boldsymbol{m}_{s,k+1} - \boldsymbol{A}_k\boldsymbol{m}_k\right)\end{bmatrix}, \begin{bmatrix}\boldsymbol{P}_{s,k+1} & \boldsymbol{P}_{s,k+1}\boldsymbol{G}_k^T\\ \boldsymbol{G}_k\boldsymbol{P}_{s,k+1} & \boldsymbol{G}_k\boldsymbol{P}_{s,k+1}\boldsymbol{G}_k^T + \boldsymbol{P}_k^{k+1}\end{bmatrix}\right). \tag{E-22}$$

Marginalizing $p(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{y}_{k:T})$ over $\boldsymbol{x}_{k+1}$ yields

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{k:T}) = N_{\boldsymbol{x}_k}\left(\boldsymbol{m}_{s,k}, \boldsymbol{P}_{s,k}\right). \tag{E-23}$$

Using the calculation rules in Appendix C.3, the mean is

$$\boldsymbol{m}_{s,k} = \boldsymbol{m}_k + \boldsymbol{G}_k\left(\boldsymbol{m}_{s,k+1} - \boldsymbol{A}_k\boldsymbol{m}_k\right). \tag{E-24}$$

Similarly, the covariance is

$$\boldsymbol{P}_{s,k} = \boldsymbol{G}_k\boldsymbol{P}_{s,k+1}\boldsymbol{G}_k^T + \boldsymbol{P}_k^{k+1} = \boldsymbol{P}_k + \boldsymbol{G}_k\left(\boldsymbol{P}_{s,k+1} - \boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T - \boldsymbol{Q}\right)\boldsymbol{G}_k^T. \tag{E-25}$$

Equations (E-24) and (E-25) resemble the update equations for the backward iterated KF. The process is initiated by setting $\boldsymbol{m}_{s,T} = \boldsymbol{m}_T$ and $\boldsymbol{P}_{s,T} = \boldsymbol{P}_T$, where $\boldsymbol{m}_T$ and $\boldsymbol{P}_T$ resemble the solution (E-12) of the forward KF at the time instant T. Recursively applying (E-24) and (E-25) yields the smoothed estimate $p(\boldsymbol{x}_k|\boldsymbol{y}_{k:T})$.

F Naming Convention

This thesis employs the naming convention of the Institute of Flight System Dynamics at the Technische Universität München. The following section details the most important parts. The respective coordinate frames are summarized in Appendix G.

Position

A typical position vector is given by

$$(\vec{\boldsymbol{r}}^{GP})_C = \begin{bmatrix} x^{GP} \\ y^{GP} \\ z^{GP} \end{bmatrix}_C . \qquad \textbf{(F-1)}$$

The vector $(\vec{\boldsymbol{r}}^{GP})_C$ describes the position of point P relative to point G, notated in the coordinate frame C.

Velocity

A typical velocity vector is given by

$$\left(\vec{\boldsymbol{v}}_K^{GP}\right)_{C_1}^{C_2} = \begin{bmatrix} u_K^{GP} \\ v_K^{GP} \\ w_K^{GP} \end{bmatrix}_{C_1}^{C_2} . \qquad \textbf{(F-2)}$$

The vector $\left(\vec{\boldsymbol{v}}_K^{GP}\right)_{C_1}^{C_2}$ describes the velocity of point P relative to point G. The velocity is of the kind K (e.g. wind, aerodynamic, kinematic), is relative to the C_2 frame and notated in the coordinate frame C_1.

Acceleration

A typical acceleration vector is given by

$$\left(\vec{\boldsymbol{a}}_K^{GP}\right)_{C_1}^{C_2C_3} = \begin{bmatrix} \dot{u}_K^{GP} \\ \dot{v}_K^{GP} \\ \dot{w}_K^{GP} \end{bmatrix}_{C_1}^{C_2C_3} . \qquad \textbf{(F-3)}$$

The vector $\left(\vec{\boldsymbol{a}}_K^{GP}\right)_{C_1}^{C_2C_3}$ describes the acceleration of point P relative to point G. The acceleration is of the kind K (e.g. wind, aerodynamic, kinematic), is relative to the C_2 frame, derived relative to the C_3 frame and notated in the coordinate frame C_1.

Rotational rate

A typical rotational rate vector is given by

$$\left(\vec{\boldsymbol{\omega}}_K^{C_2C_3}\right)_{C_1} = \begin{bmatrix} \omega_{K,x}^{C_2C_3} \\ \omega_{K,y}^{C_2C_3} \\ \omega_{K,z}^{C_2C_3} \end{bmatrix}_{C_1} . \qquad \textbf{(F-4)}$$

The vector $\left(\vec{\boldsymbol{\omega}}_{K}^{C_2C_3}\right)_{C_1}$ describes the rotational rate of the system C_3 relative to the coordinate system C_2. The rotational rate is of the kind K (e.g. wind, aerodynamic, kinematic) and notated in the coordinate frame C_1.

Rotational Acceleration

A typical rotational acceleration vector is given by

$$\left(\dot{\vec{\boldsymbol{\omega}}}_{K}^{C_2C_3}\right)_{C_1}^{C_4} = \begin{bmatrix} \dot{\omega}_{K,x}^{C_2C_3} \\ \dot{\omega}_{K,y}^{C_2C_3} \\ \dot{\omega}_{K,z}^{C_2C_3} \end{bmatrix}_{C_1}^{C_4} . \qquad \textbf{(F-5)}$$

The vector $\left(\dot{\vec{\boldsymbol{\omega}}}_{K}^{C_2C_3}\right)_{C_1}^{C_4}$ describes the rotational acceleration of the system C_3 relative to the coordinate system C_2. The rotational acceleration is of the kind K (e.g. wind, aerodynamic, kinematic), derived relative to the C_4 and notated in the coordinate frame C_1.

Forces

A typical force vector is given by

$$\left(\vec{\boldsymbol{F}}_{A}^{R}\right)_{C_1} = \begin{bmatrix} X_A^R \\ Y_A^R \\ Z_A^R \end{bmatrix}_{C_1} . \qquad \textbf{(F-6)}$$

The vector $\left(\vec{\boldsymbol{F}}_{A}^{R}\right)_{C_1}$ describes the forces acting on the point R. The force is of the kind A (e.g. aerodynamic, propulsive, gravity, total) and notated in the coordinate frame C_1.

Moments

A typical moment vector is given by

$$\left(\vec{\boldsymbol{M}}_{A}^{R}\right)_{C_1} = \begin{bmatrix} L_A^R \\ M_A^R \\ N_A^R \end{bmatrix}_{C_1} . \qquad \textbf{(F-7)}$$

The vector $\left(\vec{\boldsymbol{M}}_{A}^{R}\right)_{C_1}$ describes the moments relative to the point R. The moments are of the kind A (e.g. aerodynamic, propulsive, gravity, total) and notated in the coordinate frame C_1.

G Coordinate Frames

This section lists coordinate frames required for aerospace modelling and control.

Earth Centered Inertial Frame	
Index	I
Role	Notation frame for Newtonian inertial physics
Origin	Center of the earth
Translation	Around the sun with solar system
Rotation	None
x-axis	In equatorial plane, pointing towards vernal equinox
y-axis	In equatorial plane to form a right-hand system
z-axis	Rotation axis of the Earth

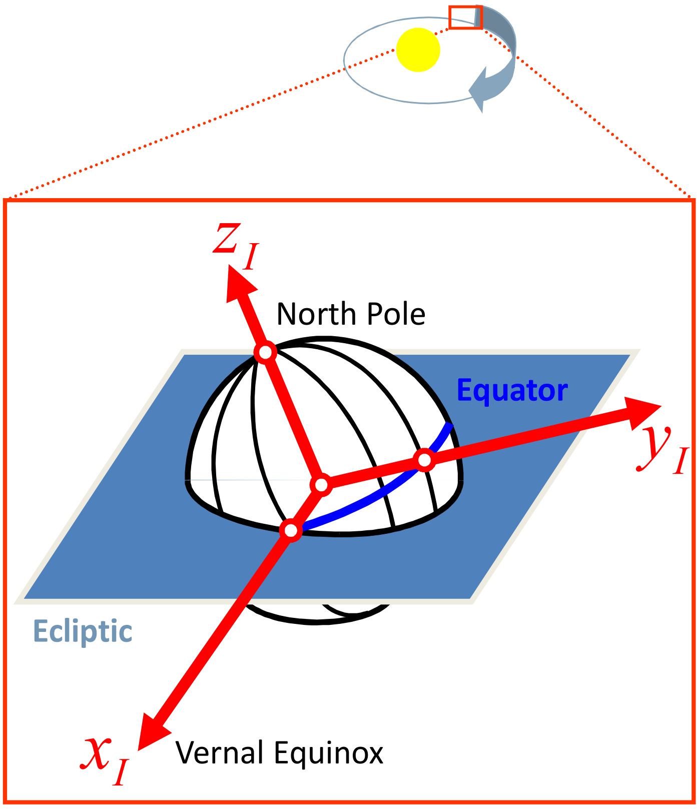


Figure G-1: ECI Frame (from [93])

