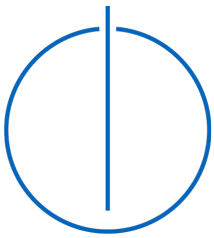
TUM School of Computation, Information and Technology

Technische Universität München

Variational Bayes for Continual Learning and Time-Series Forecasting

Richard Kurle



TUM School of Computation, Information and Technology

Technische Universität München

Variational Bayes for Continual Learning and Time-Series Forecasting

Richard Kurle

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz:	Prof. Dr.-Ing. Darius Burschka
Prüfer*innen der Dissertation:	1. Prof. Dr. Stephan Günnemann 2. apl. Prof. Dr. Georg Groh

Die Dissertation wurde am 23.09.2022 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 09.02.2023 angenommen.

Abstract

Deep neural networks trained with gradient-based methods are the workhorse of many data-driven solutions for real-world problems due to impressive performance in various domains, such as computer vision, natural language processing, and time-series modelling. However, certain applications such as continual learning with non-stationary data, multi-modal/multi-view learning, or probabilistic time-series forecasting pose challenging problems that require non-trivial extensions of the established methods. In order to address these problems, this dissertation takes the probabilistic approach to machine learning, providing a coherent framework for constructing models that combine prior or expert knowledge about latent random variables in the model with information inferred from data through the likelihood function. One of the key challenges is to develop practical algorithms that allow for tractable inference of the latent variables, usually requiring simplifying model assumptions or approximations. Variational Bayes is a well-established approximate-inference method that is applicable to deep learning in different ways: neural networks can be used to define the parameters of the conditional distributions in deep latent-variable models and deep state-space models, thus modelling aleatoric uncertainty; on the other hand, Bayesian neural networks treat the weights and biases of the neural network as latent variables, reflecting epistemic uncertainty.

This cumulative dissertation develops variational-Bayesian methods for the above-mentioned applications. In case of continual learning with non-stationary data, sequential variational approximation and adaptation methods for Bayesian neural networks are developed. The variational approximation consists of a running memory of raw data and a Gaussian distribution that summarises the rest of the data, and the proposed adaptation methods allow the model to cope with data drift. Subsequent work explains the main problem of the variational-Bayesian method used for (continual) learning: invariances in the likelihood of over-parametrised models have a detrimental effect on variational-Bayesian methods, since the invariances lead to an additional gap in the ELBO objective that incentivises posterior collapse to the prior. Another contribution is a neural-variational-inference method for learning deep latent-variable models of multiple sources (modalities or views). The proposed approach uses individual inference models for each source and the inferred beliefs are integrated via a product- or mixture-of-experts approach. For probabilistic time-series forecasting, an extension of switching Gaussian linear systems with additional state-to-switch recurrence and a decoder-type emission model is developed, allowing for improved long-term forecasts and to represent multivariate non-linear and non-Gaussian emission noise. An efficient Rao-Blackwellised particle filter is proposed for inferring the latent state variables and learning the parameters. The above-mentioned methods are evaluated extensively, with qualitative experiments on synthetic data to provide intuition, and well-known benchmarks in the respective fields to evaluate the methods quantitatively.

Acknowledgements

I would like to thank the people who supported, encouraged, and motivated me while working on this dissertation.

First, I want to thank my supervisors Stephan Günnemann and Patrick van der Smagt, who gave me this great opportunity and made sure that I stay on track, while leaving me the full freedom to explore the topics that excite me, thank you.

I am grateful that I had the opportunity to work and spend time with wonderful colleagues at the Volkswagen MLRL. Sincere thanks to Adnan, Alexej, Alexandros, Atanas, Baris, Djalel, Felix, Grady, Justin, Karolina, Maximilian K., Maximilian S., Michelle, Nutan, and Philip. Special thanks to Botond, you taught me a lot during our whiteboard sessions and discussions.

I feel very fortunate that I had the opportunity to work in an amazing environment at AWS AI Labs during my internship and later full time. Many thanks to Jan, Syama, and Tim for enabling the internship during which I learned a lot from you. Special thanks to Jan and Bernie, your continuous support, effort, and trust in me is invaluable. I feel also lucky to have collaborated with Ralf, your excitement for challenging problems is truly motivating and inspiring.

I also want to thank all my close friends for their support. Many thanks to Manon, you impressively endured the exposure to many random ideas and thoughts during the first years, while having no clue what I was talking about, thank you. Luckily, Alexej later imposed this duty on himself. Jokes aside, I really enjoyed our inspiring discussions in the kitchen and during sports, and I am extremely grateful to have you and Alina as close friends, thank you for all the support. I also want to thank Sebi, I will miss our thought provoking walks around Olympia park, thank you for keeping an eye on the actually important things. Special thanks to Caro for your support close to the finish line.

Finally, I want to thank my family and foremost my parents, thank you for supporting and believing in me throughout my life.

Contents

Acronyms

NN neural network. 2, 3, 4, 17, 18, 19, 21, 22, 23, 24, 25, 26, 35, 38, 62, 120, 121, 122
BNN Bayesian neural network. 3, 4, 7, 8, 10, 17, 18, 19, 62, 120, 122, 123
CNN convolutional neural network. 17, 122
RNN recurrent neural network. 17
MLP multilayer perceptron. 17, 20
LVM latent-variable model. 21, 22, 26, 34
DLVM deep latent-variable model. 3, 4, 7, 8, 10, 21, 22, 23, 24, 120, 122, 123
VAE variational autoencoder. 3, 22, 23, 25, 86, 123
IWAE importance-weighted autoencoder. 23
SSM state-space model. 4, 26, 27, 29, 31, 34, 35, 121
DSSM deep state-space model. 3, 4, 7, 8, 10, 26, 27, 120, 121, 122, 123
GLS Gaussian linear dynamical system. 26, 27, 31, 34
SGLS switching Gaussian linear system. 4, 100, 121
MC Monte Carlo. 8, 9, 11, 12, 14, 19, 20, 22, 121
IS importance sampling. 8, 12, 13, 14, 16
SIS sequential importance sampling. 14, 15
SMC sequential Monte Carlo. 4, 8, 13, 14, 15, 26, 27, 35, 36, 100, 121, 122
PF particle filter. 4, 14, 100
PS particle smoother. 14
MCMC Markov Chain Monte Carlo. 3, 18, 29
SGD stochastic gradient descent. 2, 11, 17
ML maximum likelihood. 2, 3, 17, 18, 21, 22, 29, 34
MAP maximum a posteriori. 3, 9, 18, 22, 29, 34
VB variational Bayes. 8, 9, 29
ELBO evidence lower bound. 4, 9, 10, 11, 12, 19, 22, 23, 29, 30, 33, 34, 36, 120, 121, 123
EM expectation maximisation. 22, 29, 30, 33
VEM variational expectation maximisation. 22
KL Kullback-Leibler divergence. 9, 10, 11, 12, 19, 22
PCA principal component analysis. 21
ICA independent component analysis. 21

Part I

Introduction and Fundamentals

1 Introduction

1.1 Motivation

Machine learning has become relevant to many scientific fields and business applications, such as computer vision, natural language processing, economics, robotics, medicine, genetics and genomics, and molecular biology [1, 2, 3, 4, 5, 6, 7]. Some of the most important use cases are explaining latent structure in the data, detecting anomalies, and making predictions on novel inputs or for future time points. To ensure robustness and reliability, it is essential to take into account sources of uncertainty, e.g. arising from a lack of sufficient training data, the randomness inherent in the process underlying the data, or changes in the data distribution. The Bayesian probabilistic framework provides a simple but principled recipe to achieve this: construct a probabilistic model that implements a priori assumptions about the problem or phenomenon at hand, and apply standard rules of probability to infer the posterior distribution over the unknown random variables—such as model parameters and additional latent variables—from information in the observed data. This probabilistic approach to machine learning can be understood as the process of inferring plausible models that are likely to have generated the observed data [8]. It allows for a coherent way to combine prior information (e.g. from domain-specific experts) with observed data, represent uncertainty, make predictions, and compare models. This dissertation builds on the Bayesian framework in combination with deep neural network (NN) models to address several fundamental challenges arising in the above-mentioned applications, such as i) continual learning and adaptation to distribution shift, ii) sensor fusion considering missing, noisy and anomalous data, and iii) long-term time-series forecasting.

The main challenge in the probabilistic approach is that models for which the posterior distribution is computationally tractable are often too simple to describe the data accurately. The Bayesian community has developed a wealth of algorithms to *approximate* the intractable integrals occurring in the inference procedure of more complex models. These can be categorised into *stochastic* sampling-based algorithms, *deterministic* analytical approximations with simpler distributions, and combinations thereof. On the other hand, deep NNs are a popular class of non-linear models that have achieved remarkable success in the last decade [9, 10], as they can approximate complex non-linear functions of high-dimensional data, while scaling well to large datasets. The NN parameters are often learned through maximum likelihood (ML) estimation with stochastic gradient descent (SGD) based optimisation algorithms without taking into account uncertainties. However, especially since the last decade, NNs are now often used within the Bayesian framework to combine the best of both worlds.

The combination of Bayesian inference algorithms and NN models, nowadays sometimes referred to as Bayesian deep learning [11, 12, 13], has a long history with the first approaches in the early 1990s [14, 15, 16, 17, 18, 19]. This thesis is concerned with two types of Bayesian approaches to

deep learning: Bayesian neural networks (BNNs) treat the NN parameters (weights and biases) as latent random variables. In contrast to deterministic point-estimation methods such as ML or maximum a posteriori (MAP) estimation, distributions over the parameter values expresses epistemic uncertainty arising from a lack of information in the training data, which can be reduced as more data is observed. The second type of Bayesian approaches to deep learning use NNs as parametrised functions that predict the parameters of conditional distributions. The conditional distributions relate the latent and observed variables (data) in directed graphical models [20]. In contrast to deterministic representations in deep NNs, distributions over latent variables express aleatoric uncertainty, i.e. the irreducible uncertainty that is inherent in the process by which the data was obtained. Two model classes that fall under this second type of approach are relevant for this dissertation, namely deep latent-variable models (DLVMs) and deep state-space models (DSSMs).

Although inference for both the parameters and latent variables is conceptually equivalent from a Bayesian perspective, practical approximate-inference algorithms differ substantially due to differences in the respective graphical model. Learning/inference of the parameters of BNNs is often based on Markov Chain Monte Carlo (MCMC) methods such as Hamiltonian Monte Carlo [18] or Langevin Monte Carlo [21], with variants that allow for mini-batch stochastic optimisation [22], variational inference [19, 23, 24], or Laplace's approximation [17]. On the other hand, the parameters of DLVMs and DSSMs are nowadays often learned through stochastic variational inference [25, 26, 27], *amortising* the cost of inference of the latent variables by using a NN encoder to predict the variational posterior approximation from the data.

Despite the several extensions and advancements of BNNs [28, 29, 30, 31, 32], DLVMs [33, 34, 35, 36, 37], and DSSMs [38, 39, 40, 41], several problems remain challenging: predictive uncertainty estimates are often not reliable. For instance, standard variational-Bayesian methods for learning BNNs result in severe underfitting and poor predictions [30], and a similar problem has also been observed in variational autoencoders (VAEs) [42]. On the other hand, sampling-based approximations are inefficient for high-dimensional integrals. Moreover, applying Bayesian deep-learning methods to more challenging or non-standard problems usually requires further approximations or model assumptions for efficient inference.

1.2 Outline and Contributions

This dissertation consists of three parts. Part I provides an introduction and the technical fundamentals for Bayesian inference used in conjunction with NNs. Ch. 1 introduces the research area, provides a concise overview of existing approaches and open problems, and highlights the contributions of this thesis. Ch. 2 establishes the Bayesian approach to machine learning and the most relevant approximate-inference techniques, which provide the basis for learning/inference in the proposed models. The three subsequent chapters of Part I then describe three model classes, respectively, which are central to the four publications that constitute this cumulative dissertation: Ch. 3 introduces BNNs and important approximate-inference methods. BNNs are used for the variational continual-learning method proposed in Ch. 6 and the analysis of the short-comings of variational inference for these models (Ch. 7). Secondly, DLVMs are introduced in Ch. 4, providing the basis for the extension to multi-source inference in Ch. 8. Finally,

DSSMs—which are used for time-series modelling and probabilistic forecasting in Ch. 9—and important recursive inference-methods for these models are described in Ch. 5.

The four peer-reviewed publications from Part II constitute the main content and contributions of this cumulative dissertation. These publications share a similar methodology, using variational-Bayesian methods for inferring model parameters or other latent variables in probabilistic models. However, the contributions address substantially different problem areas, and the inference methods were developed for three different model classes.

Ch. 6 develops a Bayesian approach to continual learning with non-stationary data. In such scenarios, data (sets) arrive sequentially, inferences and predictions should be performed immediately, and the data distribution is assumed to change over time, e.g. as a consequence of unpredictable external factors. A memory-based variational approximation is developed for sequential inference of the posterior of the NN parameters. Two proposed adaptation methods enable this sequential posterior approximation to adapt to changes in the data distribution.

The main bottleneck of the continual-learning approach is that variational Bayes for BNNs often leads to poor posterior approximations. Ch. 7 explains this through the detrimental effect of invariances in the likelihood of over-parametrised models. The invariances lead to an additional gap in the evidence lower bound (ELBO) objective compared to a purpose-built posterior approximation that takes into account the invariances. A detailed analysis of the detrimental effect of translation invariance is presented for over-parametrised Bayesian linear models. It is shown that, while the true posterior can be constructed from a mean-field parametrisation, this optimum is not achievable if the invariance gap is not taken into account in the ELBO.

Ch. 8 presents a neural-variational-inference method for DLVMs that model the generative process of data from multiple sources (modalities or views). The DLVM assumes conditionally independent observations and it is learned jointly with multiple inference models that infer beliefs from individual sources. Methods to compare (e.g. detect conflicts) and combine these beliefs are developed and evaluated experimentally.

Ch. 9 addresses efficient inference in state-space models (SSMs) for time-series forecasting. Switching Gaussian linear systems (SGLSs) are extended by a state-to-switch recurrence and a decoder-type emission model, improving long-term forecasts and allowing to model multivariate non-linear and non-Gaussian emission noise. An efficient Rao-Blackwellised particle filter is developed for this model: expectations w.r.t. variables in the conditionally Gaussian linear part of the model are computed in closed-form, and expectations w.r.t. the remaining variables are approximated using variational sequential Monte Carlo (SMC).

Part III concludes the thesis with a summary in Ch. 10 and outlining promising ideas for future research directions in Ch. 11.

1.3 List of Contributions

This cumulative dissertation is based on four peer-reviewed papers that were published at international conferences. Tab. 1.1 provides a list of the publications that contribute to this dissertation and publications that were co-authored while pursuing the degree.

Bayesian Neural Networks.
Continual Learning with Bayesian Neural Networks for Non-Stationary Data [43] (Ch. 6). Richard Kurle, Botond Cseke, Alexej Klushyn, Patrick van der Smagt, Stephan Günnemann. International Conference on Learning Representations (ICLR) 2020.
On Symmetries in Variational Bayesian Neural Nets [44]. Richard Kurle, Tim Januschowski, Jan Gasthaus, Yuyang Wang. Bayesian Deep Learning NeurIPS workshop (BDL) 2021.
On the detrimental effect of invariances in the likelihood for variational inference [45] (Ch. 7). Richard Kurle, Ralf Herbrich, Tim Januschowski, Yuyang Wang, Jan Gasthaus. Advances in Neural Information Processing Systems (NeurIPS) 2022.
Deep Latent-Variable Models.
Metrics for Deep Generative Models [46]. Nutan Chen*, Alexej Klushyn*, Richard Kurle*, Xueyan Jiang, Justin Bayer, Patrick Smagt. International Conference on Artificial Intelligence and Statistics (AISTATS) 2018. * indicates equal contributions.
Multi-Source Neural Variational Inference [47] (Ch. 8). Richard Kurle, Stephan Günnemann, Patrick Van der Smagt. Association for the Advancement of Artificial Intelligence (AAAI) 2019.
Learning Hierarchical Priors in VAEs [37]. Alexej Klushyn, Nutan Chen, Richard Kurle, Botond Cseke, Patrick van der Smagt. Advances in Neural Information Processing Systems (NeurIPS) 2019.
Deep State-Space Models.
Normalizing Kalman Filters for Multivariate Time Series Analysis [48]. Emmanuel de Bézenac, Syama Sundar Rangapuram, Konstantinos Benidis, Michael Bohlke-Schneider, Richard Kurle, Lorenzo Stella, Hilaf Hasson, Patrick Gallinari, Tim Januschowski. Advances in Neural Information Processing Systems (NeurIPS) 2020.
Deep Rao-Blackwellised Particle Filters for Time Series Forecasting [49] (Ch. 9). Richard Kurle, Syama Sundar Rangapuram, Emmanuel de Bézenac, Stephan Günnemann, Jan Gasthaus. Advances in Neural Information Processing Systems (NeurIPS) 2020.
Latent Matters: Learning Deep State-Space Models [50]. Alexej Klushyn, Richard Kurle, Maximilian Soelch, Botond Cseke, Patrick van der Smagt. Advances in Neural Information Processing Systems (NeurIPS) 2021.
Deep Explicit Duration Switching Models for Time Series [51]. Abdul Fatir Ansari, Konstantinos Benidis, Richard Kurle, Ali Caner Turkmen, Harold Soh, Alex Smola, Bernie Wang, Tim Januschowski. Advances in Neural Information Processing Systems (NeurIPS) 2021.

Table 1.1: List of peer-reviewed and published papers in chronological order per topic corresponding to background chapters 3, 4, and 5. Publications that contribute to this cumulative dissertation are listed in black and references to the respective chapters are provided. Publications that are not included in this thesis are listed in gray.

2 Probabilistic Modelling and Inference

In the Bayesian interpretation of probability theory, probability provides a *measure of reasonable expectation* of an event drawn from the probability distribution [52]; in other words, it describes the *degree of belief* in one of several hypotheses [53]. From this interpretation it is evident that probability is not an intrinsic property of the physical world. Instead, it represents the available information about the truth of the considered hypotheses, allowing for reasoning with incomplete information [54]. It is also important to note that probability is not necessarily subjective to the practitioner. However, the above interpretation states the fact that probability and inference are always conditioned on (unavoidable) assumptions such as the considered probability distribution [53]. For example, the publication in Ch. 8 uses the term *belief* to refer to the model's information about a latent variable. In this case, the probability distribution is conditioned on an inference model (encoder) and data from one or several sources.

The central theme of Bayesian inference is to update the *prior* belief $p(H)$ that some hypothesis H is true with new information (evidence) E through the *likelihood* function $p(E|H)$. This is achieved using *Bayes rule*, resulting in the *posterior* distribution

$$p(H|E) = \frac{p(E|H)p(H)}{p(E)}. \tag{2.1}$$

Note that the prior and posterior belief is not absolute; these distributions over the hypotheses are relative to the set of all considered hypotheses.

Many problems arising in machine learning can be approached by applying the above general inference methodology to probabilistic models and data. To this end, Sec. 2.1 briefly introduces two types of probabilistic models that are considered in this dissertation. Sec. 2.2 then explains how the above inference methodology can be applied to these probabilistic models. Several fundamental *approximate*-inference methods—that are relevant to subsequent chapters and the main contributions of this dissertation—are then described in Sec. 2.3, 2.4, and 2.5. Specific instantiations of these more general inference methods for the model classes considered in this dissertation are provided in the subsequent chapters.

2.1 Probabilistic Modelling

This dissertation considers and distinguishes probabilistic models with *global* parameters θ and *local* latent variables $\mathbf{x}$. A *parametric* model is a family of probability distributions that is defined through a finite set of parameters θ from a parameter space Θ, where every parametrisation $\theta \in \Theta$ of some model class denoted by $\mathcal{M}$ assigns a probability $p(\mathcal{D}|\theta, \mathcal{M})$ to the observed dataset, $\mathcal{D} = \{\mathbf{d}^{(n)}\}_{n=1}^{N}$. The data points $\mathbf{d}^{(n)}$ can be single observations or tuples, such as

pairs of inputs $\mathbf{u}^{(n)}$ and targets $\mathbf{y}^{(n)}$. In the probabilistic approach to machine learning, the observed dataset and the specific parameter values of the model thus correspond to the evidence and hypotheses in Eq. (2.1), respectively [55]. The parameters are *global* in the sense that the same parameters apply to the entire dataset. To this end, the conditional probability distribution $p(\mathcal{D}|\theta, \mathcal{M}) = \prod_{n=1}^{N} p(\mathbf{d}^{(n)}|\theta, \mathcal{M})$ serves as the *likelihood* function, and $p(\theta|\mathcal{M})$ is the prior probability of the parameters given the model class $\mathcal{M}$. Examples of such parametric models expressing uncertainty over the parameters include Bayesian linear regression and BNNs (see Ch. 3).

Many probabilistic models such as DLVMs (see Ch. 4) and DSSMs (see Ch. 5) use additional *latent variables* that are *local* in the sense that they are associated with individual observations $\mathbf{d}^{(n)}$ in the dataset. From the Bayesian perspective, both the parameters and latent variables correspond to hypotheses with a prior distribution, i.e. parameters are similarly treated as unknown random variables. In practice, however, parameters are distinguished from latent variables as they are used for different modelling purposes. Whereas parameters are associated with the entire dataset and thus fixed in their number, latent variables are usually associated with individual data points (or sets thereof) and thus grow in number with the dataset size N, i.e. $p(\mathbf{d}^{(n)}|\theta, \mathcal{M}) = \int p(\mathbf{d}^{(n)}, \mathbf{x}^{(n)}|\theta, \mathcal{M}) d\mathbf{x}^{(n)}$, for each data point in the dataset.

Distributions over parameters and latent variables also represent different notions of uncertainty [56, 57]. *Epistemic uncertainty* derives from the lack of information about the model, which could in principle be reduced, e.g. if more data was observed. This type of uncertainty is thus reflected in the distribution over model parameters. *Aleatoric uncertainty* results from the unpredictable random nature of the process by which the data was obtained, which cannot be reduced as more data is observed. This type of uncertainty can be expressed through distributions over latent variables affecting individual observations.

2.2 Bayesian Inference for Machine Learning

From a probabilistic perspective, *learning*, *prediction* and *model comparison* can all be seen as forms of inference [8]:

- *Learning* amounts to performing posterior inference to find the parameters that are most probable—within the specified prior and likelihood model—given the data:

$$p(\theta|\mathcal{D}, \mathcal{M}) = \frac{p(\mathcal{D}|\theta, \mathcal{M}) p(\theta|\mathcal{M})}{p(\mathcal{D}|\mathcal{M})}. \tag{2.2}$$

- *Prediction* involves marginalising over the model parameters in the joint distribution $p(\mathcal{D}', \theta|\mathcal{D}, \mathcal{M}) = p(\mathcal{D}'|\theta, \mathcal{M}) p(\theta|\mathcal{D}, \mathcal{M})$ of the parameters and some hypothetical/new data $\mathcal{D}'$, conditioned on the observed data $\mathcal{D}$:

$$p(\mathcal{D}'|\mathcal{D}, \mathcal{M}) = \int p(\mathcal{D}'|\theta, \mathcal{M}) p(\theta|\mathcal{D}, \mathcal{M}) d\theta. \tag{2.3}$$

  This so-called *posterior predictive* thus involves averaging the predictive distributions $p(\mathcal{D}'|\theta, \mathcal{M})$ weighted by the posterior probability of the respective parameter values.

- *Model comparison* and selection can also be considered as an inference problem, where the most probable *model class* is inferred from the data:

$$p(\mathcal{M}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathcal{M})p(\mathcal{M})}{p(\mathcal{D})}. \tag{2.4}$$

  As there is typically no good reason to prefer a particular model class a priori, models are often compared and selected based on the *marginal likelihood* $p(\mathcal{D}|\mathcal{M}) = \int p(\mathcal{D}|\theta, \mathcal{M})p(\theta|\mathcal{M})$, also referred to as *model evidence*.

The dependence on the model class $\mathcal{M}$ will be omitted in the subsequent sections for convenience. It is, however, important to note that posterior inference is always relative to the considered model class $\mathcal{M}$ and parameter space Θ. Model misspecification can therefore lead to unexpected and suboptimal results for prediction and model comparison, even if the posterior is inferred exactly [58].

Approximate inference. The standard problem when using the Bayesian approach is that computing the posterior distribution is analytically tractable only for simple models such as Gaussian linear models. Thus, one of the main challenges in Bayesian inference is to find approximations that are suitable for the considered problem and model class. Approximate-inference methods can be broadly categorised as *deterministic* (analytical) and *stochastic* (Monte Carlo) approximations or a combination of both. Deterministic approximations are characterised by providing analytical distributions, which are usually simpler than the distribution of interest that may have no known analytical form. Stochastic methods represent the distribution of interest by a set of random samples and approximate the corresponding integrals numerically by a finite average involving these samples. Several methods combine analytical and Monte Carlo methods, e.g. to approximate still intractable integrals involving analytical variational approximations (e.g. amortised variational inference, see Sec. 4.2).

The following sections (Secs. 2.3, 2.4, and 2.5) explain several important approximate-inference methods that are applicable to a plethora of different models. Subsequent chapters (Ch. 3, 4, and 5) then provide more details and present practical inference algorithms for the model classes considered in the publications contributing to this thesis. These general methods are applicable to both the inference of parameters and latent variables in probabilistic models. For instance, methods for variational approximations of the posterior of BNN parameters (weights and biases) are presented in Sec. 3.4, and amortised variational inference for the latent variables of DLVMs is explained in Sec. 4.2. On the other hand, importance sampling (IS) (Sec. 2.4) and SMC (Sec. 2.5) are used in this dissertation only for DLVMs and DSSMs, respectively. The subsequent sections therefore introduce these fundamental inference methods, denoting latent variables by $\mathbf{x}$ and observations by $\mathbf{y}$.

2.3 Variational Bayes

Variational Bayes (VB) is often used to approximate probability distributions (usually the posterior) by simpler distributions, as well as approximate expectations arising in machine-learning problems [59, 60]. The approximation is given by a distribution q from a family of

distributions $\mathcal{Q}$; it is therefore often referred to as a deterministic approximation. The variational distribution usually has a convenient and simpler structure than the true posterior, e.g. with additional independence assumptions, which allow to simplify computations involving this distribution.

2.3.1 Evidence Lower Bound: Variational Inference as Optimisation

The approximation is found by minimising the Kullback-Leibler divergence (KL) between approximating and true posterior distribution, casting inference as an optimisation problem w.r.t. the *variational distribution*:

$$q^*(\mathbf{x}) = \operatorname*{argmin}_{q(\mathbf{x})\in\mathcal{Q}} \mathrm{KL}\left[q(\mathbf{x}) \,||\, p(\mathbf{x}|\mathbf{y})\right]. \tag{2.5}$$

The central idea of variational approximation methods is that minimising the KL in Eq. (2.5) is equivalent to maximising the evidence lower bound (ELBO). This objective is often tractable or can be further approximated e.g. through Monte Carlo estimation. As implied by its name, the ELBO is a lower bound to the log marginal likelihood

$$\begin{aligned}
\log p(\mathbf{y}) &= \log \int p(\mathbf{y}|\mathbf{x})p(\mathbf{x})d\mathbf{x} \\
&= \log \int q(\mathbf{x})\frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{q(\mathbf{x})}d\mathbf{x} \\
&\geq \int q(\mathbf{x}) \log \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{q(\mathbf{x})}d\mathbf{x} =: \mathcal{L}_{\mathrm{ELBO}}\left(\mathbf{y}, q(\mathbf{x})\right).
\end{aligned} \tag{2.6}$$

The variational distribution is introduced by multiplying with $\frac{q(\mathbf{x})}{q(\mathbf{x})}$, and the ELBO is then obtained by applying Jensen's inequality [61].