Earth Centered Earth Fixed Frame	
Index	E
Role	Notation frame for Newtonian Inertial Physics
Origin	Center of the Earth
Translation	Around the sun with solar system
Rotation	None
x-axis	In equatorial plane, pointing towards vernal equinox
y-axis	In equatorial plane to form a right-hand system
z-axis	Rotation axis of the Earth

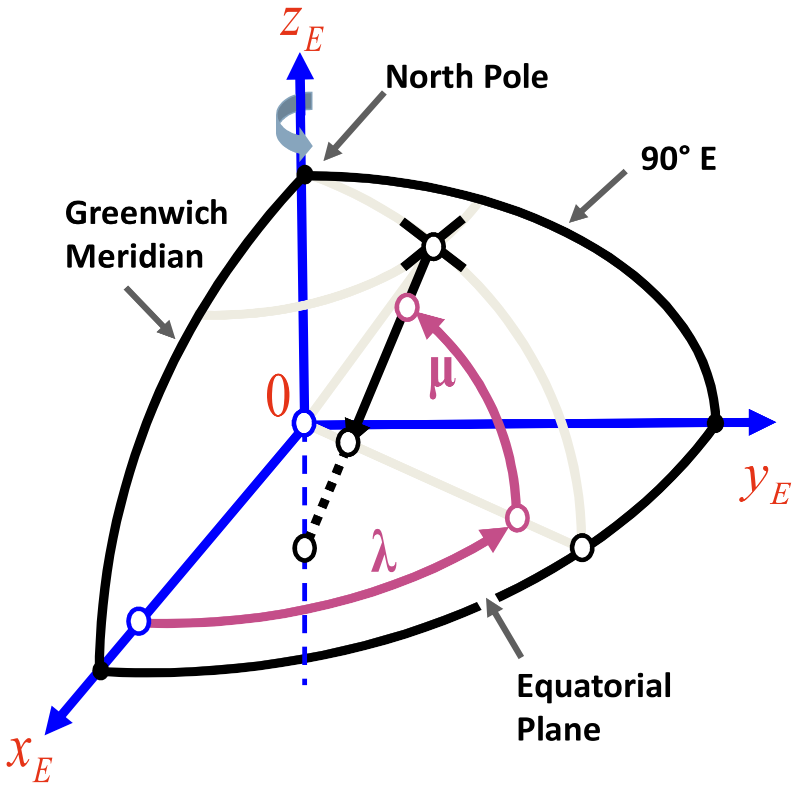


Figure G-2: ECEF Frame (from [93])

North-East-Down (NED) Frame	
Index	0
Role	Notation frame for velocity and orientation
Origin	Reference point of aircraft
Translation	Moves with aircraft reference point
Rotation	Rotates with transport rate to keep the NED-alignment
x-axis	Parallel to local geoid surface, pointing to geographic north pole
y-axis	Parallel to local geoid surface, pointing east to form a right-hand system
z-axis	Pointing downwards, perpendicular to local geoid surface

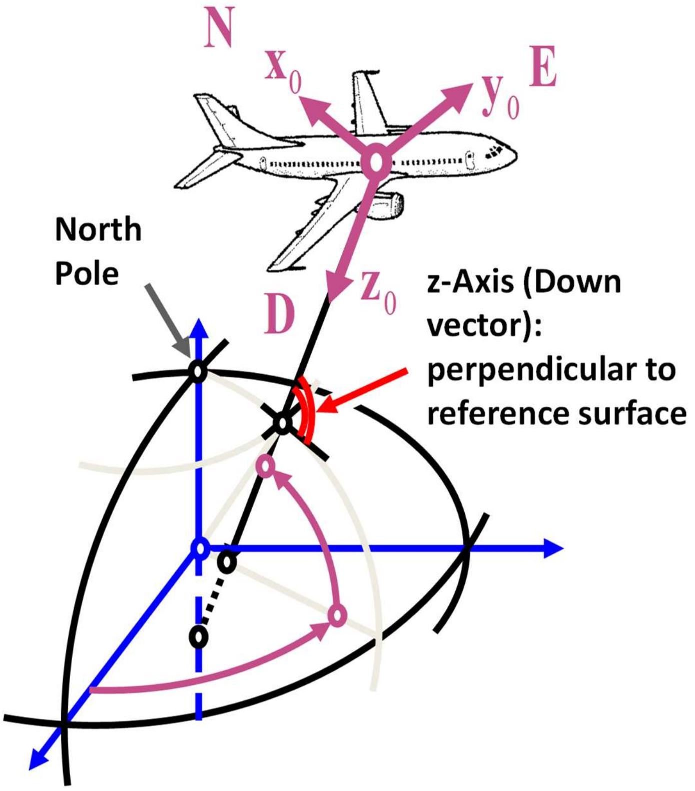


Figure G-3: NED-Frame (from [93])

Body-fixed Frame	
Index	B
Role	Notation frame
Origin	Reference point of aircraft
Translation	Moves with aircraft reference point
Rotation	Rotates with rigid body aircraft
x-axis	Pointing towards aircraft nose in symmetry plane
y-axis	Pointing to starboard wing to form a right-hand system
z-axis	Pointing downwards in symmetry plane, perpendicular to x- and y-axes

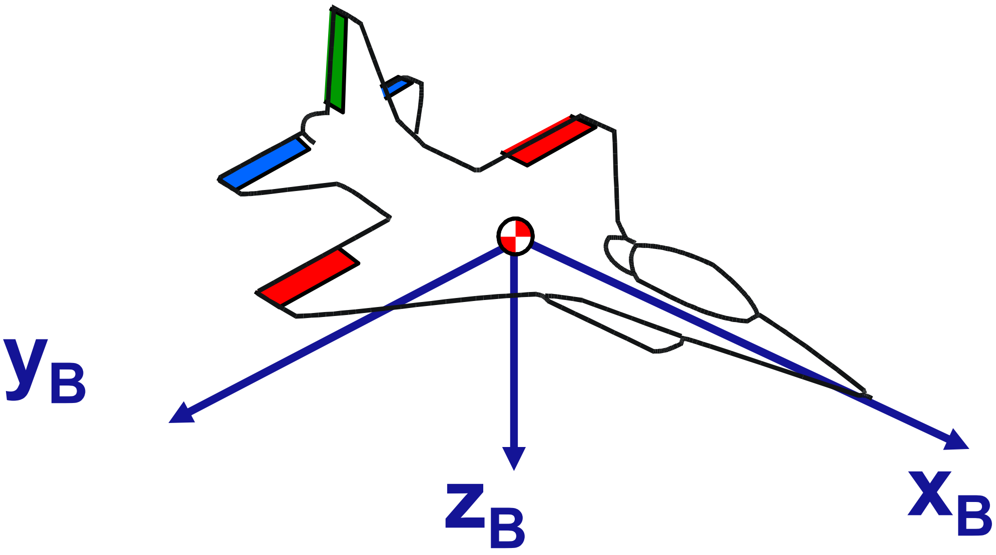


Figure G-4: Body-fixed frame (from [93])

Kinematic Frame	
Index	K
Role	Notation frame for flight path
Origin	Reference point of aircraft
Translation	Moves with aircraft reference point
Rotation	Rotates with direction of kinematic aircraft motion
x-axis	Aligned with the kinematic velocity, pointing into the direction of the kinematic velocity
y-axis	Pointing to the right, perpendicular to the x- and z- axes
z-axis	Pointing downwards, parallel to the projection of the local surface normal of the WGS-84 ellipsoid into a plane perpendicular to the x- axis

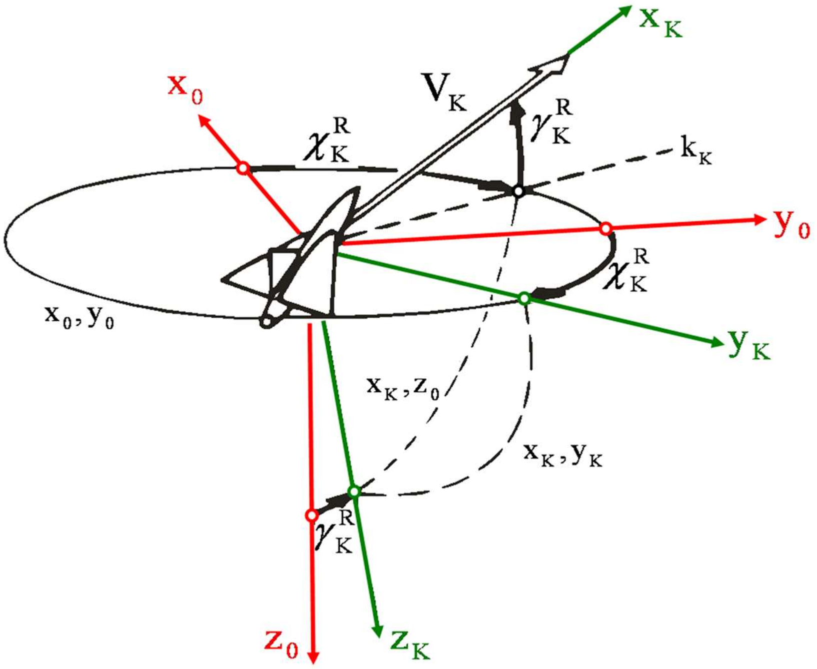


Figure G-5: Kinematic Frame (from [93])

Aerodynamic Frame	
Index	A
Role	Notation frame for aerodynamic flow
Origin	Aerodynamic reference point of aircraft
Translation	Moves with aircraft reference point
Rotation	Rotates with direction of airflow
x-axis	Aligned with aerodynamic velocity, pointing into the direction of the aerodynamic velocity
y-axis	Pointing to the right, perpendicular to the x- and z- axes
z-axis	Pointing downwards in the symmetry plane of the aircraft, perpendicular to the$x$- axis

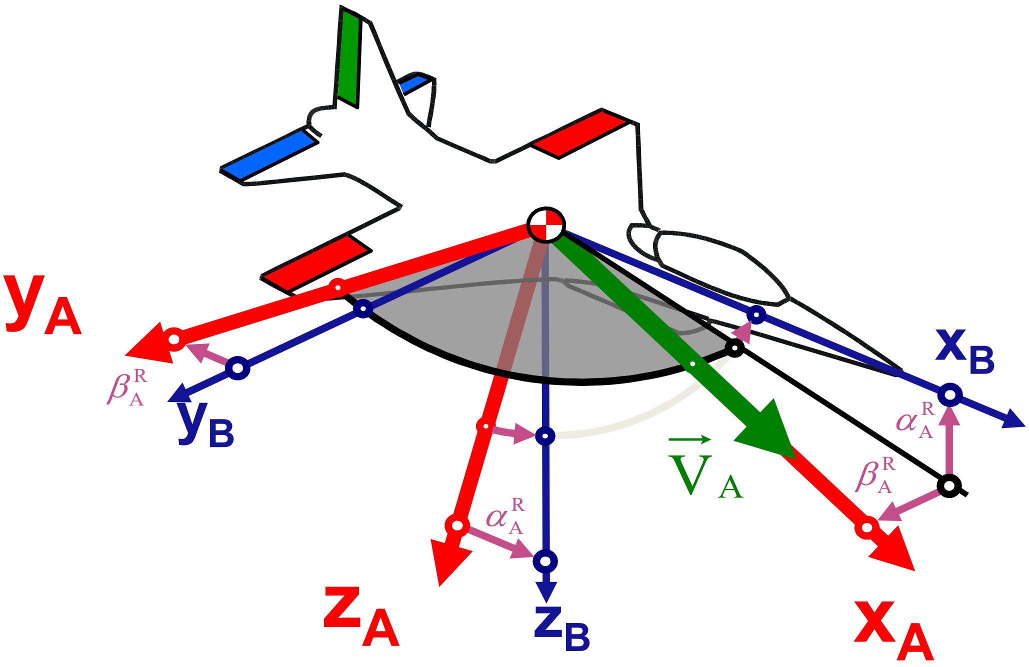


Figure G-6: Aerodynamic Frame (from [93])

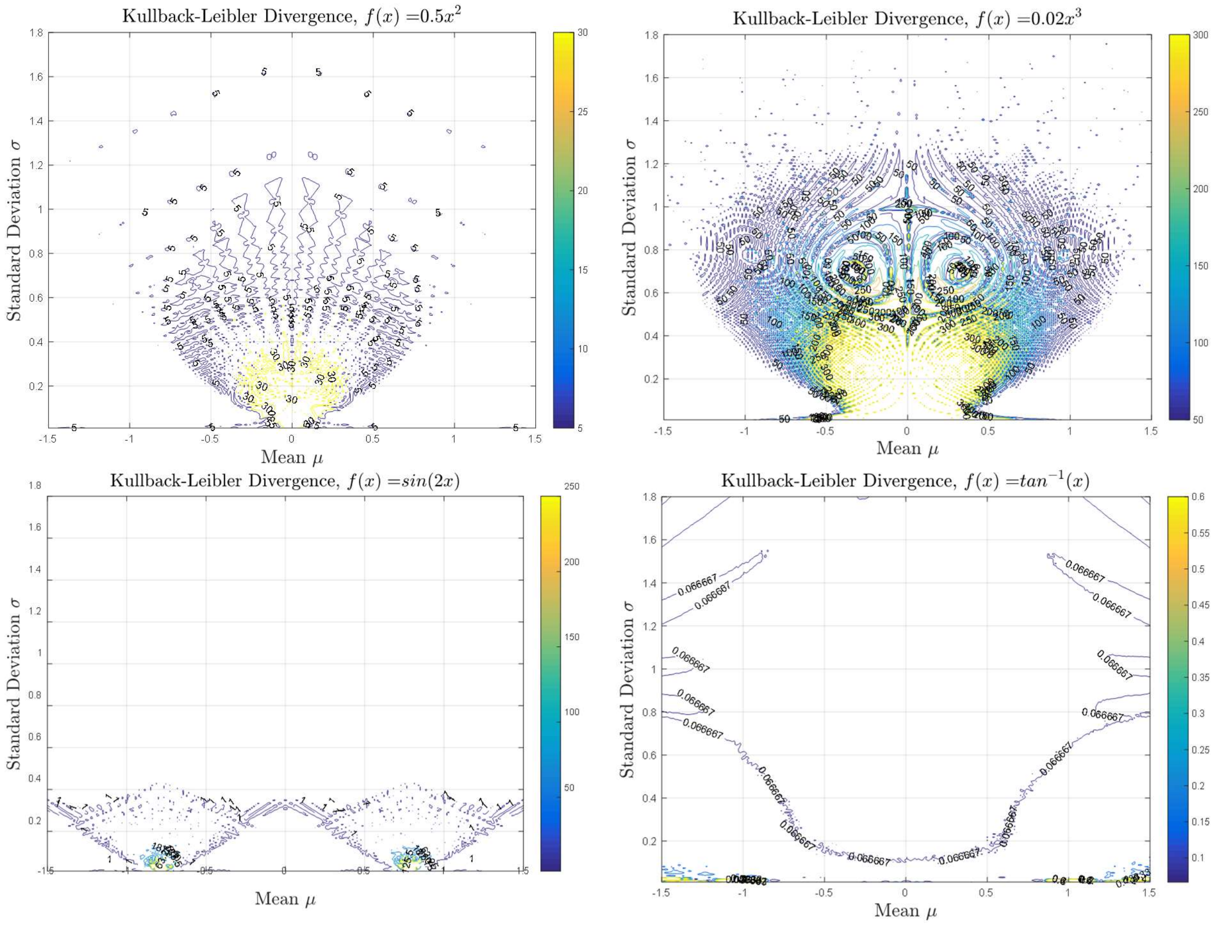
H Auxiliary Information

H.1 Simulation Parameter List for Example 4-1

Simulation Parameter	Numerical Value
General Simulation	
t_0	$0s$
dt	$0.001s$
t_{end}	$12s$
A_P	0
B_P	1
$p_B(t_0)$	$\sim N\left(-20\frac{\circ}{s}, 5.73\frac{\circ}{s}\right)$
Θ^*	-2
Baseline Controller	
A_{rm}	-1
B_{rm}	1
$p_{rm}(t_0)$	$0\frac{\circ}{s}$
$K_e \mid A_e$	$-1 \mid -1$
Adaptive Controller	
Γ	50
$Q \mid P$	$1 \mid 0.5$
σ	0.005
Θ_0	0