The ELBO is often written in one of the following forms with useful interpretations:

$$\mathcal{L}_{\mathrm{ELBO}}\left(\mathbf{y}, q\right) = \mathbb{E}_{q(\mathbf{x})}\left[\log p(\mathbf{y}|\mathbf{x}) + \log p(\mathbf{x}) - \log q(\mathbf{x})\right] \tag{2.7a}$$

$$= \underbrace{\mathbb{E}_{q(\mathbf{x})}\left[\log p(\mathbf{y}|\mathbf{x})\right]}_{\text{reconstruction}} - \underbrace{\mathrm{KL}\left[q(\mathbf{x}) \,||\, p(\mathbf{x})\right]}_{\text{info gain}} \tag{2.7b}$$

$$= \underbrace{\mathbb{E}_{q(\mathbf{x})}\left[\log p(\mathbf{y}, \mathbf{x})\right]}_{\text{neg. energy}} + \underbrace{\mathrm{H}\left[q(\mathbf{x})\right]}_{\text{entropy}} \tag{2.7c}$$

The KL form (Eq. (2.7b)) shows that the ELBO trades off the cost of reconstructing/predicting the data from the posterior approximation with the information gained from the data (w.r.t. the prior) as measured by the KL; for this reason, the KL is sometimes referred as the regularisation term. The entropy form (Eq. (2.7c)) is often preferred in other fields such as statistical mechanics, where the ELBO is known as the negative variational free energy. This form shows that $q(\mathbf{x})$ is encouraged to distribute most of the probability mass in regions of high joint probability (negative energy), while also maximising the uncertainty (entropy) of the posterior approximation. This latter form is useful for comparing VB to MAP estimation, showing that the entropy term prevents the approximation from collapsing to a point estimate.

The gap/tightness between the log marginal likelihood and the ELBO can be quantified as

$$\begin{aligned}\mathcal{L}_{\text{ELBO}}(\mathbf{y}, q) &= \int q(\mathbf{x}) \log \frac{p(\mathbf{x}, \mathbf{y})}{q(\mathbf{x})} d\mathbf{x} \\ &= \log p(\mathbf{y}) + \underbrace{\int q(\mathbf{x}) \log \frac{p(\mathbf{x}|\mathbf{y})}{q(\mathbf{x})} d\mathbf{x}}_{-\,\mathrm{KL}[q(\mathbf{x})\,||\,p(\mathbf{x}|\mathbf{y})]}.\end{aligned} \tag{2.8}$$

Thus, the gap is

$$\log p(\mathbf{y}) - \mathcal{L}_{\text{ELBO}}(\mathbf{y}, q) = \mathrm{KL}\left[q(\mathbf{x})\,||\,p(\mathbf{x}|\mathbf{y})\right], \tag{2.9}$$

where the log marginal likelihood $\log p(\mathbf{y})$ was factored out of the integral since it is independent of $\mathbf{x}$. For the same reason, maximising $\mathcal{L}_{\text{ELBO}}$ amounts to minimising $\mathrm{KL}\left[q(\mathbf{x})\,||\,p(\mathbf{x}|\mathbf{y})\right]$ w.r.t. $q(\mathbf{x})$, i.e.

$$q^*(\mathbf{x}) = \operatorname*{argmax}_{q(\mathbf{x}) \in \mathcal{Q}} \mathcal{L}_{\text{ELBO}}(\mathbf{y}, q) = \operatorname*{argmin}_{q(\mathbf{x}) \in \mathcal{Q}} \mathrm{KL}\left[q(\mathbf{x})\,||\,p(\mathbf{x}|\mathbf{y})\right]. \tag{2.10}$$

If the approximating family q contains the true posterior $p(\mathbf{x}|\mathbf{y})$, it is also the optimal solution. If this solution is also found by the optimisation algorithm, the bound is tight and the KL between approximate and true posterior is zero.

2.3.2 Monte Carlo Approximation and Stochastic Backpropagation

Though Eq. (2.10) provides a general approach for inference in probabilistic models, it remains to show how this optimisation can be turned into a practical algorithm. The remainder of this section describes the key ideas allowing for (doubly) stochastic backpropagation, lead to practical variational inference algorithms for several important models such as BNNs (Ch. 3), DLVMs (Ch. 4), and DSSMs (Ch. 5).

Variational parametrisation. The problem is simplified by restricting the family of *variational distributions* to some parametric distribution with parameters ϕ and optimising the ELBO objective w.r.t. these *variational parameters* instead:

$$\mathcal{L}_{\text{ELBO}}(\mathbf{y}, \phi) = \mathbb{E}_{q_\phi(\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{y}|\mathbf{x})\, p(\mathbf{x})}{q_\phi(\mathbf{x})}\right]. \tag{2.11}$$

$$\phi^* = \operatorname*{argmax}_{\phi} \mathcal{L}_{\text{ELBO}}(\mathcal{D}, \phi). \tag{2.12}$$

Since the optimisation problem is now w.r.t. the *variational parameters*, gradient-based parameter optimisation algorithms can be used.

Monte Carlo estimation. In contrast to the log marginal likelihood, the expectations in the ELBO can be approximated using Monte Carlo (MC) methods. An unbiased estimate of the ELBO objective can be obtained by making MC approximations, replacing the expectation w.r.t. the variational distribution by a finite average with P random samples $\mathbf{x}_p \sim q_\phi(\mathbf{x})$:

$$\hat{\mathcal{L}}_{\text{ELBO}}\left(\mathbf{y}, \phi\right) \approx \frac{1}{P} \sum_{p=1}^{P} \left[\log \frac{p(\mathbf{y}|\mathbf{x}^{(p)})\, p(\mathbf{x}^{(p)})}{q_\phi(\mathbf{x}^{(p)})} \right], \quad \text{where } \mathbf{x}^{(p)} \sim q_\phi(\mathbf{x}). \tag{2.13}$$

The form in Eq. (2.13) makes no assumptions whether some terms can be computed in closed-form. For example, if the ELBO is written in KL form (Eq. (2.7b)) or entropy form (Eq. (2.7c)), the KL or the entropy can often be computed in closed-form and need not be estimated stochastically.

Eq. (2.13) yields a tractable objective function, but it remains to show how this function can be optimised with SGD optimisation methods. The main problem is that the stochastic sampling operation $\mathbf{x}^{(p)} \sim q_\phi(\mathbf{x})$ is not differentiable, and, thus, obtaining MC estimates of the gradient w.r.t. ϕ is challenging. Two important methods that address this problem are presented subsequently.

Black-box gradient estimation. A general method for estimating gradients of the ELBO w.r.t. the variational parameters is *black-box variational inference* [62], which is also known as the score-function gradient estimator [63] and used also in other machine learning areas such as reinforcement learning for policy gradient methods [64, 65, 66]. Denoting $f_\phi(\mathbf{x}) = \log p(\mathbf{y}|\mathbf{x}) + \log p(\mathbf{x}) - \log q_\phi(\mathbf{x})$, such that $\mathcal{L}_{\text{ELBO}}\left(\mathbf{y}, \phi\right) = \mathbb{E}_{q_\phi(\mathbf{x})}\left[f_\phi(\mathbf{x})\right]$, the gradient of the ELBO can be formulated as

$$\begin{aligned}
\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{x})}\left[f_\phi(\mathbf{x})\right] &= \nabla_\phi \int q_\phi(\mathbf{x}) f_\phi(\mathbf{x}) d\mathbf{x} \\
&= \int \nabla_\phi \left(q_\phi(\mathbf{x}) f_\phi(\mathbf{x})\right) d\mathbf{x} \\
&= \int f_\phi(\mathbf{x}) \nabla_\phi q_\phi(\mathbf{x}) d\mathbf{x} + \int q_\phi(\mathbf{x}) \nabla_\phi f_\phi(\mathbf{x}) d\mathbf{x} \\
&= \mathbb{E}_{q_\phi(\mathbf{x})}\left[f_\phi(\mathbf{x}) \nabla_\phi \log q_\phi(\mathbf{x})\right] + \underbrace{\mathbb{E}_{q_\phi(\mathbf{x})}\left[\nabla_\phi f_\phi(\mathbf{x})\right]}_{=0},
\end{aligned} \tag{2.14}$$

where ∇_ϕ denotes the gradient w.r.t. the variational parameters. The last line of the derivation exploits the *log-derivative trick*, $\nabla_\phi q_\phi(\mathbf{x}) = q_\phi(\mathbf{x}) \nabla_\phi \log q_\phi(\mathbf{x})$. An unbiased gradient estimate can be obtained through MC estimation with a finite number of samples from q_ϕ. The advantage of this estimator is that it does not require f_ϕ to be differentiable; it suffices to evaluate f_ϕ and the so-called score function $\nabla_\phi \log q_\phi(\mathbf{x})$ with samples drawn from q_ϕ. Several variance reduction techniques such as Rao-Blackwellisation and control variates are used [62] in practice, however, *reparametrisation* yields gradient estimates with lower variance in most (though not all) cases [63].

Reparametrisation gradient estimation. For many common distributions, differentiability can be achieved using the *reparametrisation trick* [26, 27] and its extensions and generalisations [67, 68, 69], allowing for gradient estimation with lower variance—in most but not all cases—compared to the above black-box estimates [63]. Reparametrisation expresses a density through a deterministic invertible transformation $\mathbf{x} = \mathcal{T}(\epsilon; \phi)$ of an auxiliary random variable sampled from a simpler density $\epsilon \sim q(\epsilon)$ with fixed parameters. For instance, the Gaussian $q_\phi(\mathbf{x}) = \mathcal{N}(\mathbf{x};\, \mathbf{m}, \mathbf{V})$ can be *reparametrised* as

$$\mathbf{x} = \mathcal{T}(\epsilon; \phi) = \mathbf{V}^{\frac{1}{2}}\epsilon + \mathbf{m}, \quad \text{where } \epsilon \sim \mathcal{N}(\epsilon;\, 0, 1), \tag{2.15}$$

where $\mathbf{V}^{\frac{1}{2}}$ is the square root (Cholesky decomposition) of the covariance matrix.

Importantly, the auxiliary variable ϵ is independent of ϕ. Consequently, gradients w.r.t. expectations involving ϕ can be written in terms of expectations w.r.t. the auxiliary variable ϵ using chain rule:

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{x})}\big[f_\phi(\mathbf{x})\big] = \mathbb{E}_{q(\epsilon)}\big[\, \nabla_{\mathbf{x}} f_\phi(\mathbf{x})|_{\mathbf{x}=\mathcal{T}(\epsilon;\phi)}\, \nabla_\phi \mathcal{T}(\epsilon; \phi)\big], \tag{2.16}$$

where $f_\phi(\mathbf{x}) = \log p(\mathbf{y}|\mathbf{x}) + \log p(\mathbf{x}) - \log q_\phi(\mathbf{x})$. Sampling from $\epsilon \sim q(\epsilon)$ does not need to be differentiable since this auxiliary density has fixed parameters. Gradients of the MC estimate of Eq. (2.13) are computed analogously for either the whole ELBO objective or e.g. just the log likelihood term if the remaining terms (KL) can be computed in closed-form, e.g. if both distributions are in the exponential family [70].

2.4 Importance Sampling

Importance sampling (IS) is a sampling-based technique for approximating expectations of functions, $\mathbb{E}_{\nu(\mathbf{x})}[g(\mathbf{x})]$, where $\nu(\mathbf{x})$ is referred to as *target distribution* and $g(\mathbf{x})$ is often called the *test function*. Such integrals are very common in machine learning, e.g. for prediction and likelihood estimation. Standard MC approaches estimate the expectation directly through

$$\mathbb{E}_{\nu(\mathbf{x})}[g(\mathbf{x})] \approx \frac{1}{P}\sum_{p=1}^{P} g(\mathbf{x}^{(p)}),$$

with $\mathbf{x}^{(p)} \sim \nu(\mathbf{x})$. It is easy to see that this estimator is unbiased and the variance of the error decreases with $O(1/P)$. However, this simple estimator may be problematic in machine learning applications for several reasons. First, it is sometimes difficult to sample from $\nu(\mathbf{x})$, e.g. if the target distribution is the posterior $\nu(\mathbf{x}) = p(\mathbf{x}|\mathbf{y})$ with intractable normalisation constant. Furthermore, estimating the expectation directly with samples from $\nu(\mathbf{x})$ can be very inefficient if few samples are drawn in regions where $|g(\mathbf{x})|$ is large.

Basic Importance Sampling. A more efficient approach than standard MC is to sample instead from a *proposal distribution* $\mathbf{x}^{(p)} \sim \pi(\mathbf{x})$

$$\mathbb{E}_{\nu(\mathbf{x})}[g(\mathbf{x})] = \mathbb{E}_{\pi(\mathbf{x})}\left[\frac{\nu(\mathbf{x})}{\pi(\mathbf{x})} g(\mathbf{x})\right] \approx \sum_{p=1}^{P} w^{(p)}\, g(\mathbf{x}^{(p)}), \tag{2.17}$$

where

$$w^{(p)} = \frac{1}{P} \frac{\nu(\mathbf{x}^{(p)})}{\pi(\mathbf{x}^{(p)})} \tag{2.18}$$

are referred to as *importance-weights*. This IS estimate is unbiased and has minimum variance if $\pi(\mathbf{x}) \propto |g(\mathbf{x})|\nu(\mathbf{x})$ [71]. Although this choice of proposal distribution is never attainable as it would defeat the point of using importance sampling in the first place, it provides a useful guide for choosing an appropriate proposal distribution (e.g. in Ch. 9).

Self-normalised Importance Sampling. If evaluating $\nu(\mathbf{x})$ is difficult, the *unnormalised target distribution* $\tilde{\nu}(\mathbf{x})$ can be used by additionally estimating its normalisation constant $Z = \int \tilde{\nu}(\mathbf{x})d\mathbf{x}$ through IS:

$$\mathbb{E}_{\nu(\mathbf{x})}[g(\mathbf{x})] = \frac{\mathbb{E}_{\pi(\mathbf{x})}\left[\frac{\tilde{\nu}(\mathbf{x})}{\pi(\mathbf{x})} g(\mathbf{x})\right]}{\mathbb{E}_{\pi(\mathbf{x})}\left[\frac{\tilde{\nu}(\mathbf{x})}{\pi(\mathbf{x})}\right]} \approx \frac{\frac{1}{P}\sum_{p=1}^{P} \tilde{w}^{(p)}\, g(\mathbf{x}^{(p)})}{\frac{1}{P}\sum_{p'=1}^{P} \tilde{w}^{(p')}} = \sum_{p=1}^{P} w^{(p)}\, g(\mathbf{x}^{(p)}), \tag{2.19}$$

where the *importance-weights*

$$w^{(p)} = \frac{\tilde{w}^{(p)}}{\sum_{p'=1}^{P} \tilde{w}^{(p')}} \tag{2.20}$$

are *self-normalised* and $\tilde{w}^{(p)} = \frac{\tilde{\nu}(\mathbf{x})}{\pi(\mathbf{x})}$ are the *unnormalised importance-weights*. While both the numerator and denominator in Eq. (2.19) are unbiased, the fraction and thus the resulting estimator is biased for finite sample sizes P. IS with normalisation provides the basis for SMC methods described in Sec. 2.5.

Importance sampling in machine learning. In machine learning applications using IS, $\nu(\mathbf{x})$ is often the posterior $p(\mathbf{x}|\mathbf{y})$ and the unnormalised distribution $\tilde{\nu}(\mathbf{x})$ is the joint distribution $p(\mathbf{x}, \mathbf{y})$. The posterior is thus approximated by the set of importance-weighted particles

$$p(\mathbf{x}|\mathbf{y}) \approx \sum_{p=1}^{P} w^{(p)} \delta(\mathbf{x}^{(p)}), \tag{2.21}$$

where δ denotes the Dirac-delta function. Alternatively, a posterior approximation with uniformly weighted particles can be obtained by treating the importance-weights as the probabilities of a categorical distribution and *resampling* from this distribution.

An unbiased estimate of the marginal likelihood (normalisation constant) is given by

$$\hat{Z} := \frac{1}{P} \sum_{p=1}^{P} \tilde{w}^{(p)} \approx \int p(\mathbf{x}, \mathbf{y}) d\mathbf{x} = p(\mathbf{y}) \tag{2.22}$$

in the denominator of Eq. (2.19). Other scoring rules involving the posterior predictive can be estimated using Eq. (2.19). For instance, the *continuous ranked probability score* [72, 73, 74] is estimated to evaluate the forecasting distribution from the model in Ch. 9.

2.5 Sequential Monte Carlo

Sequential Monte Carlo (SMC) methods [75, 76] are MC methods that combine sequential importance sampling (SIS) and resampling in order to approximate a sequence of *target distributions* $\nu(\mathbf{x}_1), \ldots, \nu(\mathbf{x}_{1:T})$ with increasing dimension [77]. The resulting *target distributions* are of particular interest in themselves: for instance, various versions of particle filters (PFs) and particle smoothers (PSs) are instances of the general class of SMC methods, where e.g. the posterior, the filtering, or the smoothing distribution are used as the *target distribution*. Furthermore, expectations w.r.t. a sequence of *test functions* such as forecasting distributions can be computed sequentially by using the forecast distribution as the *test function* and the posterior or filter distribution as the *target distribution*. The main challenge of SMC is, however, to approximate the sequence of *target distributions*.

Sequential importance sampling. SIS is a recursive version of IS, where the estimates are computed sequentially, building on previous estimates. In contrast to standard IS, the proposal distribution $\pi(\mathbf{x}_{1:T}) = \prod_{t=1}^{T} \pi(\mathbf{x}_t|\mathbf{x}_{1:t-1})$ is chosen with sequential (autoregressive or Markovian) structure. The unnormalised importance-weights can then be computed recursively as

$$\begin{aligned} \tilde{w}_t^{(p)} = \frac{\tilde{\nu}(\mathbf{x}_{1:t}^{(p)})}{\pi(\mathbf{x}_{1:t}^{(p)})} &= \frac{\tilde{\nu}(\mathbf{x}_{1:t-1}^{(p)})}{\pi(\mathbf{x}_{1:t-1}^{(p)})} \frac{\tilde{\nu}(\mathbf{x}_{1:t}^{(p)})/\tilde{\nu}(\mathbf{x}_{1:t-1}^{(p)})}{\pi(\mathbf{x}_t^{(p)}|\mathbf{x}_{1:t-1}^{(p)})} \\ &= \tilde{w}_{t-1}^{(p)}\, \gamma(\mathbf{x}_t^{(p)}, \mathbf{x}_{1:t-1}^{(p)}), \end{aligned} \tag{2.23}$$

where $\gamma(\mathbf{x}_t^{(p)}, \mathbf{x}_{1:t-1}^{(p)})$ are referred to as the *incremental importance-weights*. A sensible choice for the proposal distribution $\pi(\mathbf{x}_t^{(p)}|\mathbf{x}_{1:t-1}^{(p)})$ is one that is proportional to $\tilde{\nu}(\mathbf{x}_{1:t}^{(p)})/\tilde{\nu}(\mathbf{x}_{1:t-1}^{(p)})$ to minimise the variance of the importance-weights. Approximations of the *target distribution* and the normalisation constants at every step t can be estimated analogously to standard IS. The *target distribution* is approximated as

$$\hat{\nu}(\mathbf{x}_{1:t}) := \sum_{p=1}^{P} w_t^{(p)} \delta(\mathbf{x}_{1:t}^{(p)}) \approx \nu(\mathbf{x}_{1:t}). \tag{2.24}$$

An unbiased estimate of the normalisation constant is given by

$$\hat{Z}_t := \frac{1}{P} \sum_{p=1}^{P} \tilde{w}_t^{(p)} \approx \int \tilde{\nu}(\mathbf{x}_{1:t}) = Z_t. \tag{2.25}$$

Sequential Monte Carlo. The major problem of standard SIS is the so-called *weight degeneracy* problem (see e.g. [77]): the maximum importance-weight quickly approach one and all other weights zero for increasing steps t, because IS/SIS scales poorly with the number of dimensions. The novelty of SMC compared to SIS is a *resampling* procedure that discards/duplicates particles that have low/high importance-weights, focusing computational resources on

the most relevant particles. The resampling procedure can be seen as an improvement upon standard SIS by sampling from

$$\mathbf{x}_{1:t}^{(p)} \sim \pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(p)})\hat{\nu}(\mathbf{x}_{1:t-1}) \tag{2.26}$$

rather than $\pi(\mathbf{x}_t|\mathbf{x}_{1:t-1}^{(p)})\pi(\mathbf{x}_{1:t-1}^{(p)})$. SMC thus takes advantage of the information provided in the (sample-based approximation of the) previous target distribution [78]

$$\hat{\nu}(\mathbf{x}_{1:t-1}) = \sum_{p=1}^{P} w_{t-1}^{(p)} \delta(\mathbf{x}_{1:t-1}^{(p)}). \tag{2.27}$$

Sampling from Eq. (2.26) is achieved in three steps:

1. resampling $\bar{\mathbf{x}}_{1:t-1}^{(p)} \sim \hat{\nu}(\mathbf{x}_{1:t-1})$,
2. sampling $\mathbf{x}_t^{(p)} \sim \pi(\mathbf{x}_t|\bar{\mathbf{x}}_{1:t-1}^{(p)})$,
3. concatenating $\mathbf{x}_{1:t}^{(p)} := (\mathbf{x}_t^{(p)}, \bar{\mathbf{x}}_{1:t-1}^{(p)})$.

If a resampling step is performed, the importance-weights are set uniformly as $w_t^{(p)} = 1/P$.

In order to resample from the discrete distribution over particle indices defined by the importance-weights, the weights must be normalised. It is thus useful to write the unnormalised importance-weights in SMC using the previous *normalised* importance-weights. Starting with $w_{t-1}^{(p)} = 1/P$ for $t = 1$, the unnormalised importance-weights then are

$$\tilde{w}_t^{(p)} = w_{t-1}^{(p)} \gamma(\mathbf{x}_t^{(p)}, \mathbf{x}_{1:t-1}^{(p)}), \qquad w_t^{(p)} = \frac{\tilde{w}_t^{(p)}}{\sum_{p'=1}^{P} \tilde{w}_t^{(p')}}, \tag{2.28}$$

where the *incremental importance-weights* $\gamma(\mathbf{x}_t^{(p)}, \mathbf{x}_{1:t-1}^{(p)})$ are given by Eq. (2.23).

The particle approximation of the *target distribution* at every step t is analogous to SIS, though an approximation before and after resampling, respectively, is given by

$$\hat{\nu}(\mathbf{x}_{1:t}) = \sum_{p=1}^{P} w_t^{(p)} \delta(\mathbf{x}_{1:t}^{(p)}), \tag{2.29a}$$

and

$$\bar{\nu}(\mathbf{x}_{1:t}) = \sum_{p=1}^{P} w_t^{(p)} \delta(\bar{\mathbf{x}}_{1:t}^{(p)}). \tag{2.29b}$$

Due to the normalisation, the sum of unnormalised importance-weights provides an estimate of the ratio of normalisation constants $Z_{t|t-1} = \frac{Z_t}{Z_{t-1}}$, i.e.

$$\hat{Z}_{t|t-1} = \sum_{p=1}^{P} \tilde{w}_t^{(p)} \approx Z_{t|t-1}. \tag{2.30}$$

This estimate is biased but consistent; however, the product of the estimates of these conditionals yields an unbiased estimate of the joint likelihood $\hat{Z}_t = \prod_{t=1}^{T} \hat{Z}_{t|t-1}$ [78, 79].

Resampling strategies. Reducing the *weight degeneracy* problem through resampling comes at the cost of introducing the so-called *path degeneracy* problem. Both *degeneracies* arise from the fact that IS is inefficient in high-dimensional spaces. The resampling procedure adds sampling noise [80] and causes most future particles to share the same ancestors; it thereby reduces the quality of the approximation of the past trajectory, while (in most cases) improving the approximation of future target densities [77]. Consequently, $\hat{\nu}$ should be preferred for the current estimate while $\bar{\nu}$ is in most cases preferable for future estimates. Two common approaches are often applied to reduce the *path degeneracy* problem (used e.g. in Ch. 9). One way to reduce the variance is to improve the resampling procedure. The simplest algorithm, known as *multinomial resampling*, treats the importance-weights as probabilities of a discrete distribution and samples the particle indices from this distribution with replacement. Alternative algorithms such as *systematic resampling* are known to yield favorable results in practice [81]. Moreover, *adaptive resampling* performs the resampling step only if a criterion is met, e.g. if the *effective sample size*

$$P_{\text{eff}} = \left(\sum_{p=1}^{P} \left(w_t^{(p)} \right)^2 \right)^{-1} \tag{2.31}$$

drops below a defined threshold such as $P_{\text{eff}} \leq P/2$. If resampling is performed, $w_{t-1}^{(p)} = 1/P$; otherwise, the unnormalised importance-weights are updated according to Eq. (2.28). This resampling criterion is used together with systematic resampling in the inference method proposed in Ch. 9.

3 Bayesian Neural Networks

The general methodology from Sec. 2.2 applies to parametric models, including deep NN models (see Sec. 3.1 for the model formulation). The simplest approximate-inference methods for learning the parameters are described in Sec. 3.2 and 3.3, and Sec. 3.4 shows how variational Bayes (cf. Sec. 2.3) can be used for learning BNNs.

3.1 Model Formulation

Deep Neural Networks. The most basic type of deep NNs is also referred to as multilayer perceptron (MLP). The parametric non-linear models progressively transform input vectors by applying non-linear transformations. Denoting feature vectors by $\mathbf{h}_l$ and the input vector by $\mathbf{u}$, a deep MLPs computes the function

$$\text{mlp}(\mathbf{u}) = a_L(\mathbf{W}_L\mathbf{h}_{L-1} + \mathbf{b}_L), \quad \mathbf{h}_1 = a_1(\mathbf{W}_1\mathbf{u} + \mathbf{b}_1), \quad \mathbf{h}_l = a_l(\mathbf{W}_l\mathbf{h}_{l-1} + \mathbf{b}_l), \tag{3.1}$$

where $\mathbf{W}_l$ and $\mathbf{b}_l$ denote the weight matrix and bias vector corresponding to layer l, respectively, and a_l are element-wise activation functions that introduce non-linearities. The last activation function a_L is dictated by the prediction task. For instance, in regression with additive homogeneous Gaussian noise, the real-valued range of the predictions is usually not constrained, so a_L is the identity function. For multi-class classification, the outputs are constrained to positive values summing to one (probabilities), so a_L is a softmax function. The collection of all weight matrices and bias vectors, stacked as a parameter vector, is denoted by θ.