Table H-1: Simulation Parameters for Example 4-1

H.2 Adjusted contours for Example 4-4



Kullback-Leibler Divergence, $f(x) = 0.5x^2$

Kullback-Leibler Divergence, $f(x) = 0.02x^3$

Kullback-Leibler Divergence, $f(x) = sin(2x)$

Kullback-Leibler Divergence, $f(x) = tan^{-1}(x)$

Figure H-1: Adjusted scale for the Kullback-Leibler Divergence based comparison plots in Example 4-4

H.3 Simulation Parameter List for Example 4-6

Simulation Parameter	Numerical Value
General Simulation	
t_0	$0s$
dt	$0.001s$
t_{end}	$40s$
$\boldsymbol{A}_P$	$\begin{bmatrix}0 & 1\\0 & 0\end{bmatrix}$
$\boldsymbol{B}_P$	$\begin{bmatrix}0\\1\end{bmatrix}$
$[\Phi(t_0) \quad p_B(t_0)]$	$\begin{bmatrix}28.65^\circ & 0\frac{^\circ}{s}\end{bmatrix}$
$\boldsymbol{\Theta}^*$	$[-0.4 \quad -0.1157 \quad -0.0107 \quad -0.5]$
$\boldsymbol{v}$	$\sim N\left(\begin{bmatrix}0^\circ\\0\frac{^\circ}{s}\end{bmatrix}, \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right)$
Baseline Controller	
$\boldsymbol{A}_{rm}$	$\begin{bmatrix}0 & 1\\-4 & -4\end{bmatrix}$
$\boldsymbol{B}_{rm}$	$\begin{bmatrix}0\\4\end{bmatrix}$
$[\Phi_{rm}(t_0) \quad p_{B,rm}(t_0)]$	$\begin{bmatrix}0^\circ & 0\frac{^\circ}{s}\end{bmatrix}$
$\boldsymbol{K}_e^T \mid \boldsymbol{A}_e$	$[-5 \quad -5] \mid \begin{bmatrix}0 & 1\\-5 & -5\end{bmatrix}$
Adaptive Controller	
$\boldsymbol{\Gamma}$	$25\boldsymbol{I}_{4x4}$
$\boldsymbol{Q} \mid \boldsymbol{P}$	$\boldsymbol{I}_{2x2} \mid \begin{bmatrix}1.1 & 0.1\\0.1 & 0.12\end{bmatrix}$
σ	0.005
$\boldsymbol{\Theta}_0$	$[0 \quad 0 \quad 0 \quad 0]$
Fixed Point Smoother	
T	$0.03s$
dt	$0.001s$
GP based Uncertainty Quantification	
p_H	100
γ	0.3
μ	0.15
σ_n	0.4

Table H-2: Simulation Parameters for Example 4-6

H.4 Overlapping Coefficient for Different Nonlinear Functions

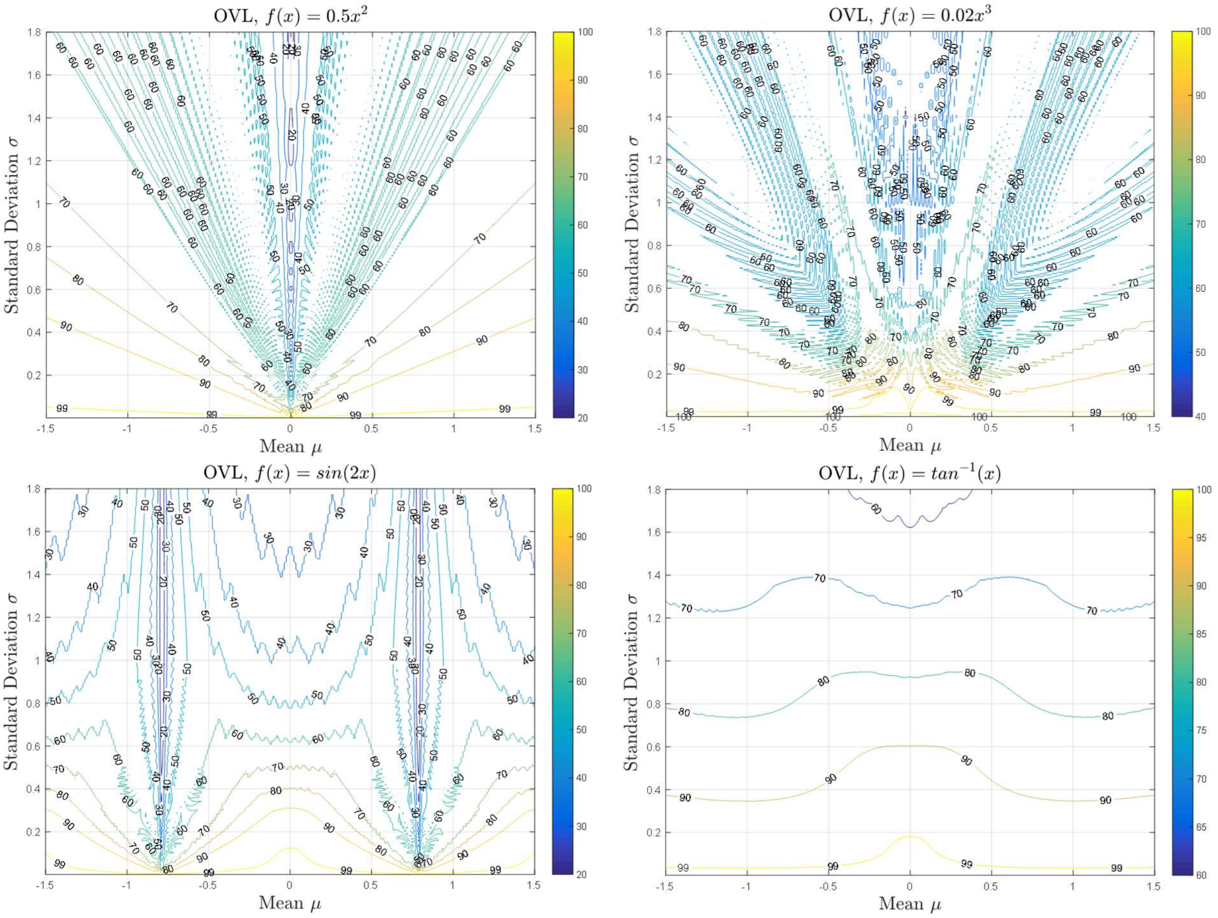


Figure H-2: Overlapping Coefficient of different nonlinear functions

H.5 Additional plots for the simulation example of section 5.1

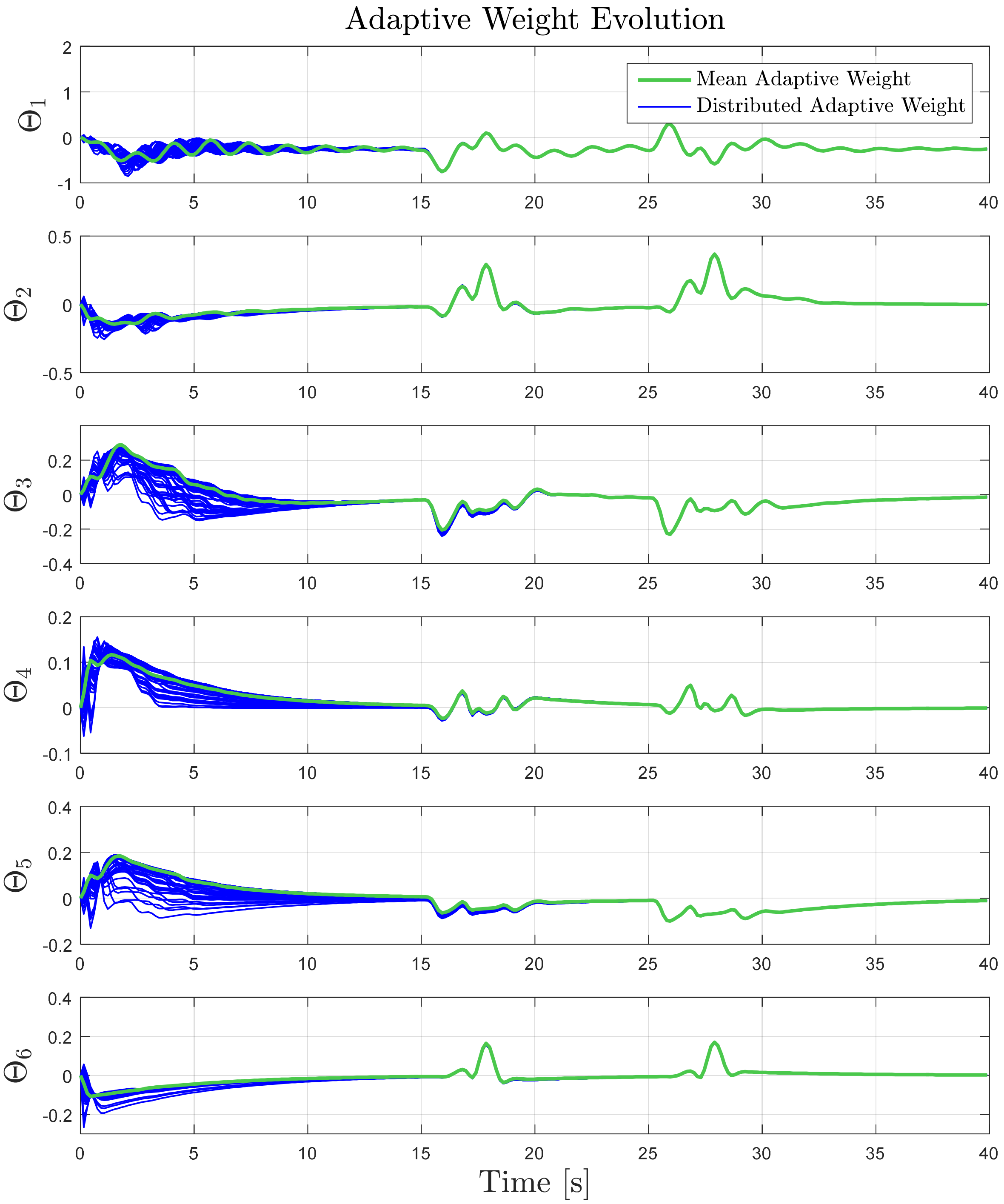


Figure H-3: Evolution of the adaptive parameters under wing-rock motion for normally distributed initial conditions

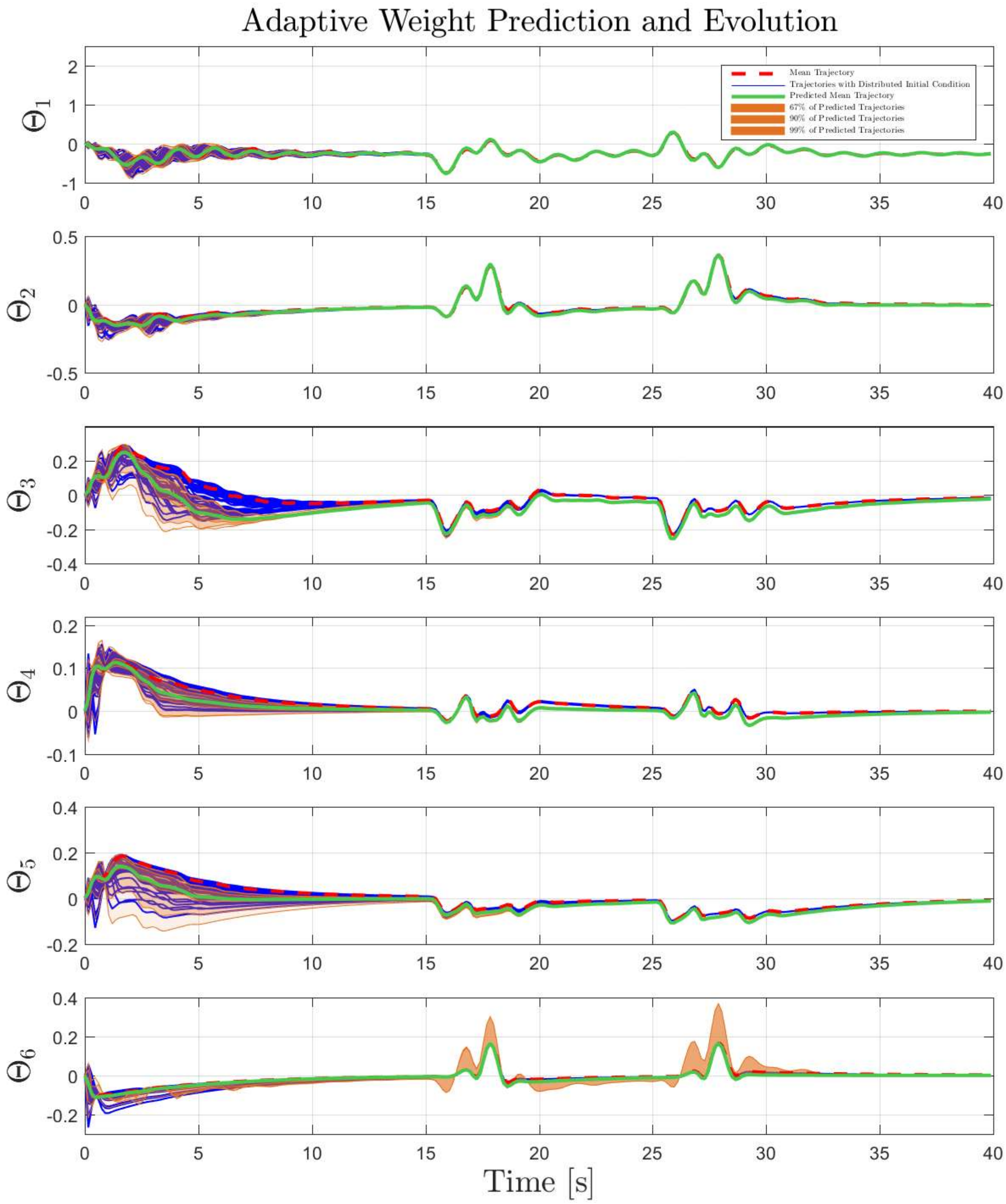


Figure H-4: Predicted evolution of the adaptive weights over 40 seconds of the wing-rock simulation leveraging a GMM