Other NN architectures that are commonly used e.g. for time-series or image data are recurrent neural networks (RNNs) and convolutional neural networks (CNNs). And various more intricate NN architectures have shown superior performance in practice, including residual networks [82], recurrent models with internal memory and a gating mechanism [83, 84], or transformers with an attention mechanism [85]. An overview of deep NN models can be found e.g. in [10, 9].

Bayesian Deep Neural Networks. The parameters of NNs are usually learnt via ML estimation through optimisation with SGD based optimisation algorithms. If the NN parameters are instead treated as hidden random variables that must be inferred from data through Bayesian inference, these models are referred to as Bayesian neural networks (BNNs). As mentioned in Ch. 2, Bayesian inference is computationally intractable for most model classes—including BNNs—and approximations are required. A BNN assigns the marginal likelihood

$$p(\mathcal{D}) = \int p(\mathcal{D}|\theta)p(\theta)d\theta \tag{3.2}$$

to a dataset $\mathcal{D} = \{\mathbf{u}^{(n)}, \mathbf{y}^{(n)}\}_{n=1}^{N}$ consisting of N inputs $\mathbf{u}^{(n)}$ and targets $\mathbf{y}^{(n)}$. A prior $p(\theta)$ is assumed over the NN weights and biases denoted by θ, and the likelihood function factorises as $p(\mathcal{D}|\theta) = \prod_{n=1}^{N} p(\mathbf{y}^{(n)}|\mathbf{u}^{(n)}, \theta)$.

In the context of BNNs, the posterior approximation will be denoted by

$$q_\phi(\theta) \approx \frac{p(\mathcal{D}|\theta)p(\theta)}{Z}, \tag{3.3}$$

where ϕ denotes the parameters of the approximating distribution such as the mean and log-scale if it is Gaussian. The following sections introduce relevant methods to approximate ϕ. Other important inference algorithms based on MCMC (e.g. [86, 87, 88, 89]) or Langevin dynamics (e.g. [22]) are not in the scope of this dissertation.

3.2 Maximum a Posteriori Approximation

The simplest approximation is a Dirac delta distribution $q_\phi(\theta) = \delta(\theta - \theta_{\text{MAP}})$, which approximates the full posterior distribution by the single most probable parameters, i.e. the MAP point estimate $\phi = \theta_{\text{MAP}}$. Though point estimates are clearly bad approximations for distributions, this approach can be justified in practice: if a lot of data (evidence) is available in comparison to the (unknown) model parameters, it can be reasonably expected that the posterior has very low entropy and will concentrate in the limit. As a benefit, the intractable integration is replaced by an optimisation problem that can be dealt with e.g. by gradient methods:

$$\begin{aligned} \theta_{\text{MAP}} &= \operatorname*{argmax}_\theta \log\, p(\theta|\mathcal{D}) \\ &= \operatorname*{argmax}_\theta \log\, p(\mathcal{D}|\theta) + \log p(\theta). \end{aligned} \tag{3.4}$$

It is well known that optimising the negative log marginal likelihood (w.r.t. the parameters) with additional L2/L1 regularisation is equivalent to MAP estimation with Gaussian/Laplace priors, respectively, and that ML estimation without regularisation corresponds to a uniform prior.

3.3 Laplace Approximation

The Laplace approximation provides a simple extension and improvement over MAP estimation. The posterior distribution is approximated locally around the MAP estimate by a Gaussian distribution $q_\phi(\theta) = \mathcal{N}\big(\theta;\, \mathbf{m}, \mathbf{V}\big)$, where the parameters of the approximating distribution are $\phi = \{\mathbf{m}, \mathbf{V}\}$. This is achieved through second order Taylor series expansion of the log-posterior around θ_{MAP}. In the first step, $\mathbf{m} = \theta_{\text{MAP}}$ is computed as in Sec. 3.2. Secondly, the Hessian $\mathbf{H}$ is computed around $\mathbf{m}$, yielding the covariance matrix parameters $\mathbf{V} = \mathbf{H}^{-1}$. To show this, it suffices to use the unnormalised posterior with possibly unknown normalisation constant. Expanding $\log f(\theta)$, where $f(\theta) \propto p(\theta|\mathcal{D})$, around $\mathbf{m}$ up to the second term results in

$$\log f(\theta) \approx \log f(\mathbf{m}) - \frac{1}{2}(\theta - \mathbf{m})^T\, \mathbf{H}\, (\theta - \mathbf{m}). \tag{3.5}$$

The first term in Eq. (3.5) is the zeroth-order term (offset) of the Taylor series. The first order term (gradient) is zero at a local optimum, $\mathbf{m} = \theta_{\text{MAP}}$. And the second term in Eq. (3.5) is the second-order term of the Taylor series, where

$$\mathbf{H} = \left. \frac{\partial^2 \log f(\theta)}{\partial(\theta)\partial(\theta)} \right|_{\theta=\mathbf{m}}$$

is the Hessian computed at $\mathbf{m}$. The quadratic form of Eq. (3.5) implies that

$$f(\theta) \approx f(\mathbf{m}) \exp\Big\{ -\frac{1}{2}\big(\theta - \mathbf{m}\big)^T \mathbf{H} \big(\theta - \mathbf{m}\big) \Big\}$$

is a Gaussian function. Subsequent normalisation thus yields the Gaussian approximation

$$q_\phi(\theta) = \mathcal{N}\big(\theta;\, \mathbf{m}, \mathbf{V}\big); \quad \mathbf{m} = \theta_{\text{MAP}}, \quad \mathbf{V} = \mathbf{H}^{-1}.$$

Computing the Hessian exactly has a computational complexity of $O(|\mathcal{D}||\phi|^2)$ that is linear in dataset size and quadratic in number of parameters. This is prohibitive for models with a large amount of parameters such as NNs. For this reason, diagonal or structured approximations of the Hessian are typically used in practice [90, 91, 92]. Unfortunately, diagonal approximations are known to result in bad posterior predictive distributions in case of BNNs [17, 93]; linearising the NN has been shown to provide better results [94].

3.4 Variational-Bayesian Approximation

Variational-Bayesian methods introduced in Sec. 2.3 can be applied to infer the weights and biases of NNs, by assuming a fixed parametric form for q_ϕ with *variational parameters* ϕ and building on MC-based gradient estimation with reparametrisation. Replacing notation for latent random variables $\mathbf{x}$ used in 2.3 by θ denoting the unknown model parameters, the ELBO for BNNs is given by

$$\begin{aligned} \hat{\mathcal{L}}_{\text{ELBO}}\left(\mathcal{D}, \phi\right) &= \frac{1}{P} \sum_{p=1}^{P} \log p(\mathcal{D}|\theta^{(p)}) + \log p(\theta^{(p)}) - \log q_\phi(\theta^{(p)}) \\ &= \frac{1}{P} \sum_{p=1}^{P} \left(\frac{N}{B} \sum_{n=1}^{B} \log p(\mathbf{y}^{(n)}|\mathbf{u}^{(n)}, \theta^{(p)}) + \log p(\theta^{(p)}) - \log q_\phi(\theta^{(p)}) \right), \end{aligned} \tag{3.6}$$

where $\theta^{(p)} \sim q_\phi(\theta)$ is sampled from a variational distribution for which *reparametrisation* is applicable. It is often possible to compute the KL in the ELBO in closed-form and thus obtain lower variance estimates, e.g. if both densities are from the exponential family [70]:

$$\hat{\mathcal{L}}_{\text{ELBO}}\left(\mathcal{D}, \phi\right) = \frac{1}{P} \frac{N}{B} \sum_{p=1}^{P} \sum_{n=1}^{B} \log p(\mathbf{y}^{(n)}|\mathbf{u}^{(n)}, \theta^{(p)}) + \text{KL}\big[q_\phi(\theta)\,||\,p(\theta)\big]. \tag{3.7}$$

The most common choice is an independent Normal *variational distribution* $q_\phi(\theta) = \mathcal{N}(\theta;\, \mathbf{m}, \mathbf{V})$, where the means and log-scales are parametrised, i.e. the variational parameters are $\phi = \{\mathbf{m}, \boldsymbol{\rho}\}$,

and the diagonal covariance matrix is given by $\mathbf{V} = \text{diag}(\exp(2 \cdot \boldsymbol{\rho}))$, where $\text{diag}(\cdot)$ denotes the diagonal matrix constructed from a vector. In practice, it is also common to replace the exponential by a softplus function.

The sampling operation can be *reparametrised* as in 2.15. For posterior approximations with diagonal (fully factorised) covariance [95] or low-rank plus diagonal covariance structure [96] in MLPs, *local reparametrisation* can be applied, providing MC (gradient) estimates with much lower variance. This reparametrisation exploits the fact that the linear transformation with Gaussian weights induces Gaussian *pre-activations* (before the non-linearity) that can be computed in closed-form; sampling from these pre-activations instead of the weights directly provides gradient estimates with significantly lower variance.

4 Deep Latent-Variable Models

The goal of probabilistic generative modelling is to approximate the generative process underlying a dataset $\mathcal{D} = \{\mathbf{y}^{(n)}\}_{n=1}^N$. This is often achieved by means of additional latent (unobserved) variables. The so-called graphical model describes the conditional independence assumptions by which latent and observed variables are related. These conditional independence assumptions and modelling choices for the respective distributions can be used to inject prior knowledge about the true model. The latent variables can be thought of as modelling additional factors that are not observed but contribute to the data generating process [97]. For instance, latent variables can represent the common state underlying multiple observations (Ch. 8), or the structured state of a dynamical system (Ch. 9).

4.1 Model Formulation

One particular class of probabilistic models that is relevant to this dissertation are latent-variable models (LVMs) and their extensions using deep neural networks, which are here referred to as deep latent-variable models (DLVMs).

Latent-Variable Models. LVMs describe the generative process by modelling that each observation $\mathbf{y}^{(n)}$ is associated with its own *local* latent variable $\mathbf{x}^{(n)}$ via the joint distribution

$$p_\theta(\mathbf{y}^{(n)}, \mathbf{x}^{(n)}) = p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})p_\theta(\mathbf{x}^{(n)}), \tag{4.1}$$

and assuming that observations are conditionally independent given the parameters θ, i.e.

$$p_\theta(\mathcal{D}) = \prod_{n=1}^N p_\theta(\mathbf{y}^{(n)}), \tag{4.2}$$

where $p_\theta(\mathbf{y}^{(n)}) = \int p_\theta(\mathbf{y}^{(n)}, \mathbf{x}^{(n)}) d\mathbf{x}^{(n)}$. While both the parameters θ and latent variables $\mathbf{x}^{(n)}$ can in principle be inferred via the Bayesian probabilistic approach, it is common to use (approximate) inference for the posterior over the latent variables and ML estimation for the parameters. In contrast to Ch. 3—where the parameters are treated as random variables—the NN parameters are here denoted by a subscript.

This model class encompass several well-known models such as factor analysis [98, 99], probabilistic principal component analysis (PCA) [100, 101], probabilistic independent component analysis (ICA) [102], Gaussian mixture models, latent Dirichlet allocation [103], sigmoid belief nets [104, 105], or the Helmholtz machine [106].

Deep Latent-Variable Models. DLVMs are LVMs in which the conditional distributions of the graphical model are parametrised by deep NNs and the latent variables have hierarchical structure of the form

$$p_\theta(\mathbf{y}^{(n)}, \mathbf{x}^{(n)}) = p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})p_\theta(\mathbf{x}^{(n)}), \tag{4.3a}$$

$$p_\theta(\mathbf{x}^{(n)}) = p_\theta(\mathbf{x}_1^{(n)}) \prod_{l=2}^{L} p_\theta(\mathbf{x}_l^{(n)}|\mathbf{x}_{<l}^{(n)}), \tag{4.3b}$$

where $\mathbf{x}^{(n)} = \{\mathbf{x}_l^{(n)}\}_{l=1}^L$ are the latent variables associated with the observations $\mathbf{y}^{(n)}$. The likelihood (Eq. (4.3a)) and the hierarchical prior (Eq. (4.3b)) of this generative model often has Markovian structure (e.g. [27, 107, 33]), although recent models with many hierarchies use the more general non-Markovian model (e.g. [108, 109]). Furthermore, several related important models such as conditional variational autoencoders (VAEs) [110] or semi-supervised models [111] can be obtained by using additional inputs $\mathbf{u}^{(n)}$.

4.2 Neural Variational Inference for DLVMs

From a Bayesian perspective, both the parameters and latent variables should be inferred by updating prior information with evidence induced by the likelihood model and data. In practice, it is often sufficient to treat only the latent variables probabilistically, while estimating the parameters through MAP or ML estimation. In case of models for which the posterior can be computed exactly—such as Factory Analysis, Gaussian Mixture models, Gaussian Linear Dynamical Systems—this can be achieved using the expectation maximisation (EM) algorithm [112]. In more complex models, the posterior can be approximated using variational-Bayesian methods (cf. Sec. 2.3). The resulting algorithm is referred to as variational expectation maximisation (VEM) and is equivalent to variational inference; it can thus be seen as a special case of variational-Bayesian methods applied to LVMs, $p_\theta(\mathbf{y}) = \int p_\theta(\mathbf{y}|\mathbf{x})p(\mathbf{x})d\mathbf{x}$, where the variational approach is applied only to the latent variables $\mathbf{x}$, and the parameters θ are approximated by the ML or MAP estimate.[1]

Building on MC approximation of the ELBO and the *reparametrisation* gradient explained in Sec. 2.3, the ELBO estimate for DLVMs is given as

$$\hat{\mathcal{L}}_{\text{ELBO}}\left(\mathcal{D}, \phi, \theta\right) \approx \sum_{n=1}^{N} \frac{1}{P} \sum_{p=1}^{P} \left[\log \frac{p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n,p)})\, p_\theta(\mathbf{x}^{(n,p)})}{q_{\phi_n}(\mathbf{x}^{(n,p)})} \right], \quad \mathbf{x}^{(n,p)} \sim q_{\phi_n}(\mathbf{x}^{(n)}). \tag{4.4}$$

As discussed in Sec. 2.3, the ELBO is often written in KL form (Eq. (2.7b)), and the KL can be computed in closed-form e.g. for distributions from the exponential family [70].

[1] The variational approach can be applied to all unknown variables of the model including parameters; some authors refer to the corresponding variational algorithm as variational-Bayesian EM [113]. In DLVMs, this is indeed possible, combining Sec. 3.4 and the approach described in this chapter.

4.2.1 Variational Autoencoder

The above approach does not scale to large datasets for two reasons: First, the ELBO is computed by summing the per-data-point ELBOs over the whole dataset; this is possible since $\hat{\mathcal{L}}_{\text{ELBO}}\left(\mathcal{D}, \phi, \theta\right) = \sum_{n=1}^{N} \hat{\mathcal{L}}_{\text{ELBO}}\left(\mathbf{y}^{(n)}, \phi, \theta\right)$ in DLVMs that makes the i.i.d. assumption $\log p_\theta(\mathcal{D}) = \sum_{n=1}^{N} \log p_\theta(\mathbf{y})$. Moreover, every per-data-point posterior $p_\theta(\mathbf{x}^{(n)}|\mathbf{y}^{(n)})$ is approximated with independent *variational parameters* $\phi = \{\phi_n\}_{n=1}^{N}$. These issues can be addressed by mini-batch MC estimation and *amortising* the per-data-point inference through shared variational parameters [25]. To this end, the variational autoencoder (VAE) [26, 27] uses an encoder NN to predict the parameters of the approximating distribution using the data as input. This approach is referred to as *amortised* or *neural variational inference*.

The variational parameters ϕ are shared for each data point, that is, $q_\phi(\mathbf{x}^{(n)}|\mathbf{y}^{(n)}) \approx p_\theta(\mathbf{x}^{(n)}|\mathbf{y}^{(n)})$. Consequently, assuming the dataset is randomly shuffled, the ELBO estimate becomes

$$\hat{\mathcal{L}}^{\text{VAE}}_{\text{ELBO}}\left(\mathcal{D}, \phi, \theta\right) \approx \frac{N}{B} \sum_{n=1}^{B} \frac{1}{P} \sum_{p=1}^{P} \left[\log w^{(n,p)} \right], \quad \mathbf{x}^{(n,p)} \sim q_\phi(\mathbf{x}^{(n)}|\mathbf{y}^{(n)}), \tag{4.5}$$

where

$$w^{(n,p)} = \frac{p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n,p)})\, p_\theta(\mathbf{x}^{(n,p)})}{q_\phi(\mathbf{x}^{(n,p)}|\mathbf{y}^{(n)})}. \tag{4.6}$$

In practice, it is usually sufficient to draw a single sample (i.e. $P = 1$) from the variational distribution, since the variance of this estimate can also be reduced by taking larger mini-batch sizes B.

4.2.2 Importance-weighted Autoencoder

$\hat{\mathcal{L}}^{\text{VAE}}_{\text{ELBO}}$ can be seen as a special case of a tighter importance-sampling ELBO, $\hat{\mathcal{L}}^{\text{IWAE}}_{\text{ELBO}}$, with importance-weights $w^{(n,p)}$ (cf. Sec. 2.4). Generalising the VAE, this approach is referred to as importance-weighted autoencoder (IWAE) [33]. It provides a tighter bound

$$\hat{\mathcal{L}}^{\text{IWAE}}_{\text{ELBO}}\left(\mathcal{D}, \phi, \theta\right) \approx \frac{N}{B} \sum_{n=1}^{B} \left[\log \frac{1}{P} \sum_{p=1}^{P} w^{(n,p)} \right], \quad \mathbf{x}^{(n,p)} \sim q_\phi(\mathbf{x}^{(n)}|\mathbf{y}^{(n)}). \tag{4.7}$$

The bound becomes tight as $P \to \infty$ (if $w^{(n,p)}$ is bounded). The variational distribution here serves the purpose of a proposal distribution and is usually chosen as a factorised Gaussian. Performing a *resampling* operation on the importance-sampling approximation of the posterior $p_\theta(\mathbf{x}^{(n)}|\mathbf{y}^{(n)}) \approx \sum_{p=1}^{P} w^{(n,p)} \delta(\mathbf{x}^{(n,p)})$ results in samples from a non-Gaussian posterior approximation that approximates the true posterior as $P \to \infty$.

An important property of the variational importance-sampling formulation that is used in Ch. 8 is the following: Even if the amortisation is chosen such that parameters of the proposal distribution are predicted from incomplete information, e.g. $q_\phi(\mathbf{x}|\mathbf{y}_1) \approx p_\theta(\mathbf{x}|\mathbf{y}_1, \mathbf{y}_2)$, samples from the true posterior can be recovered as $P \to \infty$. This is because (feedback from) the likelihood is used to update the incomplete encoder posterior approximation, reflected in the

importance-weights. In principle, the proposal could even be the prior, though this would of course be very inefficient. In contrast, the standard variational approximation used in Eq. (4.5)—and even very flexible posterior approximations based on normalising flows (e.g. [35, 114, 115, 116])—cannot approximate the true posterior to arbitrary accuracy through amortisation from incomplete information.

4.2.3 Hierarchical Variational Approximation

The previous paragraphs thus far did not discuss the hierarchical nature of the DLVMs for depths $L > 1$. The true posterior factorises in its forward form simply proportional to joint distribution,

$$p_\theta(\mathbf{x}|\mathbf{y}) \propto p_\theta(\mathbf{y}|\mathbf{x}) \prod_{l=1}^{L} p_\theta(\mathbf{x}_l|\mathbf{x}_{<l}), \tag{4.8}$$

where the superscript index n is dropped in this section, and $p_\theta(\mathbf{x}_l|\mathbf{x}_{<l}) = p_\theta(\mathbf{x}_1)$ for $l = 1$. Variational approximations often do not follow the true posterior factorisation for practical reasons such as efficiency or even tractability. With amortised/neural variational inference, the variational distribution can be factorised into several conditional distributions predicted by NNs. Two common factorisations are worth distinguishing, the latter being used e.g. in Ch. 9.

Backwards Factorisation. Since the encoder takes the data as inputs—in the opposite direction as the likelihood model—one obvious approach is to factorise the variational distribution in the reverse direction compared to the prior and likelihood model:

$$q_\phi^{\mathrm{bw}}(\mathbf{x}|\mathbf{y}) = q_\phi(\mathbf{x}_L|\mathbf{y}) \prod_{l=1}^{L-1} q_\phi(\mathbf{x}_l|\mathbf{x}_{>l}). \tag{4.9}$$

The encoder NN can thus be seen as the function inverting the model. This approach is taken e.g. in [33], although a Markov assumption is made both for the generative and inference model.

Forward Factorisation. Alternatively, information from the *forward* model can be combined with an approximate likelihood, which is computed recursively in the *backward* direction. This approach reuses the conditional Gaussians $p_\theta(\mathbf{x}_l|\mathbf{x}_{<l})$ of the forward model for its posterior approximation, while only approximating the likelihood by a Gaussian, similar in structure to Eq. (4.8). To this end, the Gaussian likelihood approximation $h_\phi(\mathbf{x}_l|\mathbf{y})$ is predicted by an encoder NN for every hierarchy of latent variables. That is, the variational approximation is

$$q_\phi^{\mathrm{fw}}(\mathbf{x}|\mathbf{y}) = \prod_{l=1}^{L} \frac{1}{Z_l} h_\phi(\mathbf{x}_l|\mathbf{y}) p_\theta(\mathbf{x}_l|\mathbf{x}_{<l}), \tag{4.10}$$

where Z_l is the normalisation constant of the respective product of Gaussian. The encoder NN consists of a hierarchy of deterministic NNs f_{ϕ_l} running in the backwards direction

$$\begin{aligned} \mathbf{d}_L &= f_{\phi_L}(\mathbf{y}), \\ \mathbf{d}_l &= f_{\phi_l}(\mathbf{d}_{l+1}),\ l = 1, \ldots, L-1. \end{aligned} \tag{4.11}$$

A further transformation then maps these deterministic representations $\mathbf{d}_l$ to the distribution parameters $\mathbf{m}_l(\mathbf{y}, \phi)$ and $\mathbf{V}_l(\mathbf{y}, \phi)$:

$$h_\phi(\mathbf{x}_l|\mathbf{y}) := g_\phi(\mathbf{d}_l). \tag{4.12}$$

This factorisation was proposed e.g. in the ladder VAE [107], although a Markov assumption is made for both the generative and inference model. Furthermore, this approach is used for the variational proposal distributions for the model in Ch. 9.

5 Deep State-Space Models

Many practical problems involve one or several sequences of time-ordered observations. Such time series data does not permit the independence assumption between observations within the same time series. The temporal dependency can be modelled explicitly with probabilistic time series models: for instance, autoregressive models directly model the (forward) conditional distributions of the observations given the previous observations; on the other hand, state-space models (SSMs) use additional latent variables to model the time series data.

Sec. 5.1 defines SSMs, and Sec. 5.2 then describes the inference and parameter estimation problems arising in these models and general formulations to address these problems. For the standard Gaussian linear dynamical system (GLS), exact solutions for the inference problems and methods for parameter estimation are presented in Sec. 5.3. Variational SMC methods for state estimation and parameter learning in deep state-space model (DSSM) are then described in Sec. 5.4. These methods provide the basis for the hybrid linear/non-linear model proposed in Ch. 9.

5.1 Model Formulation

SSMs—often referred to as Hidden Markov Models in case of discrete states—are LVMs with Markovian sequential structure.

$$p_\theta(\mathbf{y}_{1:T}, \mathbf{x}_{0:T}) = p_\theta(\mathbf{x}_0) \prod_{t=1}^{T} p_\theta(\mathbf{y}_t|\mathbf{x}_t) p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}), \tag{5.1a}$$

where $p_\theta(\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ is the *initial prior* distribution and θ denotes the model parameters. The subscript t is a discrete time index for the elements in the sequence of latent variables $\mathbf{x}_{0:T}$ and observations $\mathbf{y}_{1:T}$. Both the *dynamical model* $p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})$ and *measurement/emission model*[1] $p_\theta(\mathbf{y}_t|\mathbf{x}_t)$ are given by a possibly non-linear transformation and additive Gaussian noise

$$\mathbf{x}_t = f_\theta(\mathbf{x}_{t-1}) + \mathbf{w}_t, \qquad \mathbf{w}_t \sim \mathcal{N}\big(\mathbf{w}_t; \mathbf{0}, \mathbf{R}_\theta(\mathbf{x}_{t-1})\big). \tag{5.1b}$$

$$\mathbf{y}_t = g_\theta(\mathbf{x}_t) + \mathbf{v}_t, \qquad \mathbf{v}_t \sim \mathcal{N}\big(\mathbf{v}_t; \mathbf{0}, \mathbf{Q}_\theta(\mathbf{x}_{t-1})\big). \tag{5.1c}$$

If the conditional distributions are parametrised by NNs, these models are referred to as deep state-space models (DSSMs) in this dissertation.

[1] Depending on the perspective, i.e. for performing inference or generation, the respective term is more common.

5.2 Inference and Parameter Estimation

This section describes common inference and parameter estimation problems for SSMs and DSSMs. If the conditional distributions are linear Gaussian, efficient recursive algorithms for *exact* inference can be used (see Sec. 5.3). However, non-linear dynamical and measurement models are more important in practice, requiring *approximate*-inference algorithms such as SMC approximations (see Sec. 5.4). The variational SMC methods for DSSMs and the analytic solutions from the GLS provide the basis for the *hybrid* inference algorithm in Ch. 9.

The Markov assumption for the dynamical and emission model model has several useful consequences for inference algorithms due to the following implications [117]:

$$p_\theta(\mathbf{y}_t|\mathbf{x}_{0:t}, \mathbf{y}_{1:t-1}) = p_\theta(\mathbf{y}_t|\mathbf{x}_t), \tag{5.2a}$$
$$p_\theta(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{y}_{1:t-1}) = p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}), \tag{5.2b}$$
$$p_\theta(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{y}_{1:T}) = p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_{t:T}), \tag{5.2c}$$
$$p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1:T}, \mathbf{y}_{1:T}) = p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y}_{1:t}). \tag{5.2d}$$

Several inference problems of interest—such as state inference, forecasting, and parameter estimation—benefit from these simplifications, as explained in the following.

5.2.1 Inference and Prediction

In the sequential inference setting, several distributions besides the *posterior* $p_\theta(\mathbf{x}_{0:T}|\mathbf{y}_{1:T})$ are important in practical applications. The respective algorithms to compute these are summarised in this section and general approaches are explained. Algorithms for exact solutions in case of Gaussian linear models as well as variational SMC approximations for DSSMs are provided in Sec. 5.3 and Sec. 5.4, respectively.

Filtering. The *filter distribution* is the posterior of the state at a certain times-slice given the data up to that time-slice. It can be computed recursively from the previous filter distribution using the Markov assumptions of Eqs. (5.2a) and (5.2b):

$$p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t}) \propto p_\theta(\mathbf{y}_t|\mathbf{x}_t) \int p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}) p_\theta(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) d\mathbf{x}_{t-1}, \tag{5.3}$$

where $p_\theta(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) = p_\theta(\mathbf{x}_0)$ for $t = 1$. The resulting filtering algorithm is an online algorithm that uses results from the previous step and data up to the respective time index t only. The marginalisation (integral) is referred to as *prediction step* and the Bayes update (multiplication with the likelihood $p_\theta(\mathbf{y}_t|\mathbf{x}_t)$ and conditioning on $\mathbf{y}_t$) is referred to as *update step*.