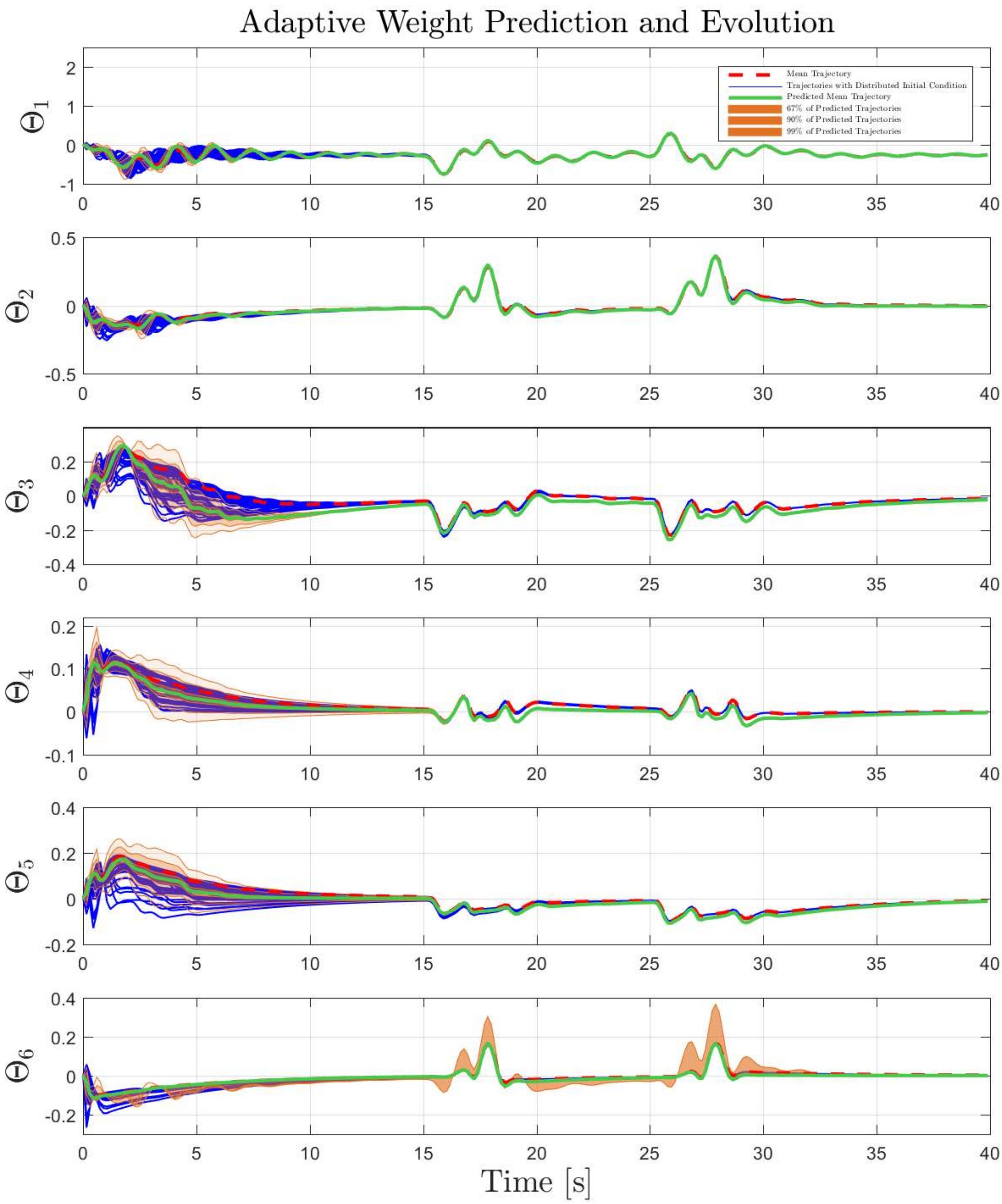


Figure H-5: Predicted evolution of the adaptive weights over 40 seconds of the wing-rock simulation leveraging a normal distribution only

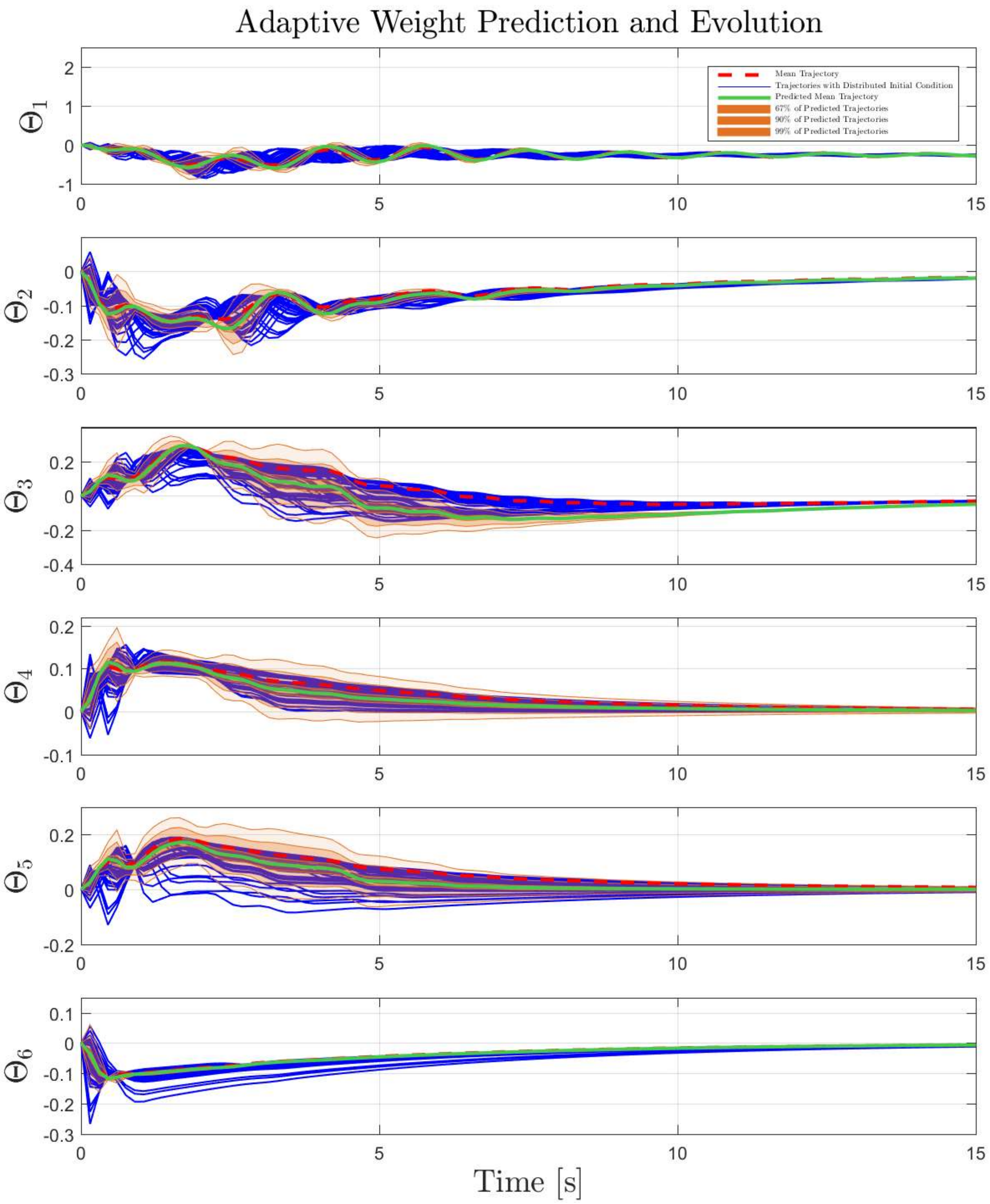


Figure H-6: Predicted evolution of the adaptive weights over 15 seconds of the wing-rock simulation leveraging a normal distribution only

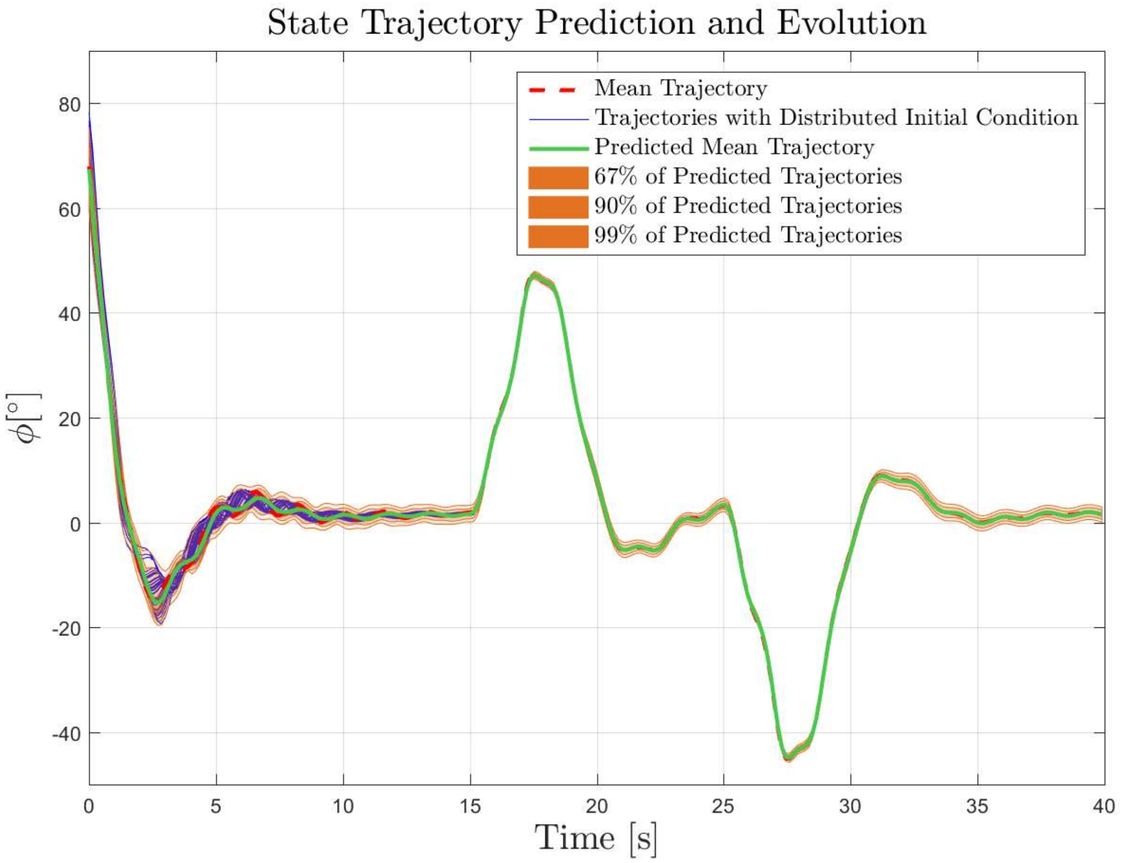


Figure H-7: Predicted roll angle ϕ over 40 seconds with GP regression based uncertainty estimation

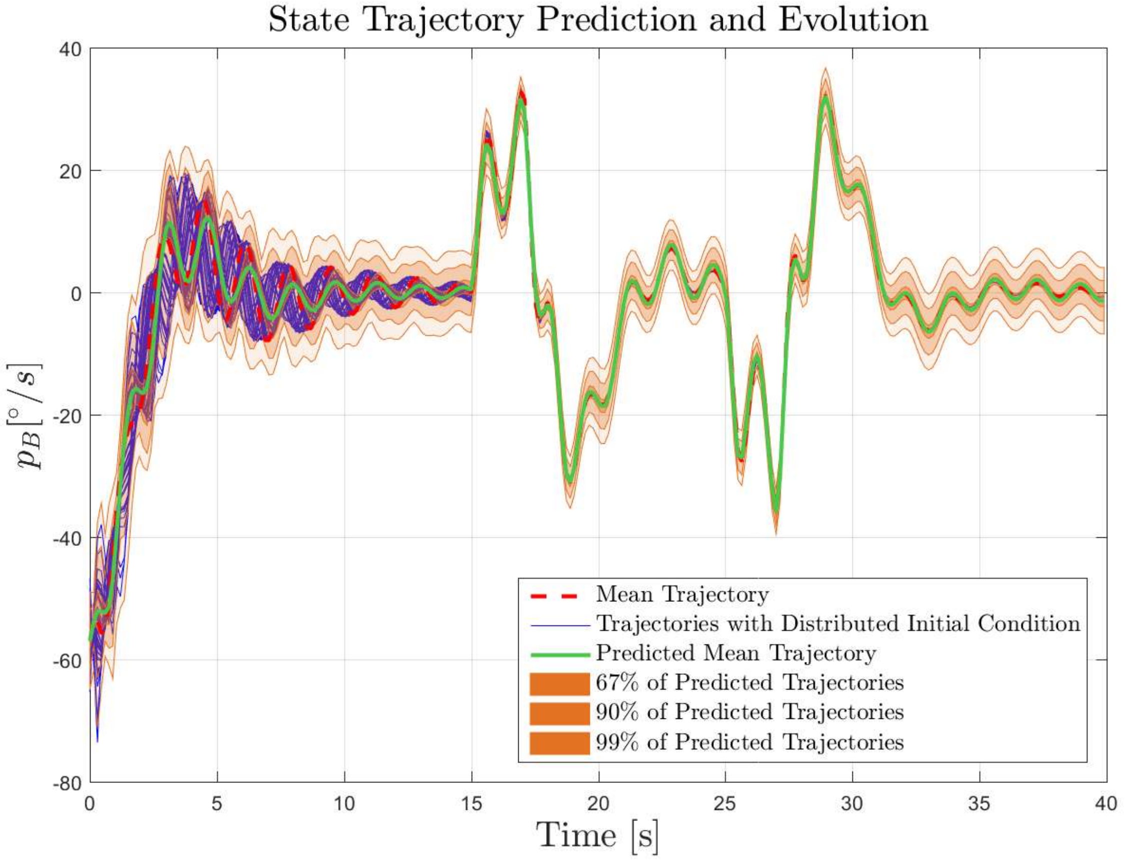


Figure H-8: Predicted roll rate p_B over 40 seconds with GP regression based uncertainty estimation

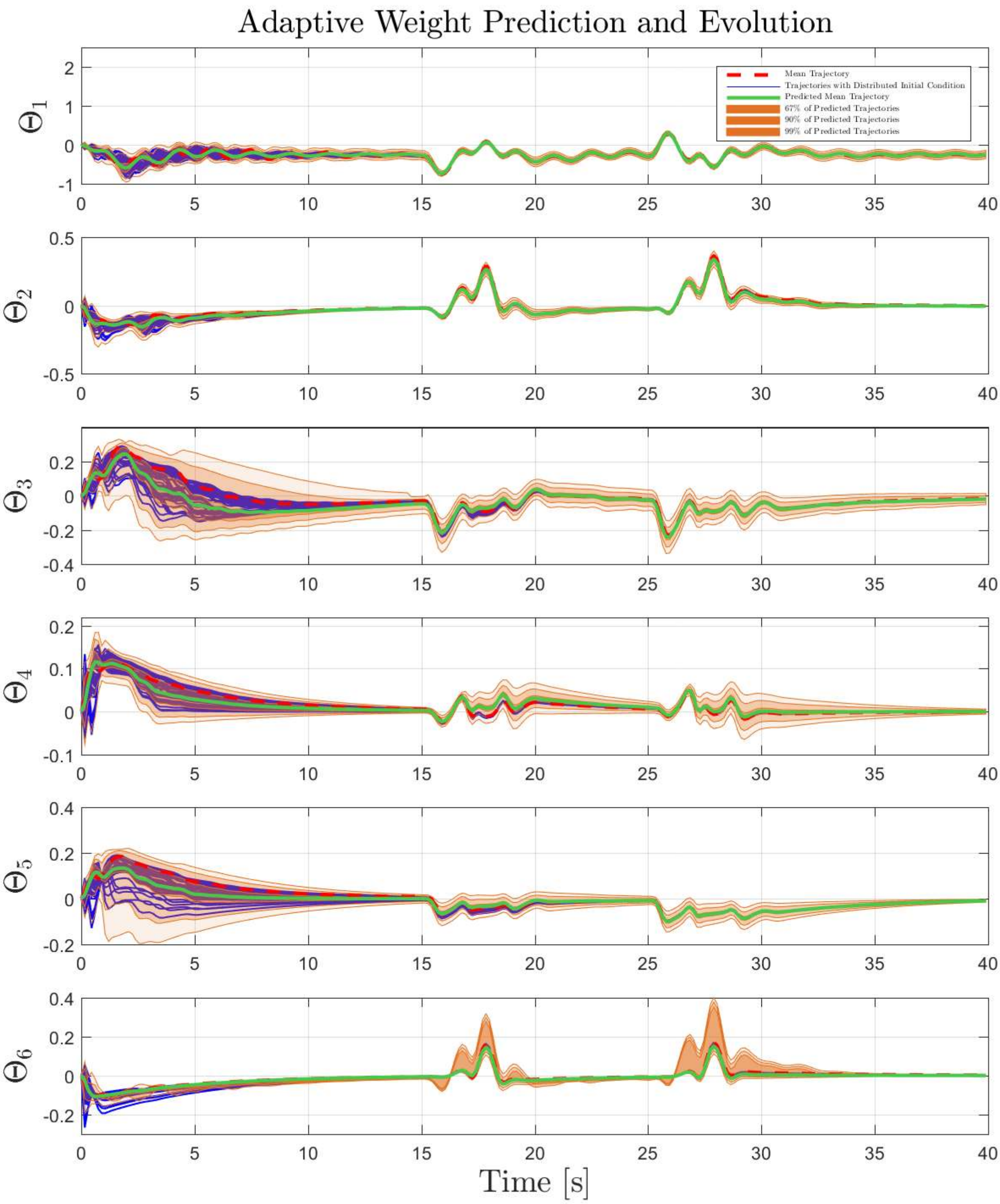


Figure H-9: Predicted evolution of the adaptive weights over 40 seconds of the wing-rock simulation leveraging a GMM and GP regression based uncertainty estimation