Smoothing. In offline scenarios, the *smoothing distribution* $p_\theta(\mathbf{x}_t|\mathbf{y}_{1:T})$ and the *full posterior* $p_\theta(\mathbf{x}_{0:T}|\mathbf{y}_{1:T})$ can be inferred given all available observations including future observations $\mathbf{y}_{t:T}$. Several variants to compute these distributions have been proposed in the literature for different types of approximations such as sigma-point and particle approximations. These

general smoothing variants include *forward-backward*, *backward-forward*, *two-filter*, and even *forward-only* formulations.

Two-filter smoothers [118, 119] compute a forward and backward filter message and combine them at every time-slice t to form $p_\theta(\mathbf{x}_t|\mathbf{y}_{1:T})$. The disadvantage of this approach is that it provides only the smoothing distribution (marginals) but not the full posterior. Furthermore, artificial marginals distributions have to be defined for every time-slice in order to make the backward message normalisable [119].

Forward-backward / backward-forward smoothers first filter forward/backward, and then recursively compute the smoothing distributions backward/forward, respectively; Interestingly, for certain models, the forward-backward variant can be reformulated as a *forward-only* approach, which is especially suitable for online applications. For examples of the backward-forward and forward-only variants, see e.g. [120, 121, 122, 123]; these are not further discussed here.

The *forward-backward* smoother is the most common variant; it is described in more detail in the following. After the last filter distribution $p_\theta(\mathbf{x}_T|\mathbf{y}_{1:T})$ (identical to last smoothing distribution) is computed, the method recurses backwards, smoothing the results from the filter:

$$p_\theta(\mathbf{x}_t|\mathbf{y}_{1:T}) = p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t}) \int \frac{p_\theta(\mathbf{x}_{t+1}|\mathbf{x}_t) p_\theta(\mathbf{x}_{t+1}|\mathbf{y}_{1:T})}{p_\theta(\mathbf{x}_{t+1}|\mathbf{y}_{1:t})} d\mathbf{x}_{t+1}. \tag{5.4a}$$

This is obtained by marginalising the future states $\mathbf{x}_{t+1}$ in the two-slice marginal posterior

$$p_\theta(\mathbf{x}_t|\mathbf{y}_{1:T}) = \int p_\theta(\mathbf{x}_t, \mathbf{x}_{t+1}|\mathbf{y}_{1:T}) d\mathbf{x}_{t+1}, \tag{5.4b}$$

where the Markov assumption of Eq. (5.2d) is used to factorise this two-slice distribution as

$$p_\theta(\mathbf{x}_t, \mathbf{x}_{t+1}|\mathbf{y}_{1:T}) = p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y}_{1:t}) p_\theta(\mathbf{x}_{t+1}|\mathbf{y}_{1:T}), \tag{5.4c}$$

and where the Markov property of Eq. (5.2b) simplifies the posterior backward transition as

$$p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y}_{1:t}) = \frac{p_\theta(\mathbf{x}_{t+1}|\mathbf{x}_t) p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t})}{p_\theta(\mathbf{x}_{t+1}|\mathbf{y}_{1:t})}. \tag{5.4d}$$

The initial distribution is computed using $p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t}) = p_\theta(\mathbf{x}_0)$ for $t = 0$.

In order to compute the marginals in Eq. (5.4b), the backward pass additionally yields two-slice marginals and thus even the *full posterior* through $p_\theta(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}) = \prod_{t=0}^{T} p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y}_{1:t})$, where for $t = 0$ and $t = T$ the respective observed and future state variables are omitted. This latter formulation of the full posterior is useful also for algorithms that aim to sample only from the posterior; for example, particle implementations of forward filtering backward sampling have lower computation complexity compared to approximating the smoothing marginals.

Forecasting. Using the results from the filter distribution, future observations can be forecasted by marginalising the filtered and predicted states:

$$\begin{aligned} p_\theta(\mathbf{y}_{t+1:T}|\mathbf{y}_{1:t}) &= \int p_\theta(\mathbf{y}_{t+1:T}|\mathbf{x}_t) p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t}) d\mathbf{x}_t \\ &= \int p_\theta(\mathbf{y}_{t+1:T}|\mathbf{x}_{t+1:T}) \int p_\theta(\mathbf{x}_{t+1:T}|\mathbf{x}_t) p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t}) d\mathbf{x}_t \, d\mathbf{x}_{t+1:T}. \end{aligned} \tag{5.5}$$

5.2.2 Parameter Estimation

Several flexible approximate-inference methods such as MCMC and VB can be used to approximate the posterior of the model parameters. Similarly to Ch. 4, however, computationally less demanding ML or MAP estimates are often sufficient, while more involved approximations are used for the states. In order to estimate the likelihood of the parameters, consider a dataset $\mathcal{D} = \{\mathbf{y}_{1:T_n}^{(n)}\}_{n=1}^N$ consisting of N sequences of possibly different lengths T_n.[2] As usual, the marginal likelihood $p_\theta(\mathcal{D}) = \prod_{n=1}^N p_\theta(\mathbf{y}_{1:T_n}^{(n)})$ is assumed to factorise over the observed sequences. Therefore, is suffices to consider a single sequence with the log marginal likelihood $\log p_\theta(\mathbf{y}_{1:T_n}^{(n)})$ in the following.

Common ML and MAP approximation approaches for SSMs estimate the log marginal likelihood e.g. directly using (approximate) filtering methods, or using the ELBO form using (approximate) smoothing methods. As explained in Ch. 4, the log marginal likelihood reformulated in the ELBO form also gives rise to the EM algorithm. Furthermore, the ELBO form shows the relation to gradient-based estimation techniques using Fisher's identity.

Direct estimation. The key to direct estimation of $\log p_\theta(\mathbf{y}_{1:T})$ is to use the sequential factorisation $p_\theta(\mathbf{y}_{1:T}) = \prod_{t=1}^T p_\theta(\mathbf{y}_t|\mathbf{y}_{1:t-1})$ and compute the conditionals as

$$\begin{aligned} \log p_\theta(\mathbf{y}_{1:T}) &= \sum_{t=1}^T \log p_\theta(\mathbf{y}_t|\mathbf{y}_{1:t-1}) \\ &= \sum_{t=1}^T \log \int p_\theta(\mathbf{y}_t|\mathbf{x}_t) p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t-1}) d\mathbf{x}_t \\ &= \sum_{t=1}^T \log \int p_\theta(\mathbf{y}_t|\mathbf{x}_t) \int p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}) p_\theta(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) d\mathbf{x}_{t-1} d\mathbf{x}_t, \end{aligned} \tag{5.6}$$

where $p_\theta(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) = p_\theta(\mathbf{x}_0)$ for $t = 1$. Each conditional $p_\theta(\mathbf{y}_t|\mathbf{y}_{1:t-1})$ can then be computed recursively as a by-product of an (approximate) filtering algorithm: The inner integral can be identified as the prediction step in the filter, and the outer integral is the predictive distribution, computed in the update step. When using gradient-based methods to optimise this direct estimate (or approximation thereof), gradients must be backpropagated through the whole chain of filtering operations.

[2] Additional controls (inputs) $\mathbf{u}_{1:T_n}^{(n)}$ are often used, but not considered in this section to simplify notation.

Fisher's identity. As an alternative to direct gradient computation, the identity

$$\nabla_\theta \; p_\theta(\mathbf{y}_{1:T}) = \mathbb{E}_{p_\theta(\mathbf{x}_{1:T}|\mathbf{y}_{1:T})}\big[\nabla_\theta \log p_\theta(\mathbf{y}_{1:T}, \mathbf{x}_{0:T})\big] \tag{5.7}$$

is commonly used to compute the gradients more efficiently. This identity is referred to as Fisher's identity (see e.g. [124]). The expectation is w.r.t. the true posterior, and, thus, smoothing is required. However, gradients are computed only w.r.t. the joint distribution—also referred to as complete likelihood—which factorises according to Eq. (5.1a); thus, gradients are no longer back-propagated through the filter chain (or the smoothing distribution).

ELBO estimation. Fisher's identity can also be regarded as the gradient of the ELBO, where the true posterior is used in place of an approximating distribution. This objective also gives rise to the EM algorithm. To see this, the ELBO is written in the entropy form

$$\log p_\theta(\mathbf{y}_{1:T}) \geq \mathbb{E}_{q(\mathbf{x}_{0:T})}\big[\log p_\theta(\mathbf{y}_{1:T}, \mathbf{x}_{0:T})\big] + H\big[q(\mathbf{x}_{0:T})\big] =: \mathcal{L}_{\text{ELBO}}(\mathbf{y}_{1:T}, q, \theta), \tag{5.8}$$

The EM algorithm alternates (coordinate ascend) between optimising the ELBO w.r.t. q and θ, respectively. In the E-step, the true posterior $q^*(\mathbf{x}_{0:T}) = p_\theta(\mathbf{x}_{0:T}|\mathbf{y}_{1:T})$ is computed as the optimal distribution and the bound becomes tight. The M-step is usually carried out in closed-form, setting the derivatives w.r.t. θ to zero and solving for the resulting equations. Fisher's identity is obtained if gradient-based methods are used in the M-step. Since the true posterior is optimal and thus held fixed in the M-step, the entropy term in Eq. (5.8) can be dropped from the optimisation and the gradient operator can be pulled into the expectation [125]. The remaining term in the ELBO objective can be simplified due to the Markov properties from Eqs. (5.2a) and (5.2b):

$$\begin{aligned}
&\mathbb{E}_{q(\mathbf{x}_{0:T})}\big[\log p_\theta(\mathbf{y}_{1:T}, \mathbf{x}_{0:T})\big] \\
&= \sum_{t=1}^{T} \underbrace{\mathbb{E}_{q(\mathbf{x}_t)}\big[\log p_\theta(\mathbf{y}_t|\mathbf{x}_t)\big]}_{\ell_{\text{emit}}} + \sum_{t=1}^{T} \underbrace{\mathbb{E}_{q(\mathbf{x}_t, \mathbf{x}_{t-1})}\big[\log p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})\big]}_{\ell_{\text{trans}}} + \underbrace{\mathbb{E}_{q(\mathbf{x}_0)}\big[\log p_\theta(\mathbf{x}_0)\big]}_{\ell_{\text{prior}}}.
\end{aligned} \tag{5.9}$$

The required one-slice marginals (smoothing distribution) and two-slice marginals are both computed e.g. in forward-backward smoothing.

Although the entropy term can be dropped for optimisation if the true posterior is obtained, it is required if the value of the log marginal likelihood is of interest besides the parameters that maximise it. Fortunately, the structure of the true posterior is Markovian (cf. Eqs. (5.2c) and (5.2d)). Consequently, the posterior of the whole state sequence $\mathbf{x}_{0:T}$ can be computed e.g. using one of the following options: a) posterior forward conditionals; b) posterior backward conditionals; or c) one-slice and two-slice (smoothing) marginals. The corresponding entropy formulations are

given as

$$H\big[q(\mathbf{x}_{0:T})\big] = -\sum_{t=1}^{T} \mathbb{E}_{q(\mathbf{x}_t,\mathbf{x}_{t-1})}\big[\log q(\mathbf{x}_t|\mathbf{x}_{t-1})\big] - \mathbb{E}_{q(\mathbf{x}_0)}\big[\log q(\mathbf{x}_0)\big] \tag{5.10a}$$

$$= -\sum_{t=0}^{T-1} \mathbb{E}_{q(\mathbf{x}_t,\mathbf{x}_{t+1})}\big[\log q(\mathbf{x}_t|\mathbf{x}_{t+1})\big] - \mathbb{E}_{q(\mathbf{x}_T)}\big[\log q(\mathbf{x}_T)\big] \tag{5.10b}$$

$$= -\sum_{t=0}^{T-1} \mathbb{E}_{q(\mathbf{x}_t,\mathbf{x}_{t+1})}\big[\log q(\mathbf{x}_t,\mathbf{x}_{t+1})\big] + \sum_{t=1}^{T-1} \mathbb{E}_{q(\mathbf{x}_t)}\big[\log q(\mathbf{x}_t)\big]. \tag{5.10c}$$

The respective distributions can be obtained e.g. from backward-forward or forward-backward smoothing.

5.3 Gaussian Linear Dynamical Systems

The GLS is the simplest and most studied SSM since inference, prediction, and likelihood estimation is tractable and efficient algorithms exist. These are described in the following, using additional controls/inputs $\mathbf{u}$ in order to provide background for Ch. 9. In the GLS, both the dynamical and emission model are linear transformations with additive Gaussian noise

$$\mathbf{x}_0 \sim \mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0), \tag{5.11a}$$

$$\mathbf{x}_t = \mathbf{A}_t\mathbf{x}_{t-1} + \mathbf{b}_t + \mathbf{w}_t, \qquad \mathbf{w}_t \sim \mathcal{N}(\mathbf{w}_t; \mathbf{0}, \mathbf{R}_t), \tag{5.11b}$$

$$\mathbf{y}_t = \mathbf{C}_t\mathbf{x}_t + \mathbf{d}_t + \mathbf{v}_t, \qquad \mathbf{v}_t \sim \mathcal{N}(\mathbf{v}_t; \mathbf{0}, \mathbf{Q}_t), \tag{5.11c}$$

where the biases $\mathbf{b}_t$ and $\mathbf{d}_t$ can also be a non-linear function of inputs/controls $\mathbf{u}_t$, but are more commonly given by a linear transformation $\mathbf{b}_t = \mathbf{B}_t\mathbf{u}_t$ and $\mathbf{d}_t = \mathbf{D}_t\mathbf{u}_t$ with state and emission control matrices $\mathbf{B}_t$ and $\mathbf{D}_t$[3]. $\mathbf{A}_t$ and $\mathbf{C}_t$ are the transition and emission matrices, and $\mathbf{R}_t$ and $\mathbf{Q}_t$ are the state and emission noise covariance matrices, respectively.

5.3.1 Inference

Inference is tractable in the GLS using the famous Kalman filter and smoother [126]. This section provides details for the (forward) filtering and forward backward smoothing recursions.

Filtering. The Bayes filter update from Eq. (5.3) can be subdivided into two steps, alternating between the *prediction step* and *update step* at every time-slice t. The *prediction step* computes the prior for slice t by taking the transition and marginalising the previous state

$$\begin{aligned} p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t-1}, \mathbf{u}_{1:t}) &= \int p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t) p_\theta(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}, \mathbf{u}_{1:t-1}) d\mathbf{x}_{t-1} \\ &= \mathcal{N}\big(\mathbf{x}_t;\, \mathbf{m}_{t|t-1}, \mathbf{V}_{t|t-1}\big), \end{aligned} \tag{5.12}$$

[3] Linear controls allow for analytical solutions in the M-step of the EM algorithm, however, the exact log likelihood and its gradients can be computed even for non-linear additive biases.

where

$$\begin{aligned}\mathbf{m}_{t|t-1} &= \mathbf{A}_t\mathbf{m}_{t-1} + \mathbf{b}_t,\\ \mathbf{V}_{t|t-1} &= \mathbf{A}_t\mathbf{V}_{t-1}\mathbf{A}_t^T + \mathbf{R}_t.\end{aligned}$$

Similarly, the *predictive distribution* (emission), which is used in the *update step* below, is obtained through marginalisation of the state in the emission

$$\begin{aligned}p_\theta(\mathbf{y}_t|\mathbf{y}_{1:t-1}) &= \int p_\theta(\mathbf{y}_t|\mathbf{x}_t)p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t-1})d\mathbf{x}_t\\ &= \mathcal{N}\big(\mathbf{y}_t;\, \mathbf{m}_{t|t}, \mathbf{V}_{t|t}\big),\end{aligned} \tag{5.13}$$

where

$$\begin{aligned}\mathbf{m}_{t|t} &= \mathbf{C}_t\mathbf{m}_{t|t-1} + \mathbf{d}_t,\\ \mathbf{V}_{t|t} &= \mathbf{C}_t\mathbf{V}_{t|t-1}\mathbf{C}_t^T + \mathbf{Q}_t.\end{aligned}$$

The *update step* is a standard Bayesian update in a Gaussian linear model

$$\begin{aligned}p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t}) &\propto p_\theta(\mathbf{y}_t|\mathbf{x}_t)p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t-1})\\ &= \mathcal{N}\big(\mathbf{x}_t;\, \mathbf{m}_t, \mathbf{V}_t\big),\end{aligned} \tag{5.14}$$

where

$$\begin{aligned}\mathbf{m}_t &= \mathbf{m}_{t|t-1} + \mathbf{G}_t\big(\mathbf{y}_t - \mathbf{m}_{t|t}\big),\\ \mathbf{V}_t &= \mathbf{V}_{t|t-1} - \mathbf{G}_t\mathbf{C}_t\mathbf{V}_{t|t-1},\\ \mathbf{G}_t &= \mathbf{V}_{t|t-1}^T\mathbf{C}_t^T\mathbf{V}_{t|t}^{-1}.\end{aligned}$$

The matrix $\mathbf{G}_t$ is often referred to as the Kalman gain matrix.

Smoothing. As mentioned in Sec. 5.2.1, there are several algorithms for computing the smoothing distribution. This section provides the relevant equations for the forward-backward smoothing variant [127], which is the most common smoother implementation; it is nowadays referred to as RTS or Kalman smoother. After having computed the last filter distribution $p_\theta(\mathbf{x}_T|\mathbf{y}_{1:T})$, forward-backward smoothing iterates backwards recursively, computing

$$\begin{aligned}p_\theta(\mathbf{x}_t|\mathbf{y}_{1:T}) &\propto \int p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y}_{1:t})p_\theta(\mathbf{x}_{t+1}|\mathbf{y}_{1:T})d\mathbf{x}_{t+1}\\ &= \mathcal{N}\big(\mathbf{x}_t;\, \tilde{\mathbf{m}}_t, \tilde{\mathbf{V}}_t\big),\end{aligned} \tag{5.15}$$

where

$$\begin{aligned}\tilde{\mathbf{m}}_t &= \mathbf{m}_t + \mathbf{H}_t(\tilde{\mathbf{m}}_{t+1} - \mathbf{m}_{t+1|t}),\\ \tilde{\mathbf{V}}_t &= \mathbf{V}_t + \mathbf{H}_t(\tilde{\mathbf{V}}_{t+1} - \mathbf{V}_{t+1|t})\mathbf{G}_t^T,\\ \mathbf{H}_t &= \mathbf{V}_t\mathbf{A}_t^T\mathbf{V}_{t|t-1}^{-1}.\end{aligned}$$

5.3.2 Parameter Estimation

Direct estimation. Estimating $\log p_\theta(\mathbf{y}_{1:T})$ directly using Eq. (5.6) is straightforward, since each incremental likelihood $p_\theta(\mathbf{y}_t|\mathbf{y}_{1:t-1})$ is computed by the Kalman filter already in Eq. (5.13). Taking the logarithm, results in

$$\begin{aligned}\log p_\theta(\mathbf{y}_{1:T}) &= \sum_{t=1}^{T} \log \mathcal{N}\big(\mathbf{y}_t;\, \mathbf{m}_{t|t}, \mathbf{V}_{t|t}\big) \\ &= \sum_{t=1}^{T} -\frac{D_y}{2}\log 2\pi + \frac{1}{2}\log|\mathbf{V}_{t|t}^{-1}| - \frac{1}{2}(\mathbf{y}_t - \mathbf{m}_{t|t})^T\mathbf{V}_{t|t}^{-1}(\mathbf{y}_t - \mathbf{m}_{t|t}),\end{aligned} \tag{5.16}$$

where D_y denotes the dimension of the observation space.

ELBO estimation. Estimating the log likelihood in the ELBO form is slightly more involved, however, gradient computation is much more efficient and the EM algorithm with exact solutions in the M-step can be used. The terms of the complete log likelihood (Eq. (5.9)) and the posterior entropy $H[q(\mathbf{x}_{1:T})]$ require the one-slice and two-slice posterior marginals, computed e.g. in forward backward or backward forward smoothers. Denote these joint Gaussian two-slice posteriors as

$$q(\mathbf{x}_{t-1}, \mathbf{x}_t) = \mathcal{N}\left(\begin{bmatrix}\mathbf{x}_{t-1}\\ \mathbf{x}_t\end{bmatrix}; \begin{bmatrix}\tilde{\mathbf{m}}_{t-1}\\ \tilde{\mathbf{m}}_t\end{bmatrix}, \begin{bmatrix}\tilde{\mathbf{V}}_{t-1} & \tilde{\mathbf{V}}_{t-1,t}\\ \tilde{\mathbf{V}}_{t,t-1} & \tilde{\mathbf{V}}_t\end{bmatrix}\right), \tag{5.17}$$

where $\tilde{\mathbf{V}}_{t-1,t} = \tilde{\mathbf{V}}_{t,t-1}^T$ is the covariance between adjacent states and $\tilde{\mathbf{V}}_{t-1}$, $\tilde{\mathbf{V}}_t$ are the covariances of the one-slice marginals (smoothing distribution). Further denote the covariance of the posterior conditionals $q(\mathbf{x}_t|\mathbf{x}_{t-1})$ and $q(\mathbf{x}_t|\mathbf{x}_{t+1})$—given by Gaussian conditioning—as

$$\tilde{\mathbf{V}}_{t|t-1} = \tilde{\mathbf{V}}_t - \tilde{\mathbf{V}}_{t-1,t}^T\tilde{\mathbf{V}}_{t-1}^{-1}\tilde{\mathbf{V}}_{t-1,t}, \tag{5.18a}$$

$$\tilde{\mathbf{V}}_{t|t+1} = \tilde{\mathbf{V}}_t - \tilde{\mathbf{V}}_{t+1,t}^T\tilde{\mathbf{V}}_{t+1}^{-1}\tilde{\mathbf{V}}_{t+1,t}. \tag{5.18b}$$

The prior term in Eq. (5.9) is given as

$$\begin{aligned}\ell_{\text{prior}} &= \mathbb{E}_{q(\mathbf{x}_0)}\big[\log \mathcal{N}(\mathbf{x}_0;\, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)\big] \\ &= -\frac{D_x}{2}\log 2\pi + \frac{1}{2}\log|\boldsymbol{\Sigma}_0^{-1}| - \frac{1}{2}\mathcal{E}(\mathbf{x}_0),\end{aligned} \tag{5.19a}$$

$$\begin{aligned}\mathcal{E}_{\text{prior}}(\mathbf{x}_0) &= \mathbb{E}_{q(\mathbf{x}_0)}\big[(\mathbf{x}_0 - \boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}_0^{-1}(\mathbf{x}_0 - \boldsymbol{\mu}_0)\big] \\ &= \operatorname{Tr}\left\{\boldsymbol{\Sigma}_0^{-1}\mathbb{E}_{q(\mathbf{x}_0)}\big[(\mathbf{x}_0 - \boldsymbol{\mu}_0)(\mathbf{x}_0 - \boldsymbol{\mu}_0)^T\big]\right\} \\ &= \operatorname{Tr}\Big\{\boldsymbol{\Sigma}_0^{-1}\big[(\tilde{\mathbf{m}}_0 - \boldsymbol{\mu}_0)(\tilde{\mathbf{m}}_0 - \boldsymbol{\mu}_0)^T + \tilde{\mathbf{V}}_0\big]\Big\}.\end{aligned} \tag{5.19b}$$

Expectations of the log transitions are w.r.t. two-slice posteriors $q(\mathbf{x}_{t-1,t}) = \mathcal{N}(\mathbf{x}_{t-1,t};\, \tilde{m}_{t-1,t}, \tilde{V}_{t-1,t})$, which are computed e.g. in forward-backward smoothing using posterior backward transitions.

The transition terms can then be computed as

$$\begin{aligned}\ell_{\text{trans}} &= \mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)}\big[\log \mathcal{N}(\mathbf{x}_t;\, \mathbf{A}_t\mathbf{x}_{t-1}+\mathbf{b}_t, \mathbf{R}_t)\big] \\ &= -\frac{D_x}{2}\log 2\pi + \frac{1}{2}\log|\mathbf{R}_t^{-1}| - \frac{1}{2}\mathcal{E}_{\text{trans}}(\mathbf{x}_{t-1,t}),\end{aligned} \tag{5.20a}$$

$$\begin{aligned}\mathcal{E}_{\text{trans}}(\mathbf{x}_{t-1,t}) &= \mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)}\big[(\mathbf{x}_t-\mathbf{A}_t\mathbf{x}_{t-1}-\mathbf{b}_t)^T\mathbf{R}_t^{-1}(\mathbf{x}_t-\mathbf{A}_t\mathbf{x}_t-\mathbf{b}_t)\big] \\ &= \operatorname{Tr}\Big\{\mathbf{R}_t^{-1}\mathbb{E}_{q(\mathbf{x}_{t-1},\mathbf{x}_t)}\big[(\mathbf{x}_t-\mathbf{A}_t\mathbf{x}_{t-1}-\mathbf{b}_t)(\mathbf{x}_t-\mathbf{A}_t\mathbf{x}_{t-1}-\mathbf{b}_t)^T\big]\Big\} \\ &= \operatorname{Tr}\Big\{\mathbf{R}_t^{-1}\big[(\tilde{\mathbf{m}}_t-\mathbf{A}_t\tilde{\mathbf{m}}_{t-1}-\mathbf{b}_t)(\tilde{\mathbf{m}}_t-\mathbf{A}_t\tilde{\mathbf{m}}_{t-1}-\mathbf{b}_t)^T \\ &\qquad + \mathbf{A}_t\tilde{\mathbf{V}}_{t-1}\mathbf{A}_t^T + \tilde{\mathbf{V}}_t - \mathbf{A}_t\tilde{\mathbf{V}}_{t-1,t} - \tilde{\mathbf{V}}_{t,t-1}\mathbf{A}_t^T\big]\Big\},\end{aligned} \tag{5.20b}$$

where $\mathbf{V}_{t,t-1}$ denotes the posterior covariance between adjacent states $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$. Finally, the emission terms result in

$$\begin{aligned}\ell_{\text{emit}} &= \mathbb{E}_{q(\mathbf{x}_t)}\big[\log \mathcal{N}(\mathbf{y}_t;\, \mathbf{C}_t\mathbf{x}_t+\mathbf{d}_t, \mathbf{Q}_t)\big] \\ &= -\frac{D_y}{2}\log 2\pi + \frac{1}{2}\log|\mathbf{Q}_t^{-1}| - \frac{1}{2}\mathcal{E}_{\text{emit}}(\mathbf{x}_t)\end{aligned} \tag{5.21a}$$

$$\begin{aligned}\mathcal{E}_{\text{emit}}(\mathbf{x}_t) &= \mathbb{E}_{q(\mathbf{x}_t)}\big[(\mathbf{y}_t-\mathbf{C}_t\mathbf{x}_t-\mathbf{d}_t)^T\mathbf{Q}_t^{-1}(\mathbf{y}_t-\mathbf{C}_t\mathbf{x}_t-\mathbf{d}_t)\big] \\ &= \operatorname{Tr}\Big\{\mathbf{Q}_t^{-1}\mathbb{E}_{q(\mathbf{x}_t)}\big[(\mathbf{y}_t-\mathbf{C}_t\mathbf{x}_t-\mathbf{d}_t)(\mathbf{y}_t-\mathbf{C}_t\mathbf{x}_t-\mathbf{d}_t)^T\big]\Big\} \\ &= \operatorname{Tr}\Big\{\mathbf{Q}_t^{-1}\big[(\mathbf{y}_t-\mathbf{C}_t\tilde{\mathbf{m}}_t-\mathbf{d}_t)(\mathbf{y}_t-\mathbf{C}_t\tilde{\mathbf{m}}_t-\mathbf{d}_t)^T + \mathbf{C}_t\tilde{\mathbf{V}}_t\mathbf{C}_t^T\big]\Big\}.\end{aligned} \tag{5.21b}$$

ML parameter estimates can be found by optimising the complete log likelihood w.r.t. the parameters of the GLS.

Although the entropy term is not required for ML or MAP parameter estimation, it can be computed using either form of Eq. (5.10), e.g.

$$H\big[q(\mathbf{x}_{0:T})\big] = \sum_{t=0}^{T}\frac{D_x}{2}(1+\log 2\pi) + \frac{1}{2}\log|\mathbf{V}_{t|t-1}|, \tag{5.22a}$$

$$= \sum_{t=0}^{T}\frac{D_x}{2}(1+\log 2\pi) + \frac{1}{2}\log|\mathbf{V}_{t|t+1}|. \tag{5.22b}$$

where $\mathbf{V}_{t|t+1} = \mathbf{V}_T$ for $t = T$, and $\mathbf{V}_{t|t-1} = \mathbf{V}_t$ for $t = 0$, respectively. The resulting log likelihood estimate in ELBO form (Eq. (5.8)) with the true posterior equals the direct estimate (Eq. (5.16)) obtained through filtering, however, the computational graphs for the loss and gradient computations differ substantially.

5.4 Variational Sequential Monte Carlo

For SSMs with non-linear/non-Gaussian dynamical and emission models, inference and parameter estimation is not tractable since integrals that occur in the marginalisations and normalisation constants cannot be computed analytically. Similarly to other non-linear LVMs (cf. Ch. 4),

approximations are required. Various methods have been proposed in the literature for non-linear/non-Gaussian SSMs, including i) Taylor approximations such as the extended Kalman filter (see e.g. [128, 129]) and second order versions [130]; ii) assumed density (Gaussian) approximations [129, 131, 132, 133, 134]; iii) variational approximations [135, 136]; and iv) SMC methods [75, 76, 77].

This section considers SMC for SSMs and then describes recent variational SMC approaches, providing the basis for inference and parameter estimation in Ch. 9. For SMC methods it is more convenient to consider the SSM with joint probability $p_\theta(\mathbf{y}_{1:T}, \mathbf{x}_{1:T})$, i.e. without initial state $\mathbf{x}_0$, to avoid sampling an initial state without corresponding observations. In the following, the initial state will therefore not be considered.

SMC for DSSMs. Building on the generic SMC method described in Sec. 2.5, the sequence of (unnormalised and normalised) target distributions in non-linear/non-Gaussian SSMs (cf. Eq. (5.1)) are the joint distribution $\tilde{\nu}(\mathbf{x}_{1:t}) = p_\theta(\mathbf{x}_{1:t}, \mathbf{y}_{1:t})$, and the posterior $\nu(\mathbf{x}_{1:t}) = p_\theta(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$, respectively; the normalisation constant is the marginal likelihood $Z_t = p_\theta(\mathbf{y}_{1:t})$ and the incremental normalisation constants are $Z_{t|t-1} = p_\theta(\mathbf{y}_t|\mathbf{y}_{1:t-1})$. The *incremental importance weights* are given as

$$\gamma(\mathbf{x}_t^{(p)}, \mathbf{x}_{1:t-1}^{(p)}) = \frac{p_\theta(\mathbf{y}_t|\mathbf{x}_t^{(p)}) p_\theta(\mathbf{x}_t^{(p)}|\mathbf{x}_{t-1}^{(p)})}{\pi(\mathbf{x}_t^{(p)}|\mathbf{x}_{1:t-1}^{(p)})}. \tag{5.23}$$

The particle approximation of the target distribution (Eq. (2.29)) then gives an approximation of the filter distribution:

$$p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t-1}) \approx \sum_{p=1}^{P} w_t^{(p)} \delta(\mathbf{x}_t^{(p)}). \tag{5.24}$$

Therefore, SMC methods applied to the filtering problem in SSMs are also referred to as *particle filters*. SMC methods can also be applied to smoothing or to infer the full posterior distribution. Existing *particle smoothers* are based on the different posterior decompositions described in Sec. 5.2 (see e.g. [76, 137, 119, 121]). Particle smoothers are commonly used for parameter estimation with gradient-based methods using Fisher's identity (cf. Sec. 5.2.2 and see e.g. [124, 138]), since SMC yields sample trajectories from the posterior[4].

Variational SMC for DSSMs. The quality of the particle approximation from SMC heavily depends on the proposal distribution. Variational SMC methods parametrise the proposal distribution $\pi_\phi(\mathbf{x}_t|\mathbf{x}_{1:t-1})$ in Eq. (5.23) with variational parameters ϕ and *learn* them jointly with the model parameters θ [40, 139, 140]. Building on neural/amortised variational inference, the parameters of the proposal distribution are predicted from previous particles $\mathbf{x}_{1:t-1}^{(p)}$ by a NN. Optionally, additional data such as the current observations $\mathbf{y}_t$, past observations $\mathbf{y}_{1:t}$, or even all observations $\mathbf{y}_{1:T}$ can be used.

[4] Particle filters also yield samples from the posterior if the particle histories (paths) are not discarded and the last importance weights are used [76]. However, this (naive) approach suffers from severe path degeneracy.

Unfortunately, Fisher's identity is not applicable since the variational parameters are learned and gradients w.r.t. this parametrised proposal distribution are not zero. However, gradient can be estimated based on the direct form or the ELBO form of the likelihood (see Sec. 5.2). In case of the direct estimation, an unbiased estimate of the marginal likelihood can be obtained through $\hat{Z}_t = \prod_{t=1}^{T} \hat{Z}_{t|t-1}$, where the estimates $\hat{Z}_{t|t-1}$ are given by Eqs. (2.30) and (2.28). Since SMC provides an unbiased estimate of the marginal likelihood, i.e. $\mathbb{E}\big[\prod_{t=1}^{T} \hat{Z}_{t|t-1}\big] = p_\theta(\mathbf{y}_{1:T})$, the log estimate $\mathbb{E}\big[\sum_{t=1}^{T} \log \hat{Z}_{t|t-1}\big] \leq \log p_\theta(\mathbf{y}_{1:T})$ is a lower bound to the log marginal likelihood due to Jensen's inequality.

This variational objective—which is used in Ch. 9—is based on particle *filtering*. However, a smoothing variational objective has also been proposed more recently [41]. This approach approximates the log marginal likelihood of the entire sequence $\mathbf{y}_{1:T}$ using particle smoothing (forward filtering backward simulation) instead of approximating the conditionals $\log p_\theta(\mathbf{y}_t|\mathbf{y}_{1:t-1})$.

Part II

Own Publications

6 Continual Learning with Bayesian Neural Networks for Non-stationary Data

Continual learning with non-stationary data considers scenarios in which data sets arrive sequentially, inferences and predictions should be performed immediately, and the data distribution may change over time. For computational reasons, the posterior approximations given data up to a certain time step should re-use previous inference computations, while avoiding information loss due to poor approximations. At the same time, changes in the data distribution require methods for adaptation. In the Bayesian framework, both sequential learning and adaptation can be approached in a natural and simple manner. To provide a more powerful posterior approximation, a memory-based variational approximation is developed that combines a Gaussian variational distribution with a running memory of raw data points. The memory is selected such that data—for which the corresponding likelihood terms cannot be well approximated by a Gaussian—are kept in raw form and the remaining data is projected into the Gaussian variational distribution. For the adaptation, two alternative approaches are proposed: Bayesian exponential forgetting, which assigns lower weighting factors to past likelihood terms; or a diffusion process assumption on the time evolution of the NN weights.

Note that in the corresponding background sections (Sec. 2.3 and Ch. 3) the NN weights and variational parameters are defined as θ and ϕ, respectively, whereas in the publication presented in this chapter, $\mathbf{w}$ is used for the weights and θ for the variational parameters.

Authors	Richard Kurle Botond Cseke Alexej Klushyn Patrick van der Smagt Stephan Günnemann	

Conference	International Conference on Learning Representation, ICLR 2020	

Contribution	Problem definition	contributed
	Literature survey	significantly contributed
	Algorithm development	contributed
	Method implementation	significantly contributed
	Experimental evaluation	significantly contributed
	Preparation of the manuscript	contributed

CONTINUAL LEARNING WITH BAYESIAN NEURAL NETWORKS FOR NON-STATIONARY DATA

Richard Kurle[*1 2] **Botond Cseke**[1] **Alexej Klushyn**[1 2]
Patrick van der Smagt[1] **Stephan Günnemann**[2]
[1]Volkswagen Group [2]Technical University of Munich

ABSTRACT

This work addresses continual learning for non-stationary data, using Bayesian neural networks and memory-based online variational Bayes. We represent the posterior approximation of the network weights by a diagonal Gaussian distribution and a complementary memory of raw data. This raw data corresponds to likelihood terms that cannot be well approximated by the Gaussian. We introduce a novel method for sequentially updating both components of the posterior approximation. Furthermore, we propose Bayesian forgetting and a Gaussian diffusion process for adapting to non-stationary data. The experimental results show that our update method improves on existing approaches for streaming data. Additionally, the adaptation methods lead to better predictive performance for non-stationary data.

1 INTRODUCTION

Continual learning (CL), also referred to as lifelong learning, is typically described informally by the following set of desiderata for computational systems: the system should (i) learn *incrementally* from a data stream, (ii) exhibit *information transfer* forward and backward in time, (iii) avoid *catastrophic forgetting* of previous data, and (iv) *adapt* to changes in the data distribution (Ring, 1997; Silver et al., 2013; Chen & Liu, 2016; Ruvolo & Eaton, 2013; Parisi et al., 2018). The necessity to adapt to non-stationary data is often not reconcilable with the goal of preventing forgetting. This problem is also known as the stability-plasticity dilemma (Grossberg, 1987).

The majority of current CL research is conducted in the context of online multi-task learning (Nguyen et al., 2018; Kirkpatrick et al., 2017; Schwarz et al., 2018; Rusu et al., 2016; Fernando et al., 2017), where the main objective is to prevent catastrophic forgetting of previously learned tasks. This focus is reasonable since changes in the statistics of the data distribution are usually an artefact of learning different tasks sequentially. However, changes in the statistics of the data can also be real properties of the data-generating process. Examples include models of energy demand, climate analysis, financial market, or user-behavior analytics (Ditzler et al., 2015). In such applications, the statistics of the current data distribution are of particular interest. Old data may be outdated and can even deteriorate learning if the *drift* in the data distribution is neglected. Consequently, CL systems for non-stationary data require adaptation methods, which deliberately *forget* outdated information.

In this work, we develop an approximate Bayesian approach for training Bayesian neural networks (BNN) (Hinton & van Camp, 1993; Graves, 2011; Blundell et al., 2015) *incrementally* with non-stationary streaming data. Similar to variational continual learning (VCL) (Nguyen et al., 2018) and the Virtual Vector Machine (VVM) (Minka et al., 2009), we approximate the posterior using a Gaussian distribution and a complementary memory of previous data. Both components are *updated* sequentially, while *adapting* to changes in the data distribution. Our main contributions are as follows:

- We propose an online approximation consisting of a diagonal Gaussian distribution and a running memory, and we provide a novel sequential update method for both components.
- We extend the online approximation by two alternative adaptation methods, thereby generalising online variational Bayes with Bayesian neural networks to non-stationary data.

We compare our sequential update method to VCL in the online-inference setting on several popular datasets, demonstrating that our approach is favorable. Furthermore, we validate our adaptation methods on several datasets with *concept drift* (Widmer & Kubat, 1996), showing performance improvements compared to online variational Bayes without adaptation.

[*]Correspondence to richard.kurle@tum.de

2 BACKGROUND: ONLINE INFERENCE

Consider a stream of datasets $\{\mathcal{D}_{t_k}\}_{k=1}^K$, where t_k are the time points at which datasets $\mathcal{D}_{t_k}$ are observed. For the moment, we assume that these datasets and the samples within are generated independently and identically distributed (i.i.d.). Methods for non-i.i.d. data are considered in Sec. 4.

In the Bayesian approach to online learning, we want to infer the posterior distribution $p(\mathbf{w}|\mathcal{D}_{t_1:t_k})$ of our model parameters, with the restriction that the data is processed sequentially.[1] Using Bayes rule, a recursive posterior inference equation emerges naturally:

$$p(\mathbf{w}|\mathcal{D}_{t_1:t_k}) \propto p(\mathbf{w}|\mathcal{D}_{t_1:t_{k-1}})\, p(\mathcal{D}_{t_k}|\mathbf{w}, \mathcal{D}_{t_1:t_{k-1}}) = p(\mathbf{w}|\mathcal{D}_{t_1:t_{k-1}})\, p(\mathcal{D}_{t_k}|\mathbf{w}), \tag{1}$$

where the last step follows from the i.i.d. assumption of the data.

In this paper, we consider Gaussian and multinomial likelihoods, parametrised by a neural network with weights $\mathbf{w}$ and prior $p(\mathbf{w}|\emptyset) = p_0(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mu_0, \sigma_0)$. Furthermore, we consider supervised learning, where $\mathcal{D}_{t_k} = \{\mathbf{d}_{t_k}^{(n)}\}_n = \{(\mathbf{x}_{t_k}^{(n)}, \mathbf{y}_{t_k}^{(n)})\}_n$ and $p(\mathbf{d}_{t_k}^{(n)}|\mathbf{w}) = p(\mathbf{y}_{t_k}^{(n)} \mid \mathrm{NN}(\mathbf{x}_{t_k}^{(n)}; \mathbf{w}))$.

2.1 ONLINE VARIATIONAL BAYES

Since exact Bayesian inference is intractable for non-trivial models, various approximations have been developed. Prominent examples include sequential Monte Carlo (Liu & Chen, 1998), assumed density filtering (Maybeck, 1982), and online variational Bayes (Opper, 1998; Ghahramani, 2000; Sato, 2001; Broderick et al., 2013). Online variational Bayes (VB) approximates the posterior of Eq. (1) by a parametrised distribution $q_{\theta_{t_k}}(\mathbf{w}) \approx p(\mathbf{w}|\mathcal{D}_{t_1:t_k})$ through a sequence of projections:

$$q_{\theta_{t_k}}(\mathbf{w}) = \operatorname*{argmin}_{q_\theta} \mathrm{KL}\big[q_\theta(\mathbf{w}) \,\|\, Z_{t_k}^{-1}\, q_{\theta_{t_{k-1}}}(\mathbf{w})\, p(\mathcal{D}_{t_k}|\mathbf{w})\big], \tag{2}$$

where Z_{t_k} is the normalisation constant. The above minimisation is equivalent to maximising the evidence lower bound (ELBO) $\mathcal{L}_{t_k}(\theta; \mathcal{D}_{t_k}) = \mathbb{E}_{q_\theta(\mathbf{w})}\big[\log p(\mathcal{D}_{t_k}|\mathbf{w})\big] - \mathrm{KL}\big[q_\theta(\mathbf{w}) \,\|\, q_{\theta_{t_{k-1}}}(\mathbf{w})\big]$. In this work, we consider diagonal Gaussian posterior approximations $q_{\theta_{t_k}}(\mathbf{w})$ for the neural network weights, similar to Nguyen et al. (2018).

2.2 ONLINE VARIATIONAL BAYES WITH MEMORY

Online approximate Bayesian inference methods inevitably suffer from an information loss due to the posterior approximation at each time-step. An alternative approach to online learning is to store and update a representative dataset/generative model—and to use it as a memory—in order to improve inference (Robins, 1995; Lopez-Paz & Ranzato, 2017; Shin et al., 2017; Kamra et al., 2017). Memory-based online learning has also been combined with online Bayesian inference methods (Minka et al., 2009; Nguyen et al., 2018). A common property of these approaches is to represent the (current) posterior approximation by a product of two factors

$$p(\mathbf{w}|\mathcal{D}_{t_1:t_k}) \approx q_{\theta_{t_k}}(\mathbf{w})\, p(\mathcal{M}_{t_k}|\mathbf{w}) \tag{3}$$

and update them sequentially as new data $\mathcal{D}_{t_k}$ is observed. The factor $p(\mathcal{M}_{t_k}|\mathbf{w}) = \prod_m^M p(\mathbf{m}_{t_k}^{(m)}|\mathbf{w})$ is the likelihood of a set of $M = |\mathcal{M}|$ data points, which we refer to as *running memory*; and $q_{\theta_{t_k}}(\mathbf{w})$ is a Gaussian distribution, which summarises the rest of the data $\bar{\mathcal{D}}_{1:t_k} = \mathcal{D}_{1:t_k} \setminus \mathcal{M}_{t_k}$.

In case of VCL, the factors in Eq (3) are updated in two steps, which we refer to as (i) *memory update* and (ii) *Gaussian update*: (i) a new memory $\mathcal{M}_{t_k} \subset \mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}}$ is selected using heuristics such as *random* selection or the *k-center* method (a greedy algorithm that selects K data points based on geometric properties of $\mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}}$.); (ii) the Gaussian distribution is updated with the remaining data $\bar{\mathcal{D}}_{t_k} = \mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}} \setminus \mathcal{M}_{t_k}$ (using Eq. (2)) to obtain $q_{\theta_{t_k}}(\mathbf{w}) \approx q_{\theta_{t_{k-1}}}(\mathbf{w})\, p(\bar{\mathcal{D}}_{t_k}|\mathbf{w})$.

Note that we cannot sample directly from the posterior approximation in Eq. (3) and thus we cannot easily evaluate quantities such as the posterior predictive distribution. VCL therefore performs a second projection

$$\tilde{q}_{\theta_{t_k}}(\mathbf{w}) = \operatorname*{argmin}_{q_\theta} \mathrm{KL}\Big[q_\theta(\mathbf{w}) \,\|\, \tilde{Z}_{t_k}^{-1}\, q_{\theta_{t_k}}(\mathbf{w})\, p(\mathcal{M}_{t_k}|\mathbf{w})\Big]. \tag{4}$$

This distribution should not be confused with the recursively updated variational distribution (Eq. (2)).

[1] A strict definition of online learning requires single data samples at each time step instead of batches $\mathcal{D}_{t_k}$.

3 IMPROVING MEMORY-BASED ONLINE VARIATIONAL BAYES

In this section, we focus on two problems of existing approaches using online VB with a running memory: (i) the *memory update* does not take into account the approximation error or approximation capabilities of the variational distribution; (ii) the *Gaussian update*—performed by optimising the ELBO (Eq. (2)) only with data $\bar{\mathcal{D}}_{t_k}$—can fail for streaming data. This is because VB yields poor posterior approximations if the dataset is too small or the neural network architecture has too much capacity (cf. Ghosh et al. (2018), Fig. 1). In Secs. 3.2 and 3.3, we propose improvements to these two update methods. The mathematical background for our approach is provided in Sec. 3.1.

3.1 PROPERTIES OF THE GAUSSIAN VARIATIONAL APPROXIMATION

There are two important properties of the Gaussian variational approximation that we will exploit later: (i) Gaussian approximate posterior distributions factorise into a product of Gaussian terms corresponding to the prior and each likelihood term; (ii) the ELBO can be written as the sum of the approximation's normalisation constant and a sum of residuals corresponding to these factors.

Let $p_0(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mu_0, \Sigma_0)$ be a Gaussian prior and $p(\mathcal{D}|\mathbf{w}) = \prod_n p(\mathbf{d}^{(n)}|\mathbf{w})$ be the likelihood of the observed data $\mathcal{D}$. Furthermore, let $q_\theta(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mu, \Sigma)$ denote the corresponding Gaussian variational approximation with $\theta = \{\mu, \Sigma\}$. Assume that μ and Σ are the optimal parameters corresponding to a (local) maximum of the ELBO $\mathcal{L}(\mu, \Sigma; \mathcal{D})$. The optimality conditions $\partial_\mu \mathcal{L}(\mu, \Sigma; \mathcal{D}) = 0$ and $\partial_\Sigma \mathcal{L}(\mu, \Sigma; \mathcal{D}) = 0$ can be rewritten as follows (Knowles & Minka, 2011; Opper & Archambeau, 2008; Cseke et al., 2013) (cf. App. C):

$$\Sigma^{-1}\mu = \Sigma_0^{-1}\mu_0 + \sum_n \Big(\partial_\mu \mathbb{E}_{q_\theta(\mathbf{w})}\big[\log p(\mathbf{d}^{(n)}|\mathbf{w})\big] - 2\partial_\Sigma \mathrm{E}_{q_\theta(\mathbf{w})}\big[\log p(\mathbf{d}^{(n)}|\mathbf{w})\big]\mu\Big), \tag{5a}$$

$$\Sigma^{-1} = \Sigma_0^{-1} - 2\sum_n \partial_\Sigma \mathbb{E}_{q_\theta(\mathbf{w})}\big[\log p(\mathbf{d}^{(n)}|\mathbf{w})\big]. \tag{5b}$$

Since the sum of natural parameters corresponds to a product in distribution space, the above equations show that—at a local optimum—the approximation $q_\theta(\mathbf{w})$ factorises in the same way as the posterior $p(\mathbf{w}|\mathcal{D})$. It can be written in the form $q_\theta(\mathbf{w}) = {Z_q}^{-1} p_0(\mathbf{w}) \prod_n \mathbf{r}^{(n)}(\mathbf{w})$, where the factors $\mathbf{r}^{(n)}(\mathbf{w})$ are Gaussian functions with natural parameters given by Eqs. (5a) and (5b), and where $Z_q = \int p_0(\mathbf{w}) \prod_n \mathbf{r}^{(n)}(\mathbf{w})\, d\mathbf{w}$ is the normalisation constant. These Gaussian functions $\mathbf{r}^{(n)}(\mathbf{w})$ each correspond to the contribution of the likelihood $p(\mathbf{d}^{(n)}|\mathbf{w})$ to the posterior approximation $q_\theta(\mathbf{w})$.

The resulting factorisation implies that the ELBO $\mathcal{L}(\mu, \Sigma; \mathcal{D})$ can be written in the form (Opper & Winther, 2005) (c.f. App. D)

$$\mathcal{L}(\mu, \Sigma; \mathcal{D}) = \log Z_q + \sum_n \mathbb{E}_{q_\theta(\mathbf{w})}\big[\log p(\mathbf{d}^{(n)}|\mathbf{w}) - \log \mathbf{r}^{(n)}(\mathbf{w})\big]. \tag{6}$$

If the terms $p(\mathbf{d}^{(n)}|\mathbf{w})$ were (diagonal) Gaussian in $\mathbf{w}$, they would each cancel with the corresponding (diagonal) Gaussian term, leaving only $\log Z_q$. Intuitively, the residual terms in Eq. (6) can be used to quantify the quality of the Gaussian approximation.

3.2 MEMORY UPDATE

The authors of VCL propose to use a memory to compensate the information loss resulting from the Gaussian approximation of the posterior distribution. However, their *memory update* is independent of the approximation error that is due to the chosen distributional family (diagonal Gaussian). An alternative *memory update*, which specifically targets the above mentioned information loss, has been introduced previously for VVM. Although the latter method was developed for expectation propagation in a (linear) logistic regression model—and is thus not directly applicable to online VB—we show that some of its properties can be transferred to the variational inference setting. The central idea is to replace the likelihood terms that can be well approximated by a Gaussian distribution by their Gaussian proxies $p(\mathbf{d}_{t_k}|\mathbf{w}) \approx \mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k})$ resulting in $q_{\theta_{t_k}}(\mathbf{w})$; and retain the data corresponding to the rest of the likelihood terms in the memory. To score a *candidate* memory, Minka et al. (2009) proposed to maximise the KL divergence between the model given in the form of Eq. (3)

and a Gaussian posterior approximation, that is, maximise $\mathrm{KL}\big[\tilde{Z}_{t_k}^{-1}\, q_{\theta_{t_k}}(\mathbf{w})\, p(\mathcal{M}|\mathbf{w}) \,\|\, \tilde{q}_{\theta_{t_k}}(\mathbf{w})\big]$. However, this score function is intractable, because the expectation in the KL includes the likelihood $p(\mathcal{M}|\mathbf{w})$. In the following, we develop a tractable score function applicable to VB. Intuitively, we can use Eq. (6) to test how much $\mathcal{L}(\mu, \Sigma; \mathcal{D})$ changes if we replace the exact likelihood terms (of all data which is not contained in the *candidate* memory) by their Gaussian approximations.

To achieve this, we need to find Gaussian approximations for every data point in the *candidate* memory. We first approximate the posterior distribution using both $\mathcal{D}_{t_k}$ and $\mathcal{M}_{t_{k-1}}$:

$$\tilde{q}_{\theta_{t_k}}(\mathbf{w}) = \operatorname*{argmin}_{q_\theta} \mathrm{KL}\Big[q_\theta(\mathbf{w}) \,\|\, \tilde{Z}_{t_k}^{-1} q_{\theta_{t_{k-1}}}(\mathbf{w})\, p(\mathcal{D}_{t_k}|\mathbf{w})\, p(\mathcal{M}_{t_{k-1}}|\mathbf{w})\Big]. \tag{7}$$

Next, we use Eqs. (5a) and (5b) to calculate the natural parameters of all Gaussian terms. In practice, we estimate the natural parameters using (unbiased) Monte-Carlo estimators for the expectations. We have now available the likelihood terms and their Gaussian approximations. This allows us to write $\mathcal{L}(\theta_{t_k}; \mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}})$ in the form of Eq. (6):

$$\mathcal{L}(\theta_{t_k};\, \mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}}) = \log Z_{q_{t_k}} + \sum_{\mathbf{d}_{t_k} \in \mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}}} \mathbb{E}_{\tilde{q}_{\theta_{t_k}}(\mathbf{w})}\big[\log p(\mathbf{d}_{t_k}|\mathbf{w}) - \log \mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k})\big],$$

where $\mathbf{d}_{t_k}$ are the samples in $\mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}}$ and where $\mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k})$ are the Gaussian approximation of the corresponding likelihood terms. Note that $\mathbf{r}_{t_k}$ does not only depend on $\mathbf{d}_{t_k}$, however, we omit the dependence on the remaining data for notational convenience.

If the likelihood $p(\mathbf{d}_{t_k}|\mathbf{w})$ is close to the Gaussian $\mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k})$ in expectation w.r.t. the approximate posterior $q_{\theta_{t_k}}(\mathbf{w})$, then its contribution to $\mathcal{L}(\theta_{t_k}; \mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}})$ is small. Similarly, likelihood terms that cannot be well approximated by the respective Gaussian have a large contribution, and, hence, the corresponding data should be kept in the memory. For this reason, we propose the score function

$$S_{t_k}(\mathcal{M};\, \mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}}) = \sum_{\mathbf{d}_{t_k} \in \mathcal{M}} \mathbb{E}_{\tilde{q}_{\theta_{t_k}}(\mathbf{w})}\big[\log p(\mathbf{d}_{t_k}|\mathbf{w}) - \log \mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k})\big], \tag{8}$$

and the corresponding *memory update* $\mathcal{M}_{t_k} = \operatorname{argmax}_{\mathcal{M}}\ S_{t_k}(\mathcal{M};\, \mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}})$. Note that since all residual terms are computed independently, the update results in selecting the top M terms.