H.6 Additional plots for the simulation example of section 5.2

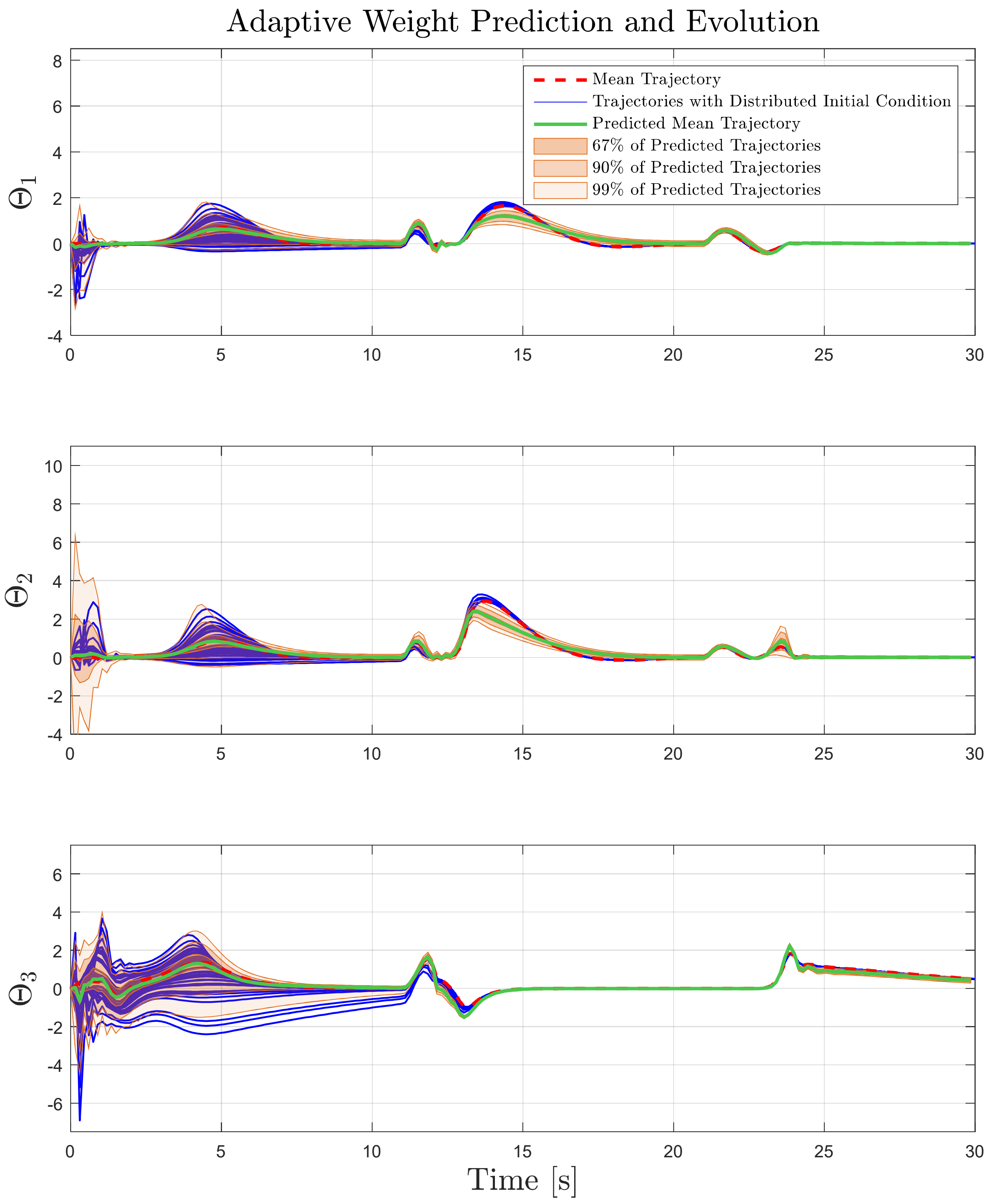


Figure H-10: Predicted evolution of the adaptive weights over 30 seconds in the short period simulation example leveraging a GMM

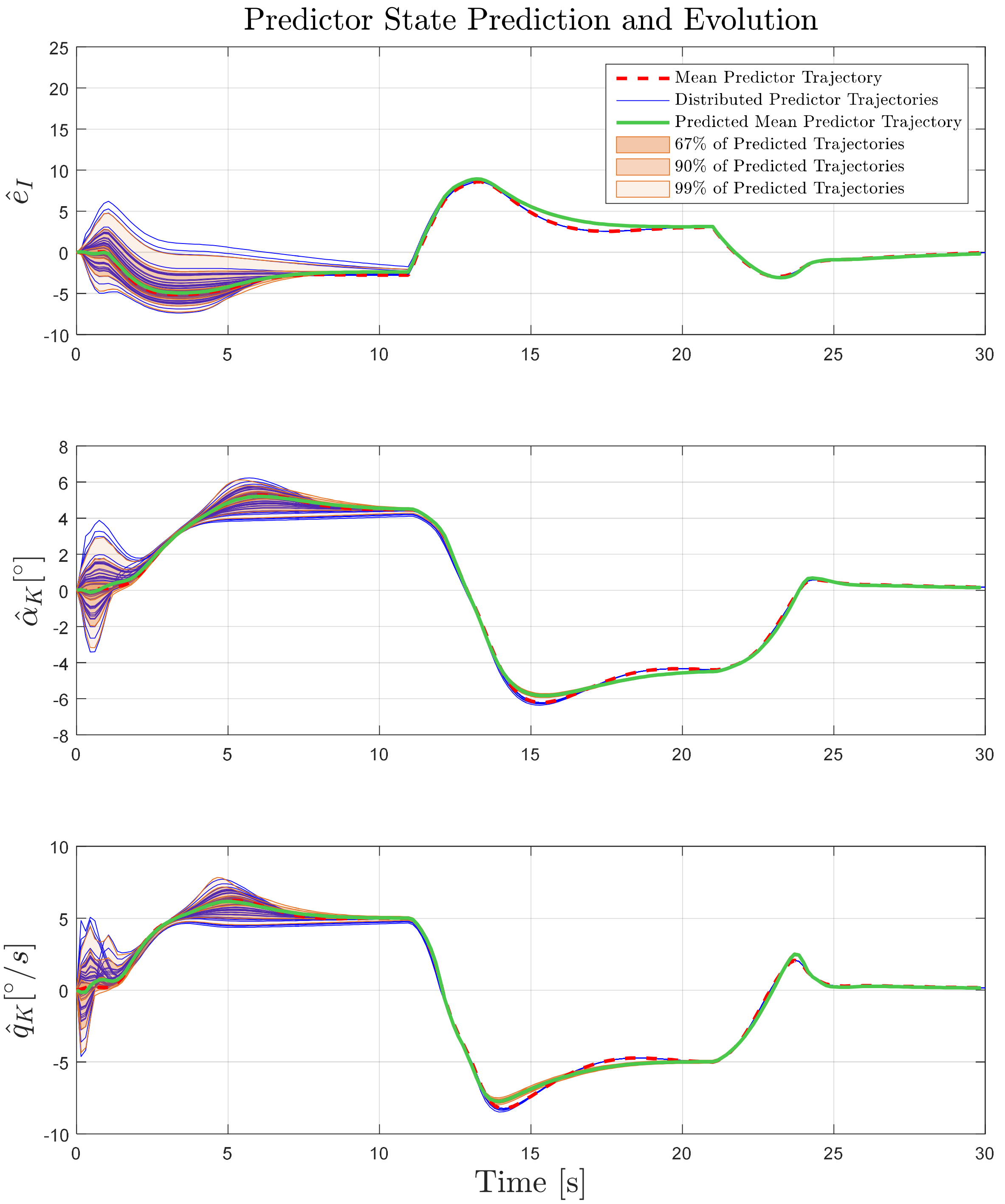


Figure H-11: Predicted evolution of the predictor states over 30 seconds in the short period simulation example leveraging a GMM