3.3 Gaussian Update

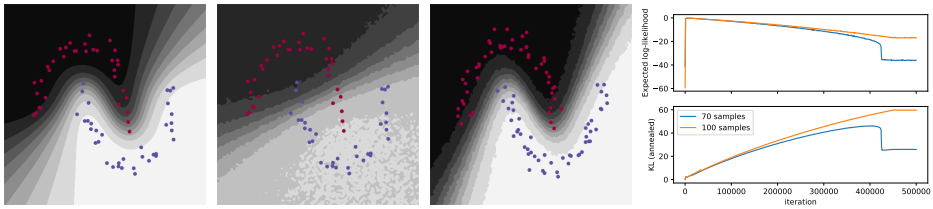
The *Gaussian update* follows from the *memory update* presented in the previous section: once the memory $\mathcal{M}_{t_k}$ has been selected, we update the Gaussian distribution with the approximations corresponding to the rest of the data $\mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}} \setminus \mathcal{M}_{t_k}$. We can update $q_{\theta_{t_k}}(\mathbf{w})$ in two equivalent ways:

$$q_{\theta_{t_k}}(\mathbf{w}) = q_{\theta_{t_{k-1}}}(\mathbf{w}) \prod_{\mathbf{d}_{t_k} \notin \mathcal{M}_{t_k}} \mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k}), \tag{9a}$$

$$q_{\theta_{t_k}}(\mathbf{w}) = \tilde{q}_{\theta_{t_k}}(\mathbf{w}) \,/ \prod_{\mathbf{d}_{t_k} \in \mathcal{M}_{t_k}} \mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k}). \tag{9b}$$

Note again that the natural parameters of $\mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k})$ are estimated using Monte Carlo and the products in the above equations imply a summation of the natural parameters. In order to reduce the variance of this sum of estimators, we use Eq. (9a) if $|\mathcal{D}_{t_k}| \leq |\mathcal{M}_{t_k}|$, and Eq. (9b) if $|\mathcal{D}_{t_k}| > |\mathcal{M}_{t_k}|$. Furthermore, we can compute the average bias from all natural parameter estimates (see App. C). We reduce the bias of our estimates by subtracting the average bias from all estimates. Note that a further option to update $q_{t_k}(\mathbf{w})$ would be to use VB on the data $\mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}} \setminus \mathcal{M}_{t_k}$ to compute the update $q_{\theta_{t_k}}(\mathbf{w}) \approx q_{\theta_{t_{k-1}}}(\mathbf{w})\, p(\mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}} \setminus \mathcal{M}_{t_k}|\mathbf{w})$. The latter approach is numerically more stable but computationally more expensive. It also turned out that it is less favorable to the update using Eq. (9a) or Eq. (9b) in case of small datasets $\mathcal{D}_{t_k}$, because VB applied to BNNs with small datasets often leads to a poor fit.

Previous work hypothesised that this problem is an artifact of the ELBO and not an optimisation problem (Trippe & Turner, 2018; Turner & M. Sahani, 2011). We provide further evidence in Fig. 1, where we infer the posterior of a Bayesian neural network with VB, using 70 and 100 data samples respectively and compare it to posterior inference with MCMC. In case of VB with 70 samples, the posterior approximation yields a model that is almost linear. These difficulties of posterior inference with variational Bayes are especially problematic in case of the streaming data setting, where the number of observations at each time-step is typically very small. The *Gaussian update* proposed above can alleaviate the problem of having to train BNNs with small datasets. Specifically, we have $N_{t_k} + M$ instead of N_{t_k} data points to find a better optimum of the ELBO.



(a) MCMC, 70 samples (b) VB, 70 samples (c) VB, 100 samples (d) ELBO terms during training

Figure 1: Posterior predictive distribution in the xy-plane (grey) of a Bayesian neural network with 2 layers of 16 units, tanh activations, prior $p_0(\mathbf{w}) = \mathcal{N}(\mathbf{w}; 0, 1)$, and Bernoulli likelihood. In case of variational Bayes (Figs. 1b, 1c), the KL divergence of the ELBO is annealed from $\beta = 0$ to $\beta = 1$ over many iterations (450k annealing, 50k ELBO). Fig. 1d shows that the approximation trades off the expected log-likelihood for a better KL divergence as β is increased. With 70 data points, the annealed KL jumps to a significantly lower value, resulting in an almost linear decision boundary. By contrast, MCMC yields a much better predictive distribution for the same number of samples. Data is visualised in red and blue.

4 Variational Bayes with Model Adaptation

The incremental learning methods discussed so far assume i.i.d. data (cf. Sec. 2, Eq. (1)). This assumption can be reasonable even in scenarios with changing data distributions, e.g. when the data drift is an algorithmic artifact rather than a real phenomenon. For example, in online multi-task or curriculum learning we want to learn a model of all tasks, but we may choose to learn the tasks incrementally for various reasons (e.g. Nguyen et al., 2018; Kirkpatrick et al., 2017; Schwarz et al., 2018; Rusu et al., 2016; Fernando et al., 2017). However, such approaches are not applicable for modeling non-stationary data: one of the properties of online VB is that the variance of the Gaussian posterior approximation shrinks at a rate of $O(N)$, where N is the total amount of data (e.g. Opper, 1998). Consequently, learning comes to a halt as $t \to \infty$. To overcome this issue, the model needs to be extended by a method that enables it to adapt to changes in the data distribution, e.g., by deliberately forgetting the belief inferred from previous data.

In the following, we describe two alternative methods for adapting to changing data. In Sec. 4.1, we impose Bayesian exponential forgetting, which forgets previous data exponentially by weighting the likelihood terms (or their approximations). In Sec. 4.2, we implement the adaptation through a diffusion process applied to the neural network parameters. Compared to the online learning scenario, we make the following assumptions: (i) we observe datasets $\mathcal{D}_{t_k}$ at potentially non-equidistant time steps t_k; (ii) data within $\mathcal{D}_{t_k}$ is assumed i.i.d., however, not between different datasets $\mathcal{D}_{t_k}$ and $\mathcal{D}_{t_{k+1}}$.

In both approaches, we realise adaptation by an additional *forgetting step* before observing the new data $\mathcal{D}_{t_{k+1}}$. We denote the distribution, which results from applying the *forgetting step* to the posterior approximation $q_{\theta_{t_k}}(\mathbf{w})\, p(\mathcal{M}_{t_k}|\mathbf{w})$ by $p_{t_{k+1}}(\mathbf{w})$.

4.1 Adaptation with Bayesian Forgetting

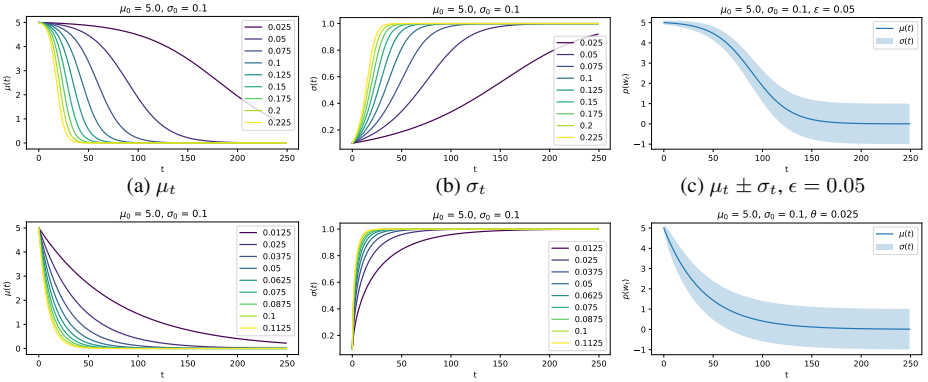
Model adaptation through forgetting can be achieved by decaying the likelihood based on the temporal recency of the data (Graepel et al., 2010; Honkela & Valpola, 2003). It has been explored previously as an alternative to filtering and is referred to as Bayesian exponential forgetting (Kulhavý & Zarrop, 1993). This approach defines a forgetting operator that yields $p(\mathbf{w}_{t_{k+1}}|\mathcal{D}_{t_1:t_k})$ directly. Here, we use a continuous-time version of this forgetting operation that can be formulated as

$$p(\mathbf{w}|\mathcal{D}_{t_1:t_K}) \propto p_0(\mathbf{w}) \prod_{k=1}^{K} p(\mathcal{D}_{t_k}|\mathbf{w})^{(1-\epsilon)^{\frac{t_K - t_k}{\tau}}}, \tag{10}$$

where τ is a time-constant corresponding to the average of the time-lags $\Delta t_{k+1} = t_{k+1} - t_k$. The distribution defined in Eq. (10) can be formulated recursively (cf. App. F) as

$$p(\mathbf{w}|\mathcal{D}_{t_1:t_{k+1}}) \propto p_0(\mathbf{w})^{1-(1-\epsilon)^{\Delta t_{k+1}/\tau}} p(\mathbf{w}|\mathcal{D}_{t_1:t_k})^{(1-\epsilon)^{\Delta t_{k+1}/\tau}} p(\mathcal{D}_{t_{k+1}}|\mathbf{w}). \tag{11}$$

This equation can be viewed as Bayes rule (Eq.(1)) applied after the *forgetting step*. The first two terms of Eq. (11) can be identified as the forgetting operation, applied to the current posterior. In



(a) μ_t (b) σ_t (c) $\mu_t \pm \sigma_t$, $\epsilon = 0.05$

(d) μ_t (e) σ_t (f) $\mu_t \pm \sigma_t$, $\theta = 0.025$

Figure 2: Time-evolution of distribution parameters of Bayesian Forgetting (top) and the Ornstein-Uhlenbeck process (bottom) for different adaptation parameter values. The initial distribution (at $t = 0$) can be seen as the approximate posterior at some time-step t_k.

order to apply this operation to our posterior approximation $q_{\theta_{t_k}}(\mathbf{w})\, p(\mathcal{M}_{t_k}|\mathbf{w})$, we modify it by an additional weighting factor for each likelihood term in the memory. Denoting the age of a memory item $\mathbf{m}$ by $\Delta t_k(\mathbf{m})$, the forgetting operation for this new posterior approximation then results in

$$p_{t_{k+1}}(\mathbf{w}) \propto p_0(\mathbf{w})^{1-(1-\epsilon)^{\Delta t_{k+1}/\tau}} \times \left[q_{\theta_{t_k}}(\mathbf{w}) \prod_{\mathbf{m}\in\mathcal{M}_{t_k}} p(\mathbf{m}|\mathbf{w})^{(1-\epsilon)^{\Delta t_k(\mathbf{m})/\tau}} \right]^{(1-\epsilon)^{\Delta t_{k+1}/\tau}}$$
$$= \left[p_0(\mathbf{w})^{1-(1-\epsilon)^{\Delta t_{k+1}/\tau}} q_{\theta_{t_k}}(\mathbf{w})^{(1-\epsilon)^{\Delta t_{k+1}/\tau}} \right] \times \prod_{\mathbf{m}\in\mathcal{M}_{t_k}} p(\mathbf{m}|\mathbf{w})^{(1-\epsilon)^{\Delta t_{k+1}(\mathbf{m})/\tau}}, \quad (12)$$

where $\Delta t_{k+1}(\mathbf{m}) = \Delta t_k(\mathbf{m}) + \Delta t_{k+1}$. As can be seen from Eq. (12), BF acts on both factors of the posterior approximation independently: in case of the memory, it re-weights the respective likelihood terms by updating $\Delta t_{k+1}(\mathbf{m})$. For the Gaussian term $q_{\theta_{t_k}}(\mathbf{w})$, BF leads to a weighted product with the prior distribution (i.e. the first two terms of Eq. (12)), resulting in a Gaussian with parameters

$$\sigma^{-2}_{t_{k+1}} = \left(1 - (1-\epsilon)^{\Delta t_{k+1}/\tau}\right)\sigma_0^{-2} + (1-\epsilon)^{\Delta t_{k+1}/\tau}\sigma^{-2}_{t_k},$$
$$\sigma^{-2}_{t_{k+1}}\mu_{t_{k+1}} = \left(1 - (1-\epsilon)^{\Delta t_{k+1}/\tau}\right)\sigma_0^{-2}\mu_0 + (1-\epsilon)^{\Delta t_{k+1}/\tau}\sigma^{-2}_{t_k}\mu_{t_k}.$$

For $\Delta t_{k+1} \to \infty$, the likelihood term in Eq. (12) converges to the uniform distribution and the Gaussian term reverts to the prior. We note, however, that while Eq. (11) is an exact recursive form of Eq. (10), the online VB approximation of Eq. (11) is not generally identical to the (offline) VB approximation of Eq. (10) due to its successive approximations. For tuning the hyperparameter ϵ, we note that the weighting of likelihood terms corresponds to an effective dataset size of $1/\epsilon \cdot N$ (if all datasets are of equal size N). In Fig. 2, we also visualise the forgetting operation applied to the Gaussian part of the posterior approximation for multiple values of ϵ.

4.2 Adaptation with Diffusion Processes

Model adaptation can also be realised by using dynamic model parameters that evolve according to a stochastic process. In this case, adaptation is achieved by the stochastic transition $p_{t_{k+1},t_k}(\mathbf{w}'|\mathbf{w})$ resulting in a prediction distribution

$$p_{t_{k+1}}(\mathbf{w}') = \int p_{t_{k+1},t_k}(\mathbf{w}'|\mathbf{w})\, p(\mathbf{w}|\mathcal{D}_{t_1:t_k})\, d\mathbf{w}, \quad (13)$$

where we consider Gaussian transitions $p_{t_{k+1},t_k}(\mathbf{w}'|\mathbf{w})$. However, this operation is generally not tractable for our posterior approximation $q_{\theta_{t_k}}(\mathbf{w})\, p(\mathcal{M}_{t_k}|\mathbf{w})$. Moreover, the forgetting operation implied by the transition does not retain the product form as in the case of BF. For this reason, we consider only a Gaussian posterior approximation (without memory) for this approach, that is $p_{t_{k+1}}(\mathbf{w}') = \int p_{t_{k+1},t_k}(\mathbf{w}'|\mathbf{w})\, q_{\theta_{t_k}}(\mathbf{w})\, d\mathbf{w}$.

As mentioned in Sec. 4.1, BF yields the prior distribution for $\Delta t_{k+1} \to \infty$. This is a desirable property, since it corresponds to forgetting all information conveyed by the data. In case of a Gaussian prior, the only Gaussian process that fulfills this requirement is the *Ornstein-Uhlenbeck* (OU) process given by the stochastic differential equation $d\mathbf{w}_t = \theta \cdot (\mu_0 - \mathbf{w}_t)\, dt + \sigma_0 \sqrt{2\theta}\, dW_t$, where θ is the stiffness parameter which controls the drift rate towards μ_0. To decouple the adaptation parameter from the rate at which data is observed, we rescale the stiffness parameter as $\theta = {}^{a}/_{\tau}$. The resulting prediction distribution $p_{t_{k+1}}(\mathbf{w}) = \mathcal{N}\big(\mu_{t_{k+1}},\, \sigma^2_{t_{k+1}}\big)$ is defined by the parameters

$$\mu_{t_{k+1}} = \big(1 - e^{-a\frac{\Delta t_{k+1}}{\tau}}\big)\mu_0 + e^{-a\frac{\Delta t_{k+1}}{\tau}}\mu_{t_k},$$
$$\sigma^2_{t_{k+1}} = \big(1 - e^{-2a\frac{\Delta t_{k+1}}{\tau}}\big)\sigma_0^2 + e^{-2a\frac{\Delta t_{k+1}}{\tau}}\,\sigma^2_{t_k}.$$

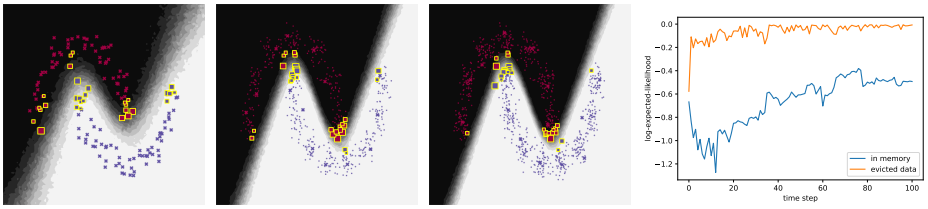
An interesting observation is that both parameters evolve independently of each other. In contrast to BF, the mean and variance—instead of the natural parameters—follow an exponential decay. The hyperparameter a can be determined e.g. through the half-time of the exponential decay of the mean parameter, given as $\tau_{1/2} = {}^{1}/_{\theta}$. We visualise the time evolution of the above parameters in Fig. 2.

5 Related Work

There are many Bayesian approaches to online learning, which differ mostly in the approximation of the posterior distribution at each time-step. Sequential Monte Carlo (Liu & Chen, 1998) approximates the posterior by a set of particles. Assumed Density Filtering (ADF) (Maybeck, 1982) and Bayesian online learning (Opper, 1998) are deterministic posterior approximations based on moment matching. Other deterministic approaches are based on Laplace's approximation (MacKay, 1992): Kirkpatrick et al. (2017) use multiple diagonal Gaussian posterior approximations of previous time-steps to regularise future tasks; Ritter et al. (2018) use a single (block-diagonal) posterior approximation, summarising all previous time-steps. The latter method is closer to Bayesian online inference, as it is an approximation of Eq. (1). Our work is based on online VB (Opper, 1998; Ghahramani, 2000; Sato, 2001; Broderick et al., 2013), which approximates the posterior at every time-step by minimising the KL-divergence between a parametric (here Gaussian) and the true posterior distribution. In contrast to online VB, we approximate the posterior by a Gaussian distribution and a running memory.

Other approaches are based on various types of episodic memory, motivated by their empirical success in preventing catastrophic forgetting. The basic idea of rehearsal (Ratcliff, 1990) is to train on both the new data and a subset of previous data or pseudo samples (Robins, 1995; Shin et al., 2017; Kemker & Kanan, 2017) sampled from a generative model. The memory-based online inference methods most similar to our approach are VCL (Nguyen et al., 2018) and VVM (Minka et al., 2009). Both methods use a Gaussian distribution and a running memory to approximate the posterior. VCL uses heuristics such as *random selection* or the *k-center method* to update the memory. However, both heuristics select the memory independently of the Gaussian approximation. By contrast, VVM updates the memory with data that cannot be well approximated by the Gaussian distribution. VVM uses expectation propagation for the posterior approximation in a logistic regression model and, therefore, it is not directly applicable to our work. We transferred the main idea of VVM to online VB and developed the corresponding *memory update* method. In our case, the memory is updated with data for which the ELBO changes most if the corresponding likelihood functions are approximated by a Gaussian. In contrast to these two approaches, we extend our model by an *adaptation method* that allows to cope with non-stationary data.

Many adaptation methods were developed in the context of *concept drift*, however, few of these approaches operate in the Bayesian framework. For example McInerney et al. (2015) treat the learning dynamics of their model as a non-stationary process that allows for adaptation. In contrast, our approach uses an evolving prior and a well defined forgetting mechanism that gives a better control over the learning process. A more closely related approach uses the extended Kalman filter to estimate the optimal parameters of a logistic regression classifier Su et al. (2008). However, they consider a transition model which is equivalent to a Wiener process (in unit-time) and therefore does not revert to the prior. By contrast, our approach (Sec. 4.2) models the dynamics as a prior-reverting OU process. BF (Kulhavý & Zarrop, 1993) has been applied as an alternative to adaptation with an explicit transition model (e.g. Honkela & Valpola, 2003; Graepel et al., 2010). Compared to previous work, we used a continuous-time version of BF and extended it to our posterior appoximation consisting of a Gaussian and a running memory.



(a) t_0 (b) t_{25} (c) t_{50} (d) LML

Figure 3: Two-moons dataset. Predictive distribution (Figs. 3a – 3c) of a BNN (gray) and running memory (rectangular shapes, size is proportional to the score), chosen by the *memory update* proposed in Sec. 3.2. Data from t_k and $t_{<k-1}$ is visualised as large circles and small dots, respectively. Fig. 3d shows the one-step-ahead (predictive) LML (divided by the number of samples) for data that will be selected for the memory and data that will be evicted. Data that will be selected in the memory tends to have a significantly lower predictive likelihood.

	GRS (ours)	k-center	random
Concrete	$\mathbf{-0.779 \pm 0.039}$	-0.798 ± 0.039	-0.800 ± 0.039
Boston	$\mathbf{-0.619 \pm 0.111}$	-0.638 ± 0.093	-0.664 ± 0.156
Energy	$\mathbf{0.365 \pm 0.440}$	-0.119 ± 0.128	-0.078 ± 0.087
Yacht	$\mathbf{1.925 \pm 0.229}$	1.658 ± 0.291	1.648 ± 0.254
Spam	$\mathbf{-0.216 \pm 0.016}$	-0.219 ± 0.015	-0.217 ± 0.016
Wine	$\mathbf{-1.165 \pm 0.056}$	-1.212 ± 0.059	-1.194 ± 0.070
MNIST	$\mathbf{-0.148 \pm 0.005}$	-0.158 ± 0.005	-0.153 ± 0.005

Table 1: average test LML, averaged over the last 10% time-steps. Mean and std. deviations are computed over 16 independent runs. The memory size is 15 for Concrete, Boston, Energy and Yacht, 25 for Spam and Wine, and 150 for MNIST. Bold indicates best (average) results.

6 Experiments

We validate our proposed inference methods in two stages. In Sec. 6.1, we compare our *memory update* and *Gaussian update* (Sec. 3) to existing memory-based online inference methods on several standard machine learning datasets. In Sec. 6.2, we evaluate our *adaptation methods* (Sec. 4) on commonly used datasets with *concept drift* (Widmer & Kubat, 1996), where the conditional distribution of labels given the features changes over time (i.e. non-stationary data in the context of predictive models).

We found that training (variational) Bayesian neural networks on streaming data is challenging, specifically, our approach requires model parameters very close to a local optimum since Eqs. (5a) and (5b) hold only at local extrema of the ELBO. To overcome these difficulties, we use several methods to reduce the variance of the gradient estimates for learning: (i) we apply the local reparametrisation trick (Kingma et al., 2015); (ii) we use the Adam optimiser (Kingma & Ba, 2014); and (iii) we use multiple Monte Carlo samples to estimate the gradients (cf. Tab. 2 for details). Furthermore, we developed methods for determining hyperparameters of the Gaussian prior and the initialisation distribution of Bayesian neural networks. The idea is similar to the initialisation method proposed by Glorot & Yoshua Bengio (2010) and He et al. (2015): we choose the prior and the posterior initialisation such that the mean and standard deviation of the activations in every layer are approximately zero and one, respectively. We refer to App. H and App. I for a derivation and further details.

We use the following metrics for evaluation: (i) the avg. test log-marginal likelihood (LML) $N_{\text{test}}^{-1} \sum_n \log \mathbb{E}_{\tilde{q}_{\theta_{t_k}}(\mathbf{w})} \left[p(\mathbf{d}_{\text{test}}^{(n)} | \mathbf{w}) \right]$, where $\mathbf{d}_{\text{test}}^{(n)}$ is a sample from a heldout test dataset; (ii) the avg. one-step-ahead LML $N_{t_{k+1}}^{-1} \sum_n \log \mathbb{E}_{\tilde{q}_{\theta_{t_k}}(\mathbf{w})} \left[p(\mathbf{d}_{t_{k+1}}^{(n)} | \mathbf{w}) \right]$, where $\mathbf{d}_{t_{k+1}}^{(n)}$ is data observed at time-step t_{k+1}. Both metrics measure the predictive performance, however (i) can be used in the online setting, where the data is i.i.d.; and (ii) is typically used to evaluate models with non-stationary streaming data.

6.1 Online Learning

In this section, we evaluate our running memory (Sec. 3) in an online learning setting. To illustrate how our *memory update* works, we start our evaluation with a qualitative assessment: we train a model on 2-dimensional toy data (two-moons), where we can visualise the selected memory. The BNN has 2 layers with 16 units and $tanh$ activations, and has a prior $p_0(\mathbf{w}) = \mathcal{N}(\mathbf{w}; 0, 1)$ on all

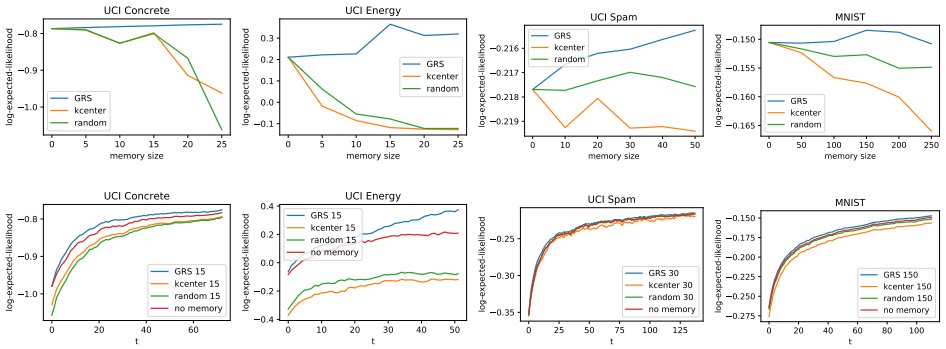


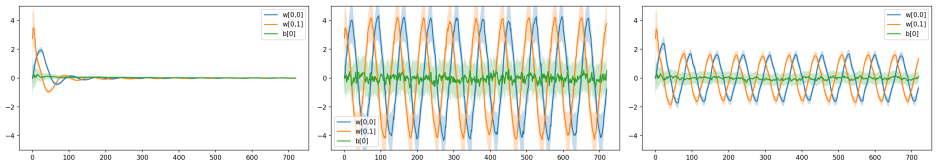
Figure 4: Average test LML, evaluated for several memory sizes (top), and evaluated over time (bottom) for a specific memory size (cf. corresponding legend). Cf. Sec. 6.1 for details and App. A for further results.

weights and biases. The memory-size is $M = 30$. The model observes 150 data samples at time-step t_0 and 15 samples at all consecutive time-steps. In Fig. 3, we visualise the selected memory and the corresponding scores for time-steps t_0, t_{25}, and t_{50}, respectively. We can make the empirical observation that our method favors data close to the decision boundary. Furthermore, in Fig. 3d, we visualise the one-step-ahead LML for data that will be selected and evicted (in the next time-step), respectively. This shows that our *memory update* tends to select data for which the model has a low predictive LML. These observations support our intuition that the memory is indeed complementary to the Gaussian approximation, selecting data for which the likelihood cannot be well approximated by a Gaussian function. In Fig. 9 of the supplementary material, we visualised the running memory for a model trained on MNIST, showing that the memory also accumulates diverse samples over time.

We evaluate our memory-based online inference method (Sec. 3) quantitatively on several standard machine learning datasets, including regression (UCI Boston, UCI Concrete, UCI Energy, UCI Yacht) and classification (MNIST, UCI Spam, UCI Wine) tasks. Here, we refer to our approach as Gaussian Residual Scoring (GRS). We compare GRS to the respective *memory update* and *Gaussian update* methods proposed in VCL (Nguyen et al., 2018) (cf. Sec. 2.2). Refer to App. B for an explanatory list of compared update methods. Online learning is performed by observing N_{t_k} samples per time-step (cf. Tab. 2 for the experiment setup and hyperparameters.). For evaluation, we use a random held-out test dataset (20% of the data). We perform each experiment with 16 different random data splits and random seeds for the model parameter initialisation. In Fig. 4, we plot the test LML, averaged over the 16 runs, against the memory size, and the LML over all time-steps. In most cases, *random selection* and the *k-center* method start with a worse initial fit at t_0. This is because these methods perform the initial *Gaussian update* by optimising the ELBO with $N_{t_0} - M$ samples at t_0; by contrast, *GRS* uses a *Gaussian update* that first optimises the ELBO with N_{t_0} samples and subsequently discounts the contribution of the memory. In Tab. 1, we report the mean and std. deviation of the LML, where the mean and std. deviation are taken over the 16 independent runs, each averaged over the last 10% time-steps. The results demonstrate the superior predictive performance of our update methods. We also note that the experiments on the smaller datasets (cf. Tab. 2 in App. B) result in a high variance among the random data splits and random seeds. This is the case for all compared methods and it could not be remedied e.g. by using annealing or a different prior.

6.2 Adaptation

In this section, we evaluate our adaptation methods (Sec. 4) in settings with *concept drift*. We begin with a simple logistic regression problem, where the data $\mathcal{D}_{t_k} = \{(\mathbf{x}_{t_k}, \mathbf{y}_{t_k})\}_n$, $\mathbf{x}_{t_k} \in \mathbb{R}^2$, $\mathbf{y}_{t_k} \in \mathbb{R}$ is sampled from $\mathbf{x}_{t_k} \sim \text{Uniform}(-3, 3)$, $\mathbf{y}_{t_k} \sim \text{Bernoulli}(\sigma(\mathbf{w}_{t_k}\mathbf{x}_{t_k}))$. The true model has two time-dependent parameters $\mathbf{w}^0_{t_k} = 10\sin(\alpha \cdot t_k)$, $\mathbf{w}^1_{t_k} = 10\cos(\alpha \cdot t_k)$, where $\alpha = 5\,\text{deg}/\text{sec}$ and where we observe data at $t_k \in [0,\ 1,\ \ldots\ , 720]$. Fig. 5 shows the learned model parameters for standard online learning (without adaptation), OU process transitions, and Bayesian forgetting. If the time-dependence of the data is ignored (in case of online VB), the class labels are distributed with equal probability in the whole input space. Consequently, as $t \to \infty$, the weights of the model without adaptation shrink to 0. By contrast, the posterior means of BF and the OU process follow a sinusoidal curve as the parameters of the true model.



(a) Online (no adaptation) (b) Ornstein-Uhlenbeck process (c) Bayesian forgetting

Figure 5: Mean and std deviation of the approximate posterior distributions of a logistic regression model over 720 time-steps. The model is trained on a toy classification problem with rotating class boundaries (cf. Sec. 6.2). Online VB (left) quickly converges to zero mean, whereas Bayesian forgetting and OU-process transitions lead to a sinusoidal curve as in the true model.

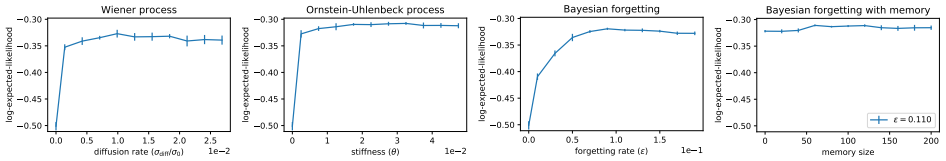


Figure 6: One-step ahead LML on Covertype dataset. Subplots show 3 different adaptation methods (3 left plots), evaluated for several values of the respective adaptation parameter, and Bayesian forgetting with $\epsilon = 0.11$, evaluated for multiple memory sizes (right).

We also evaluate our adaptation methods quantitatively on 3 datasets with *concept drift* (Weather, Gas Sensor Array Drift, Covertype). We compare online VB (without adaptation), the Wiener process (a special case of the OU process), the OU process, and Bayesian Forgetting (with and without memory). All compared variants use the same model architecture and hyperparameters (cf. Tab. 2 in the supplementary material). We report the one-step-ahead LML, where the expectation is approximated with 500 Monte Carlo samples. Results are averaged over the last 50% time-steps, because we are interested in the continual learning performance, and the first few time-steps will be similar for most methods. We report the mean and std. deviation over 8 independent runs with different random seeds. In Fig. 6 (and Fig. 10 in the appendix), we plot the LML against 10 adaptation parameter values (of the respective adaptation method), where the value zero corresponds to online VB. The LML for BF with different memory sizes and a fixed forgetting rate $\epsilon = 0.11$ is shown in Fig. 6. As can be seen from the results, all adaptation methods significantly improve the performance compared to online VB. Interestingly, the Ornstein-Uhlenbeck process performs better than Bayesian Forgetting, however, using a running memory with Bayesian Forgetting closes the gap.

7 Conclusion

In this work, we have addressed online inference for non-stationary streaming data using Bayesian neural networks. We have focused on posterior approximations consisting of a Gaussian distribution and a complementary running memory, and we have used variational Bayes to sequentially update the posteriors at each time-step. Existing methods update these two components without having an interaction between them, and they lack methods to adapt to non-stationary data. We have proposed a novel update method, which treats both components as complementary, and two novel adaptation methods (in the context of Bayesian neural networks with non-stationary data), which gradually revert to the prior distribution if no new data is observed.

Future research could extend our work by drift detection methods and use them to infer the adaptation parameters. This work could also be extended by developing adaptation methods for gradual, abrupt, or recurring changes in the data distribution. Finally, we observed that variational Bayesian neural networks with a uni-modal approximate posterior often find poor local minima if the dataset is small and models are complex. This is especially challenging in scenarios with streaming data. While our Gaussian update alleviates this problem to a certain degree, further research in extending the approximation family beyond Gaussians could be beneficial. Progress in this direction would improve our proposed methods and allow to scale them to more complex models.

REFERENCES

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, pp. 1613–1622, 2015.

T. Broderick, N. Boyd, A. Wibisono, A. C. Wilson, and M. I. Jordan. Streaming variational bayes. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pp. 1727–1735, USA, 2013. Curran Associates Inc.

Z. Chen and B. Liu. *Lifelong Machine Learning*. Morgan & Claypool Publishers, 2016.

B. Cseke, M. Opper, and G. Sanguinetti. Approximate inference in latent gaussian-markov models from continuous time observations. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26*, pp. 971–979. Curran Associates, Inc., 2013.

G. Ditzler, M. Roveri, C. Alippi, and R. Polikar. Learning in nonstationary environments: A survey. *Computational Intelligence Magazine, IEEE*, 10:12–25, 11 2015.

C. Fernando, D. Banarse, C. Blundell, Y. Zwols, D. Ha, A. A. Rusu, A. Pritzel, and D. Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *CoRR*, abs/1701.08734, 2017.

Z. Ghahramani. Online variational Bayesian learning. *NIPS Workshop on Online Learning*, 2000.

S. Ghosh, J. Yao, and F. Doshi-Velez. Structured variational learning of Bayesian neural networks with horseshoe priors. In J. Dy and A. Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 1744–1753, 2018.

X. Glorot and Y. Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Y. W Teh and M. Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-scale Bayesian click-through rate prediction for sponsored search advertising in Microsoft's Bing search engine. In *Proceedings of the 27th International Conference on Machine Learning, ICML*, pp. 13–20, USA, 2010. Omnipress.

A. Graves. Practical variational inference for neural networks. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 24*, pp. 2348–2356. Curran Associates, Inc., 2011.

S. Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science*, 11(1):23 – 63, 1987.

K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, Washington, DC, USA, 2015. IEEE Computer Society.

G. E. Hinton and D. van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, pp. 5–13, New York, NY, USA, 1993. ACM.

A. Honkela and H. Valpola. On-line variational bayesian learning. In *In Proc. of the 4th Int. Symp. on Independent Component Analysis and Blind Signal Separation*, pp. 803–808, 2003.

N. Kamra, U. Gupta, and Y. Liu. Deep Generative Dual Memory Network for Continual Learning. *arXiv e-prints*, art. arXiv:1710.10368, Oct 2017.

R. Kemker and C. Kanan. Fearnet: Brain-inspired model for incremental learning. *CoRR*, abs/1711.10563, 2017.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 28*, pp. 2575–2583. Curran Associates, Inc., 2015.

J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

D. A. Knowles and T. P. Minka. Non-conjugate variational message passing for multinomial and binary regression. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 24*, pp. 1701–1709. Curran Associates, Inc., 2011.

T. Kulhavý and M. B. Zarrop. On a general concept of forgetting. *International Journal of Control*, 58(4):905–924, 1993.

J. S. Liu and R. Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443):1032–1044, 1998.

D. Lopez-Paz and M. Ranzato. Gradient episodic memory for continual learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6470–6479, USA, 2017. Curran Associates Inc. ISBN 978-1-5108-6096-4.

D. J. C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Compututation*, 4(3):448–472, May 1992.

P. S. Maybeck. *Stochastic Models, Estimation and Control*. Mathematics in science and engineering. Academic Press, 1982.

J. McInerney, R. Ranganath, and D. Blei. The population posterior and bayesian modeling on streams. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 28*, pp. 1153–1161. Curran Associates, Inc., 2015.

T. P. Minka, R. Xiang, and Y. A. Qi. Virtual Vector Machine for Bayesian online classification. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, pp. 411–418, Arlington, Virginia, USA, 2009. AUAI Press.

C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

M. Opper. A Bayesian approach to on-line learning. In D. Saad (ed.), *On-line Learning in Neural Networks*, pp. 363–378. Cambridge University Press, New York, NY, USA, 1998. ISBN 0-521-65263-4.

M. Opper and C. Archambeau. The variational gaussian approximation revisited. *Neural Computation*, 21:786–92, 10 2008.

M. Opper and O. Winther. Expectation consistent free energies for approximate inference. In L. K. Saul, Y. Weiss, and L. Bottou (eds.), *Advances in Neural Information Processing Systems 17*, pp. 1001–1008. MIT Press, 2005.

G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter. Continual lifelong learning with neural networks: A review. *ArXiv*, abs/1802.07569, 2018.

R. Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97 2:285–308, 1990.

M. B. Ring. Child: A first step towards continual learning. *Machine Learning*, 28(1):77–104, 1997.

H. Ritter, A. Botev, and D. Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 3742–3752, USA, 2018. Curran Associates Inc.

A. Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell. Progressive neural networks. *CoRR*, abs/1606.04671, 2016.

P. Ruvolo and E. Eaton. ELLA: An efficient lifelong learning algorithm. In *In Proc. of the 30th International Conference on Machine Learning*, pp. 507–515, 2013.

M. Sato. Online model selection based on the variational bayes. *Neural Computation*, 13(7): 1649–1681, 2001.

J. Schwarz, J. Luketina, W. Czarnecki, A. Grabska-Barwinska, Y. W. Teh, R. Pascanu, and R. Hadsell. Progress & compress: A scalable framework for continual learning. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

H. Shin, J. K. Lee, J. Kim, and J. Kim. Continual learning with deep generative replay. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 2990–2999. Curran Associates, Inc., 2017.

D. Silver, Q. Yang, and L. Li. Lifelong machine learning systems: Beyond learning algorithms. In *AAAI Spring Symposium Series*, 2013.

B. Su, Shen, Y-D., and Xu, W. Modeling concept drift from the perspective of classifiers. In *2008 IEEE Conference on Cybernetics and Intelligent Systems*, pp. 1055–1060, Sep. 2008.

B. Trippe and R. Turner. Overpruning in Variational Bayesian Neural Networks. *arXiv e-prints*, pp. arXiv:1801.06230, Jan 2018.

R. E. Turner and M. M. Sahani. Two problems with variational expectation maximisation for time-series models. In D. Barber, T. Cemgil, and S. Chiappa (eds.), *Bayesian Time series models*, chapter 5, pp. 109–130. Cambridge University Press, 2011.

G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23:69–101, 1996.

8 APPENDIX

A FURTHER EXPERIMENTAL RESULTS

A.1 MEMORY

Here we provide additional experimental results for the *memory update* and *Gaussian update* from Sec. 3. We conducted experiments on 3 additional datasets (UCI Boston, UCI Yacht, UCI Red Wine). The influence of the memory size and the performance over time (for a specific memory size) are shown in Fig. 7 (corresponding to Fig. 4 in the main text).

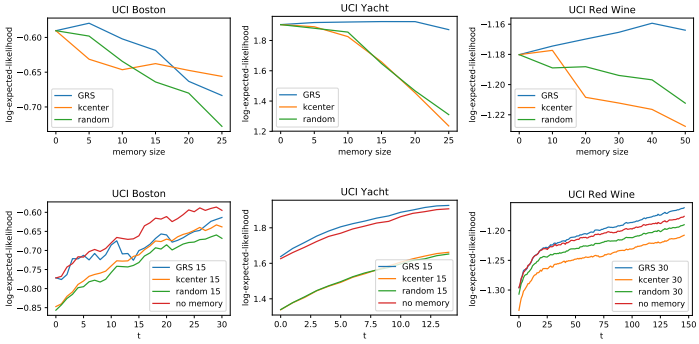


Figure 7: Average test LML on further datasets not included in the main text. Evaluated for several memory sizes (top), and evaluated over time (bottom) for a specific memory size. Cf. Sec. 6.1, App. A.1 for details.

Furthermore, we test the *memory update* and *Gaussian update* of GRS separately on UCI Energy and UCI Concrete. For this purpose, i) we combine the k-center method with our *Gaussian update* from Sec. 3.3; and ii) we use our *memory update* from Sec. 3.2 and update the Gaussian distribution by optimizing Eq. (2) with $\mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}} \backslash \mathcal{M}_{t_k}$ (re-fitting) . The results are shown in Fig. 8. As can be seen, GRS performs better than one of the components used in combination with a baseline method. *GRS with refit* performs especially bad, similar to the baselines *k-center* and *random*. This is because refitting requires optimising the ELBO with a small dataset. As mentioned in Sec. 3.3 (cf. Fig. 1), Bayesian neural networks with VB perform bad on small datasets due to over-regularisation. Consequently, in case of refitting, a good *memory update* can lead to a worse overall performance due to a much worse *Gaussian update*. While this general issue with Bayesian neural networks (learned with VB) is beyond the scope of this work, it is an important future research direction.

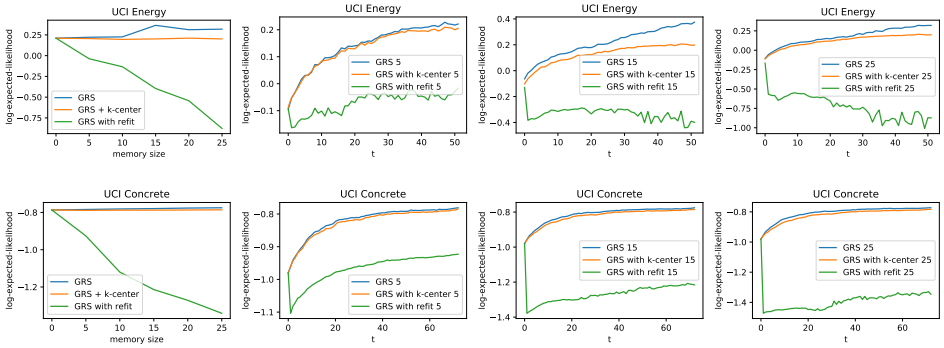
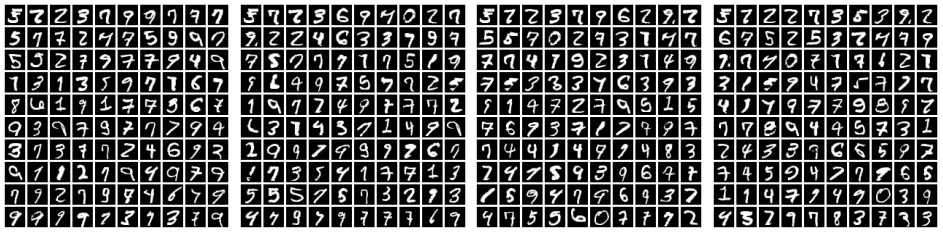


Figure 8: Average test LML on UCI Energy (top) and UCI Concrete (bottom). *GRS* denotes our approach (Sec. 3), *GRS with k-center* replaces our *memory update* by the k-center method; *GRS with refit* replaces our *Gaussian update* by the optimization of Eq. (2) with $\mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}} \backslash \mathcal{M}_{t_k}$. Evaluated for several memory sizes (left), and evaluated over time (right) for 3 different memory sizes. Hyperparameters are chosen as in Sec. 6.1.

To better understand our *memory update* using the score function from Eq. (8), we visualise the running memory for a model trained on MNIST in Fig. 9.



(a) t_{25} (b) t_{50} (c) t_{75} (d) t_{100}

Figure 9: Running memory at different time-steps on MNIST (cf. Sec. 6.1), with a memory size $N = 100$. The *memory update* tends to select non-typical data while showing diversity.

A.2 Adaptation

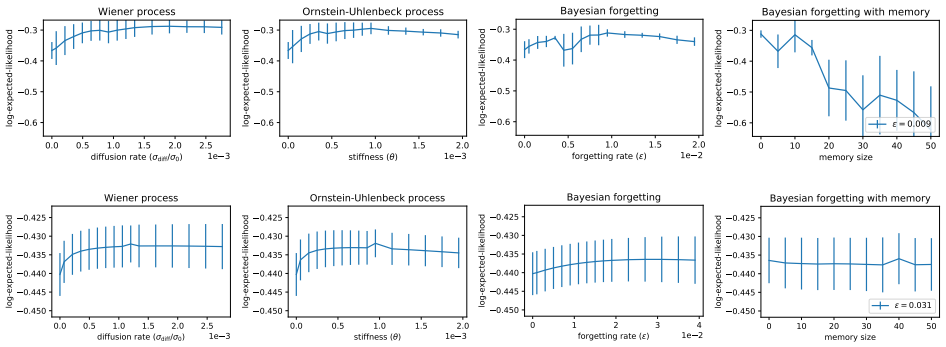
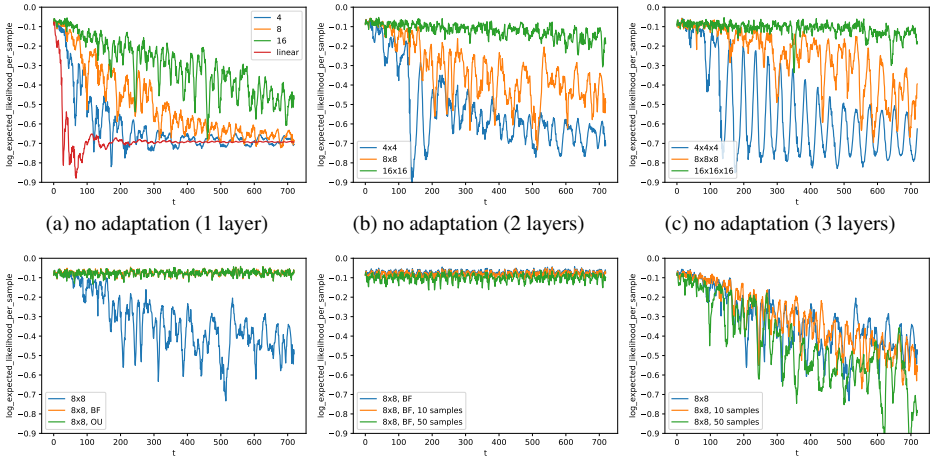


Figure 10: One-step ahead LML on Gas Sensor Array Drift dataset (top) and Weather dataset (bottom). Subplots show 3 different adaptation methods (left), evaluated for several values of the respective adaptation parameter, and Bayesian forgetting with $\epsilon = 0.0095$ (Gas Sensory Array Drift) and $\epsilon = 0.031$ (Weather), evaluated for multiple memory sizes (right).

We also evaluated our adaptation methods on 2 additional datasets (Gas Sensor Array Drift, Weather). In Fig. 10, we visualise the influence of the adaptation parameter for these datasets. Note that the range of the adaptation parameters is on a much smaller range compared to the experiments on Covertype (Sec. 6.2). For larger values, the performance starts to degrade. Surprisingly, the memory degrades the performance in case of the Gas Sensor Array Drift dataset.

A.3 Catastrophic Forgetting with Online VB and Bayesian Neural Networks

Here we provide further experimental results for the behavior of online VB (Secs. 2.1, 3) in case of non-stationary data. For this purpose, we train Bayesian neural networks with different architectures on the toy classification problem with a rotating decision boundary from Sec. 6.2, however, with 150 data samples per time-step. In Fig. 11, we visualise the training LML for different architectures, including a linear model. It can be seen that Bayesian neural networks with higher complexity (i.e. more layers or more units) drop slower in performance compared to the linear model. However, this is not a desired property for online VB, since exact online Bayesian inference would yield the same posterior distribution as offline Bayesian inference. In case of our toy classification data (where the time dependence is ignored), online inference should not be able to classify the data as $t \to \infty$. Instead, this learning behavior shows that online VB with Gaussian approximate posterior distributions is prone to catastrophic forgetting.



(a) no adaptation (1 layer) (b) no adaptation (2 layers) (c) no adaptation (3 layers)

(d) BF and OU (2 layers) (e) BF with memory (f) no adaptation with memory

Figure 11: LML for toy classification problem with rotating class boundaries (cf.Sec. A)

B Experiment Setup

The following is an explanatory list of the update methods used in Sec. 6.1:

- k-center (VCL): Uses the k-center method (Sec.2.2) for the *memory update* and Eq. (2) with $(\mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}}) \setminus \mathcal{M}_{t_k}$ for the *Gaussian update*.
- random (VCL): Uses random selection (Sec.2.2) for the *memory update* and Eq. (2) with $(\mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}}) \setminus \mathcal{M}_{t_k}$ for the *Gaussian update*.
- GRS (Gaussian Residual Scoring, ours): Uses Eq. (8) for the *memory update* (Sec. 3.2) and performs the *Gaussian update* by first using Eqs. (2) with $(\mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}})$ and subsequently using Eqs. (5a), (5b) for removing the local contributions of $\mathcal{M}_{t_k}$ (cf. Sec. 3.3).

Similarly, the following list summarises the adaptation methods used in Sec. 6.2:

- Wiener process: Posterior approximation consists of $q_{\theta_{t_k}}(\mathbf{w})$ only. Transition $p(\mathbf{w}_{t_{k+1}}|\mathbf{w}_{t_k})$ is given by a *random walk*. We used a diffusion that is proportional to the prior standard deviation in every neural network layer (cf. Sec. 4.2). No memory used.
- Ornstein-Uhlenbeck process: Posterior approximation consists of $q_{\theta_{t_k}}(\mathbf{w})$ only. Transition $p(\mathbf{w}_{t_{k+1}}|\mathbf{w}_{t_k})$ is given by the *Ornstein-Uhlenbeck* process (cf. Sec. 4.2). No memory used.
- Bayesian forgetting: Posterior approximation consists of $q_{\theta_{t_k}}(\mathbf{w})$ only. No state-space model assumption, instead uses Bayesian exponential forgetting (cf. Sec. 4.1).
- Bayesian forgetting with memory: Posterior approximation consists of $q_{\theta_{t_k}}(\mathbf{w})$ and $\mathcal{M}_t$. No state-space model assumption, instead uses Bayesian exponential forgetting (cf. Sec. 4.1).

In Tab. 2, we summarise experimental setup (hyperparameters) used for Secs. 6.1 and 6.2.

C Factorisation Property of the Gaussian Variational Approximation

Here we derive the factorisation property of the Gaussian variational approximation distribution by expressing the natural parameters of the Gaussian approximation as a sum. This can be shown for the Gaussian approximation at a local optimum of the ELBO. For a Gaussian prior and posterior the

Table 2: Experiment setup for online experiments. N_{t_0} is the number of observed samples at the first time-step and $N_{t_{1:k}}$ is the dataset size of all other time-steps. M refers to the number of samples in the memory. We evaluated 5 different memory sizes for each experiment in Sec. 3 and 10 sizes for experiments in Sec. 4, where the sizes are equally spaced in the given range. K_{train} and K_{term} is the number of MC samples used for training and for estimating the Gaussian terms respectively. I_{t_0} and $I_{t_{1:K}}$ refer to the number of iterations for the first time-step and all subsequent time-steps, respectively. The architecture denotes the number of units for each hidden layer (e.g. $[16, 16]$ denotes 2 hidden layers with 16 units each).

	N_{t_0}	$N_{t_{1:k}}$	M	K_{train}	K_{term}	I_{t_0}	$I_{t_{1:K}}$	Architecture
Concrete	100	10	$[5 \dots 25]$	1k	50k	50k	10k	$[16, 16]$
Boston	100	10	$[5 \dots 25]$	1k	50k	50k	10k	$[16, 16]$
Energy	100	10	$[5 \dots 25]$	1k	50k	50k	10k	$[16, 16]$
Yacht	100	10	$[5 \dots 25]$	1k	50k	50k	10k	$[16, 16]$
Spam	250	25	$[10 \dots 50]$	400	50k	50k	10k	$[32, 32]$
Wine	250	25	$[10 \dots 50]$	400	50k	50k	10k	$[32, 32]$
MNIST	2.5k	250	$[50 \dots 250]$	40	50k	50k	10k	$[64, 64]$
GasSensorArray	500	50	$[5 \dots 50]$	200	100k	50k	10k	$[8, 8]$
Weather	1k	100	$[5 \dots 50]$	100	100k	50k	10k	$[16, 16]$
Covertype	1k	1k	$[10 \dots 200]$	400	100k	50k	10k	$[32, 32]$

ELBO is given as

$$\mathcal{L}(\mu^*, \Sigma^*) = -\frac{1}{2}\Big(\log|\Sigma_0| - \log|\Sigma^*| - d + (\mu^* - \mu_0)^T \Sigma_0^{-1}(\mu^* - \mu_0) + Tr(\Sigma^* \Sigma_0^{-1}) \Big) + \sum_{n=1}^{N} \mathbb{E}_{\mathbf{w} \sim q_{\theta^*}(\mathbf{w})} \Big[\log p(\mathbf{d}^{(n)}|\mathbf{w}) \Big].$$

At a local optimum, we have $\frac{\partial \mathcal{L}(\mu^*, \Sigma^*)}{\partial \mu^*} = 0$, which yields

$$\sum_{n=1}^{N} \frac{\partial}{\partial \mu^*} \mathbb{E}_{\mathbf{w} \sim q_{\theta^*}(\mathbf{w})} \Big[\log p(\mathbf{d}^{(n)}|\mathbf{w}) \Big] = \Sigma_0^{-1}(\mu^* - \mu_0).$$

Hence, we obtain

$$\mu^* = \mu_0 + \Sigma_0 \sum_{n=1}^{N} \frac{\partial}{\partial \mu^*} \mathbb{E}_{\mathbf{w} \sim q_{\theta^*}(\mathbf{w})} \Big[\log p(\mathbf{d}^{(n)}|\mathbf{w}) \Big]. \tag{14}$$

Similarly, we have $\frac{\partial \mathcal{L}(\mu^*, \Sigma^*)}{\partial \Sigma^*} = 0$, which yields

$$\sum_{n=1}^{N} \frac{\partial}{\partial \Sigma^*} \mathbb{E}_{\mathbf{w} \sim q_{\theta^*}(\mathbf{w})} \Big[\log p(\mathbf{d}^{(n)}|\mathbf{w}) \Big] = -\frac{1}{2}(\Sigma^*)^{-1} + \frac{1}{2}\Sigma_0^{-1}. \tag{15}$$

Hence, we obtain

$$\Sigma^* = \Big(\Sigma_0^{-1} - 2 \sum_{n=1}^{N} \frac{\partial}{\partial \Sigma^*} \mathbb{E}_{\mathbf{w} \sim q_{\theta^*}(\mathbf{w})} \Big[\log p(\mathbf{d}^{(n)}|\mathbf{w}) \Big] \Big)^{-1}. \tag{16}$$

Next, we calculate the natural parameters from Eqs. (14), (16):

$$\begin{aligned} \Lambda^* &= \Lambda_0 + \sum_{n=1}^{N} -2 \frac{\partial}{\partial \Sigma^*} \mathbb{E}_{\mathbf{w} \sim q_{\theta^*}(\mathbf{w})} \Big[\log p(\mathbf{d}^{(n)}|\mathbf{w}) \Big] \\ &= \Lambda_0 + \sum_{n=1}^{N} \Lambda^{(n)}. \end{aligned}$$

$$\begin{aligned}
\eta^* = \Lambda^*\mu^* &= \Big(\Lambda_0 + \sum_{n=1}^{N}\Lambda^{(n)}\Big)\mu^* \\
&= \Lambda_0\Big(\mu_0 + \Sigma_0\sum_{n=1}^{N}\frac{\partial}{\partial\mu^*}\mathbb{E}_{\mathbf{w}\sim q_{\theta^*}(\mathbf{w})}\Big[\log p(\mathbf{d}^{(n)}|\mathbf{w})\Big]\Big) + \sum_{n=1}^{N}\Lambda^{(n)}\mu^* \\
&= \Lambda_0\mu_0 + \sum_{n=1}^{N}\Big(\frac{\partial}{\partial\mu^*}\mathbb{E}_{\mathbf{w}\sim q_{\theta^*}(\mathbf{w})}\Big[\log p(\mathbf{d}^{(n)}|\mathbf{w})\Big] + \Lambda^{(n)}\mu^*\Big) \\
&= \eta_0 + \sum_{n=1}^{N}\eta^{(n)}.
\end{aligned}$$

Monte Carlo estimation: The natural parameters $\Lambda^{(n)}$, $\eta^{(n)}$ can be estimated with Monte Carlo, by replacing the expectation with an empirical mean. Since the parameters Λ^* and Λ_0 (and η^*, η_0 respectively) are known, the total bias of the parameter estimates can be computed:

$$\Lambda_\mathrm{b} = (\Lambda^* - \Lambda_0) - \sum_{n=1}^{N}\Lambda^{(n)}, \qquad\qquad \eta_\mathrm{b} = (\eta^* - \eta_0) - \sum_{n=1}^{N}\eta^{(n)}.$$

We use this to reduce the bias for the individual terms:

$$\hat{\Lambda}^{(n)} = \Lambda^{(n)} - \frac{1}{N}\Lambda_\mathrm{b}, \qquad\qquad \hat{\eta}^{(n)} = \eta^{(n)} - \frac{1}{N}\eta_\mathrm{b}.$$

D ELBO IN RESIDUALS FORM

Here we show how the ELBO can be written in the form of Eq. (6). Let us define the variational distribution in the factorised form $q_\theta(\mathbf{w}) = Z_q^{-1}p(\mathbf{w})\prod_{n=1}^{N}\mathbf{r}^{(n)}(\mathbf{w})$ (cf. Sec.3.1). We can then write the ELBO as

$$\begin{aligned}
\mathcal{L}(\mu,\Sigma;\mathcal{D}) &= \mathbb{E}_{q_\theta(\mathbf{w})}\left[\sum_{n=1}^{N}\log p(\mathbf{d}^{(n)}|\mathbf{w}) + \log p(\mathbf{w}) - \log q_\theta(\mathbf{w})\right] \\
&= \mathbb{E}_{q_\theta(\mathbf{w})}\left[\sum_{n=1}^{N}\big(\log p(\mathbf{d}^{(n)}|\mathbf{w}) + \log\mathbf{r}^{(n)}(\mathbf{w}) - \log\mathbf{r}^{(n)}(\mathbf{w})\big) + \log p(\mathbf{w}) - \log q_\theta(\mathbf{w})\right] \\
&= \mathbb{E}_{q_\theta(\mathbf{w})}\left[\sum_{n=1}^{N}\big(\log p(\mathbf{d}^{(n)}|\mathbf{w}) - \log\mathbf{r}^{(n)}(\mathbf{w})\big) + \log\frac{p(\mathbf{w})\prod_{n=1}^{N}\mathbf{r}^{(n)}(\mathbf{w})}{q_\theta(\mathbf{w})}\right] \\
&= \mathbb{E}_{q_\theta(\mathbf{w})}\left[\sum_{n=1}^{N}\big(\log p(\mathbf{d}^{(n)}|\mathbf{w}) - \log\mathbf{r}^{(n)}(\mathbf{w})\big) + \log\frac{Z_q\cdot q_\theta(\mathbf{w})}{q_\theta(\mathbf{w})}\right] \\
&= \log Z_q + \sum_{n}\mathbb{E}_{q_\theta(\mathbf{w})}\big[\log p(\mathbf{d}^{(n)}|\mathbf{w}) - \log\mathbf{r}^{(n)}(\mathbf{w})\big].
\end{aligned}$$

E MEMORY UPDATE SCORE FUNCTION

In Eq. 8, the expectation involving Gaussian terms can be calculated analytically:

$$\begin{aligned}
\mathbb{E}_{\tilde{q}_{\theta_{t_k}}(\mathbf{w})}\Big[\log\mathbf{r}_{t_k}^{(m)}(\mathbf{w})\Big] &= \int\tilde{q}_{\theta_{t_k}}(\mathbf{w})\big(\eta^{(n)}\mathbf{w} - \frac{1}{2}\Lambda^{(n)}\mathbf{w}^2\big)d\mathbf{w} \\
&= \eta^{(n)}\mu^{(n)} - \frac{1}{2}\Lambda^{(n)}\big((\mu^*)^2 + \Sigma^*\big) \\
&= \eta^{(n)}(\Lambda^*)^{-1}\eta^* - \frac{1}{2}\Lambda^{(n)}\big((\Lambda^*)^{-1}\eta^*\big)^2 - \frac{1}{2}\Lambda^{(n)}(\Lambda^*)^{-1}
\end{aligned}$$

The expectation involving non-Gaussian terms (in Eq. 8) has no closed-form solution. We therefore estimate $\mathbb{E}_{\tilde{q}_{\theta_{t_k}}(\mathbf{w})}\big[\log p(\mathbf{d}_{t_k}^{(n)}|\mathbf{w})\big]$ using Monte-Carlo.

F BAYESIAN FORGETTING - RECURSIVE FORMULATION

Here we show how Bayesian forgetting can be rearranged into a recursive formulation. We first bring this formula into a similar form as Eq. (1), extracting the most recent likelihood term:

$$\begin{aligned} p(\mathbf{w}|\mathcal{D}_{t_1:t_{K+1}}) &\propto p_0(\mathbf{w}) \cdot \prod_{k=1}^{K+1} p(\mathcal{D}_{t_k}|\mathbf{w})^{(1-\epsilon)^{\frac{t_{K+1}-t_k}{\tau}}} \\ &= p_0(\mathbf{w}) \cdot \prod_{k=1}^{K} p(\mathcal{D}_{t_k}|\mathbf{w})^{(1-\epsilon)^{\frac{t_{K+1}-t_k}{\tau}}} \cdot p(\mathcal{D}_{t_{K+1}}|\mathbf{w}). \end{aligned}$$

The first two terms can be rewritten as

$$\begin{aligned} p_0(\mathbf{w}) \cdot \prod_{k=1}^{K} p(\mathcal{D}_{t_k}|\mathbf{w})^{(1-\epsilon)^{\frac{t_{K+1}-t_k}{\tau}}} &= p_0(\mathbf{w}) \cdot \prod_{k=1}^{K} p(\mathcal{D}_{t_k}|\mathbf{w})^{(1-\epsilon)^{\frac{t_{K+1}-t_K+t_K-t_k}{\tau}}} \\ &= p_0(\mathbf{w}) \cdot \prod_{k=1}^{K} p(\mathcal{D}_{t_k}|\mathbf{w})^{(1-\epsilon)^{\frac{t_K-t_k}{\tau}} \cdot (1-\epsilon)^{\frac{t_{K+1}-t_K}{\tau}}} \\ &= p_0(\mathbf{w}) \cdot \left(\prod_{k=1}^{K} p(\mathcal{D}_{t_k}|\mathbf{w})^{(1-\epsilon)^{\frac{t_K-t_k}{\tau}}} \right)^{(1-\epsilon)^{\frac{t_{K+1}-t_K}{\tau}}} \\ &\propto p_0(\mathbf{w}) \cdot \left(\frac{p(\mathbf{w}|\mathcal{D}_{t_1:t_K})}{p_0(\mathbf{w})} \right)^{(1-\epsilon)^{\frac{t_{K+1}-t_K}{\tau}}} \\ &= p_0(\mathbf{w})^{1-(1-\epsilon)^{\frac{t_{K+1}-t_K}{\tau}}} \cdot p(\mathbf{w}|\mathcal{D}_{t_1:t_K})^{(1-\epsilon)^{\frac{t_{K+1}-t_K}{\tau}}} \end{aligned}$$

Hence, we have shown that the posterior can be expressed recursively as

$$p(\mathbf{w}|\mathcal{D}_{t_1:t_{k+1}}) \propto p_0(\mathbf{w})^{1-(1-\epsilon)^{\Delta t_{k+1}/\tau}} p(\mathbf{w}|\mathcal{D}_{t_1:t_k})^{(1-\epsilon)^{\Delta t_{k+1}/\tau}} p(\mathcal{D}_{t_{k+1}}|\mathbf{w}).$$

The parameters of the Gaussian part $q_{\theta_{t_k}}(\mathbf{w})$ of the posterior approximation (after applying the forgetting operation) can be calculated easily from the above equation.

Natural parameters:

$$\Lambda_{t_{k+1}} = \Lambda_0 \cdot (1 - (1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}}) + \Lambda_{t_k} \cdot ((1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}})$$

$$\eta_{t_{k+1}} = \eta_0 \cdot (1 - (1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}}) + \eta_{t_k} \cdot ((1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}})$$

Covariance parameter:

$$\begin{aligned} \Sigma_{t_{k+1}} &= \left(\Lambda_0 \cdot (1 - (1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}}) + \Lambda_{t_k} \cdot ((1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}}) \right)^{-1} \\ &= \left(\Sigma_0^{-1} \cdot (1 - (1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}}) + \Sigma_{t_k}^{-1} \cdot ((1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}}) \right)^{-1} \end{aligned}$$

Location parameter:

$$\begin{aligned} \mu_{t_{k+1}} &= \Sigma_{t_{k+1}} \cdot \eta_{t_{k+1}} \\ &= \Sigma_{t_{k+1}} \left(\eta_0 \cdot (1 - (1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}}) + \eta_{t_k} \cdot ((1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}}) \right) \\ &= \Sigma_{t_{k+1}} \left(\Sigma_0^{-1}\mu_0 \cdot (1 - (1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}}) + \Sigma_{t_k}^{-1}\mu_{t_k} \cdot ((1-\epsilon)^{\frac{t_{k+1}-t_k}{\tau}}) \right) \end{aligned}$$

G PSEUDO-ALGORITHM

We provide the pseudo algorithm of GRS (Sec. 3) with Bayesian forgetting in Alg. 1. The computational complexity (at each of the K time-steps) is dominated by i) the minimisation of the KL

divergence, ii) estimating the Gaussian factors, and iii) scoring the memory. The KL minimisation requires I_{t_k} sequential iterations with $N_{t_k} + M$ data samples and K_{train} Monte Carlo samples. The latter can both be processed in parallel on parallel hardware. The estimation of the Gaussian factors requires $N_{t_k} + M$ sequential iterations with K_{term} parallel Monte Carlo samples. The dominating computation for calculating the scores is the evaluation of the likelihood for $N_{t_k} + M$ data samples and K_{train} Monte Carlo samples, both of which can be processed in parallel. The highest scoring candidate memory is given by the top-M highest scoring data points, thus, the computational complexity of Eq. (8) is only linear in the number of samples.

Algorithm 1 Gaussian Residual Scoring with Bayesian forgetting. The function *EstimateGaussianFactors* corresponds to Eqs. (5a), (5b) (cf. also App. C). The function *ApplyForgetting* corresponds to Eq. (12). Note that $p_{t_k}(\mathbf{w})$ includes the adapted likelihood of the memory and all subsequent functions involving the memory use this adapted likelihood.

Inputs: $p_0(\mathbf{w})$, $q_{\theta_{t_0}}(\mathbf{w})$, τ, K
for k in $[0...K]$ **do**
  $t_k = \text{GetTimeStamp}(k)$
  $\Delta t_k = t_k - t_{k-1}$
  $\mathcal{D}_{t_k} = \text{GetData}(t_k)$
  if $k == 0$ **then**
    $p_{t_k}(\mathbf{w}) = p_0(\mathbf{w})$
  else
    $p_{t_k}(\mathbf{w}) = \text{ApplyForgetting}\big(p_0(\mathbf{w}), q_{\theta_{t_{k-1}}}, \mathcal{M}_{t_{k-1}}, \Delta t_k\big)$ {Sec. 4.1}
  end if
  $\tilde{q}_{\theta_{t_k}}(\mathbf{w}) = \text{argmin}_{q_\theta} \text{KL}\big[q_\theta(\mathbf{w}) \parallel \tilde{Z}_{t_k}^{-1} p_{t_k}(\mathbf{w})\, p(\mathcal{D}_{t_k}|\mathbf{w})\big]$ {Sec. 3.2, Sec. 4.1}
  $\{\mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k})\}_{\mathbf{d}_{t_k} \in \mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}}} = \text{EstimateGaussianFactors}\big(\tilde{q}_{\theta_{t_k}}(\mathbf{w}), \mathcal{D}_{t_k}, \mathcal{M}_{t_{k-1}}\big)$ {Sec. 3.1}
  $\mathcal{M}_{t_k} = \text{argmax}_{\mathcal{M}}\ S_{t_k}(\mathcal{M}; \mathcal{D}_{t_k} \cup \mathcal{M}_{t_{k-1}})$ {Sec. 3.2}
  if $|\mathcal{D}_{t_k}| \leq |\mathcal{M}_{t_k}|$ **then**
    $q_{\theta_{t_k}}(\mathbf{w}) = p_{t_k}(\mathbf{w}) \prod_{\mathbf{d}_{t_k} \notin \mathcal{M}_{t_k}} \mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k})$ {Sec. 3.3}
  else
    $q_{\theta_{t_k}}(\mathbf{w}) = \tilde{q}_{\theta_{t_k}}(\mathbf{w}) \;/\; \prod_{\mathbf{d}_{t_k} \in \mathcal{M}_{t_k}} \mathbf{r}_{t_k}(\mathbf{w}; \mathbf{d}_{t_k})$ {Sec. 3.3}
  end if
end for

H PRIOR PARAMETERS

Here we develop a heuristic to choose the initial prior $p_0(\mathbf{w})$. As this will not be specific to the online or continual setting, we drop the time index in this section, denoting the prior as $p(\mathbf{w})$. Furthermore, we consider only Gaussian distributions with a diagonal covariance matrix. Assume that the data is standardised, that is, the first two moments are zero and one. A reasonable choice for the prior parameters is such that the first two moments of the prior-predictive distribution equals the first two moments of the data distribution. We go one step further and constrain the pre-activations of every neural network layer to have moments zero and one. Denote all weight matrices and weight biases by $\mathbf{w} = \{\mathbf{W}_l\}_l \cup \{\mathbf{b}_l\}_l$, and let $\mathbf{x}_0$ denote the input data. Let us further denote the pre-activation (before non-linearity) of layer l and unit i as follows.

$$\mathbf{x}_l^i = \sum_j \mathbf{W}_l^{i,j} f_{l-1}(\mathbf{x}_{l-1}^j) + \mathbf{b}_l^i.$$

The constraints are then given as follows.

$$\mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\Big[\mathbb{E}_{\mathbf{x}_0 \sim p(\mathbb{D})}\big[\mathbf{x}_l^i\big]\Big] = 0, \qquad \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\Big[\mathbb{E}_{\mathbf{x}_0 \sim p(\mathbb{D})}\big[(\mathbf{x}_l^i)^2\big]\Big] = 1.$$

The first constraint can be easily fulfilled by setting the prior mean to zero for all parameters.

$$\mu_l^{i,j} = 0.$$

This follows immediately from $\mathbf{W}_l \perp\!\!\!\perp f_{l-1}(\mathbf{x}_{l-1})$ and the expectation of products of independent random variables. The second moment can then be calculated as follows.

$$\begin{aligned}
\mathbb{E}_{\mathbf{w}\sim p(\mathbf{w}),\mathbf{x}_0\sim p(\mathbb{D})}\left[(\mathbf{x}_l^i)^2\right] &= \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\left[\mathbb{E}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[(\mathbf{x}_l^i)^2\right]\right] \\
&= \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\left[\mathrm{Var}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[\mathbf{x}_l^i\right] + 0\right] \\
&= \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\left[\mathrm{Var}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[\sum_j^{N_{l-1}} \mathbf{W}_l^{i,j} f_{l-1}(\mathbf{x}_{l-1}^j) + \mathbf{b}_l^i\right]\right] \\
&= \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\left[\sum_j^{N_{l-1}} \left(\mathbf{W}_l^{i,j}\right)^2 \cdot \mathrm{Var}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[f_{l-1}(\mathbf{x}_{l-1}^j)\right]\right] \\
&= \sum_j^{N_{l-1}} \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\left[\left(\mathbf{W}_l^{i,j}\right)^2 \cdot \mathrm{Var}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[f_{l-1}(\mathbf{x}_{l-1}^j)\right]\right] \\
&= \sum_j^{N_{l-1}} \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\left[\left(\mathbf{W}_l^{i,j}\right)^2\right] \cdot \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\left[\mathrm{Var}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[f_{l-1}(\mathbf{x}_{l-1}^j)\right]\right] \\
&=: \sum_j^{N_{l-1}} \mathbb{E}_{\mathbf{w}\sim p(\mathbf{w})}\left[\left(\mathbf{W}_l^{i,j}\right)^2\right] \cdot c_{f_{l-1}} \\
&= c_{f_{l-1}} \cdot \sum_j^{N_{l-1}} \mathrm{Var}_{\mathbf{w}\sim p(\mathbf{w})}\left[\mathbf{W}_l^{i,j}\right] \\
&= N_{l-1} \cdot c_{f_{l-1}} \cdot \mathrm{Var}_{\mathbf{w}\sim p(\mathbf{w})}\left[\mathbf{W}_l^{i,j}\right].
\end{aligned}$$

Here we introduced $c_{f_{l-1}}$ to denote a correction factor for the non-linearity f_{l-1}. In case of the linear function, we will have $c_{f_{l-1}} = 1$. For arbitrary non-linearities, we can estimate this factor numerically, assuming that the pre-activations are distributed according to $\mathcal{N}(0, 1)$.

$$c_{f_{l-1}} = \mathrm{Var}_{\mathbf{x}_{l-1}^j\sim\mathcal{N}(0,1)}\left[f_{l-1}(\mathbf{x}_{l-1}^j)\right]$$

This can be done beforehand and the factors for common activation functions can be stored in a lookup table. Finally, plugging in the constraint for the second moment in the above equation, we obtain the following prior variance.

$$\left(\sigma_l^{i,j}\right)^2 = \mathrm{Var}_{\mathbf{w}\sim p(\mathbf{w})}\left[\mathbf{W}_l^{i,j}\right] = \frac{1}{N_{l-1} c_{f_{l-1}}} \tag{17}$$

I Posterior Initialisation

It is known that a proper initialisation of standard neural networks is crucial for the optimisation process Glorot & Yoshua Bengio (2010); He et al. (2015). In Bayesian neural networks, the matter becomes even more complicated, since we have to deal additionally with the variance of the Monte Carlo estimator due to re-parametrisation. Analogous to the choice of prior parameters, we seek a posterior initialisation that yields the first two moments of zero and one. A naive attempt would be to initialise the posterior with the prior parameters. However, the significant noise in the Monte Carlo estimation typically leads to bad optimisation results and even numerical instabilities. We propose an initialisation method which fulfills our constraints but allows us determine the variance of the initial posterior with two hyperparameters α and β.

Let us denote the mean and log-scale parameters of the approximate posterior as $\theta = \{\theta_\mu, \theta_{\log\sigma}\}$. We choose the following initialisation distributions.

$$q(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \theta_\mu,\ e^{2\theta_{\log\sigma}}),$$

where

$$p(\theta_\mu) = \mathcal{N}\left(\theta_\mu; \mu_{\theta_\mu}, \sigma^2_{\theta_\mu}\right),$$

and

$$p(\theta_{\log\sigma}) = \mathcal{N}\big(\theta_{\log\sigma}; \mu_{\theta_{\log\sigma}}, \sigma^2_{\theta_{\log\sigma}}\big).$$

Here and in the following, we dropped the time index for the approximate posterior, as well as the indices l, i, and j for the model parameters θ.

We follow a similar derivation as in Sec. H. As for the prior, the mean of the initialisation distribution will be zero for all parameters.

$$\mu_{\theta_\mu} = 0.$$

For the second moment, the derivation is as follows.

$$\begin{aligned}
&\mathbb{E}_{\theta\sim p(\theta), \mathbf{w}\sim q(\mathbf{w}|\theta), \mathbf{x}_0\sim p(\mathbb{D})}\left[(\mathbf{x}_l^i)^2\right] = \mathbb{E}_{\theta\sim p(\theta)}\left[\mathbb{E}_{\mathbf{w}\sim q(\mathbf{w}|\theta)}\left[\mathbb{E}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[\left(\mathbf{x}_l^i\right)^2\right]\right]\right] \\
&= \mathbb{E}_{\theta\sim p(\theta)}\left[\mathbb{E}_{\mathbf{w}\sim q(\mathbf{w}|\theta)}\left[\mathrm{Var}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[\sum_j^{N_{l-1}} \mathbf{W}_l^{i,j}\cdot f_{l-1}(\mathbf{x}_{l-1}^j)\right] + 0\right]\right] \\
&= \mathbb{E}_{\theta\sim p(\theta)}\left[\mathbb{E}_{\mathbf{w}\sim q(\mathbf{w}|\theta)}\left[\sum_j^{N_{l-1}} (\mathbf{W}_l^{i,j})^2\right]\mathrm{Var}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[\mathbf{x}_{l-1}^j\right]\right] \\
&= \mathbb{E}_{\theta\sim p(\theta)}\left[\mathbb{E}_{\mathbf{w}\sim q(\mathbf{w}|\theta)}\left[\sum_j^{N_{l-1}} (\mathbf{W}_l^{i,j})^2\right]\right]\cdot \mathbb{E}_{\theta\sim p(\theta)}\left[\mathbb{E}_{\mathbf{w}\sim q(\mathbf{w}|\theta)}\left[\mathrm{Var}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[f_{l-1}(\mathbf{x}_{l-1}^j)\right]\right]\right] \\
&= \sum_j^{N_{l-1}} \mathbb{E}_{\theta\sim p(\theta)}\left[\mathbb{E}_{\mathbf{w}\sim q(\mathbf{w}|\theta)}\left[(\mathbf{W}_l^{i,j})^2\right]\right]\cdot \mathbb{E}_{\theta\sim p(\theta)}\left[\mathbb{E}_{\mathbf{w}\sim q(\mathbf{w}|\theta)}\left[\mathrm{Var}_{\mathbf{x}_0\sim p(\mathbb{D})}\left[f_{l-1}(\mathbf{x}_{l-1}^j)\right]\right]\right] \\
&=: \sum_j^{N_{l-1}} \mathbb{E}_{\theta\sim p(\theta)}\left[\theta_\mu^2 + e^{2\cdot\theta_{\log\sigma}}\right]\cdot c_{f_{l-1}} \\
&= N_{l-1}\cdot c_{f_{l-1}}\cdot \mathbb{E}_{\theta\sim p(\theta)}\left[\theta_\mu^2 + e^{2\cdot\theta_{\log\sigma}}\right] \\
&= N_{l-1}\cdot c_{f_{l-1}}\cdot \left(\mathbb{E}_{\theta\sim p(\theta)}\left[\theta_\mu^2\right] + \mathbb{E}_{\theta\sim p(\theta)}\left[e^{2\cdot\theta_{\log\sigma}}\right]\right) \\
&= N_{l-1}\cdot c_{f_{l-1}}\cdot \left(\mathbb{E}_{\theta\sim p(\theta)}\left[\theta_\mu\right]^2 + \mathrm{Var}_{\theta\sim p(\theta)}\left[\theta_\mu\right] + e^{2\mathbb{E}[\theta_{\log\sigma}]+2\mathrm{Var}[\theta_{\log\sigma}]}\right) \\
&= N_{l-1}\cdot c_{f_{l-1}}\cdot \left(\mu_{\theta_\mu}^2 + \sigma_{\theta_\mu}^2 + e^{2\mu_{\theta_{\log\sigma}}+2\sigma^2_{\theta_{\log\sigma}}}\right) \\
&= N_{l-1}\cdot c_{f_{l-1}}\cdot \left(\sigma_{\theta_\mu}^2 + e^{2\mu_{\theta_{\log\sigma}}+2\sigma^2_{\theta_{\log\sigma}}}\right).
\end{aligned}$$

Hence, the second constraint is as follows.

$$\frac{1}{N_{l-1}\cdot c_{f_{l-1}}} = \sigma_{\theta_\mu}^2 + e^{2\mu_{\theta_{\log\sigma}}+2\sigma^2_{\theta_{\log\sigma}}}.$$

In contrast to Sec. H, we are now under-constrained by 2 parameters. We therefore introduce two hyperparameters α and β. We first determine $\alpha := \sigma_{\theta_{\log\sigma}}$, for which we generally choose small values $\alpha \approx 0$ ($\alpha = 0$ corresponds to initialising all posterior variances in the given layer with the same value). The second hyperparameter $\beta \in [0, 1]$ determines how much of the total variance is due to the variance of the location parameter and how much variance is due to the variance of the log-scale parameter. Inserting α and introducing β we obtain the following equations.

$$\sigma_{\theta_\mu}^2 = \frac{\beta}{N_{l-1}\cdot c_{f_{l-1}}},$$

and

$$e^{2\mu_{\theta_{\log\sigma}}+2\alpha^2} = \frac{1-\beta}{N_{l-1}\cdot c_{f_{l-1}}}.$$

Solving the last equation for $\mu_{\theta_{\log \sigma}}$, the result is as follows.

$$\mu_{\theta_{\log \sigma}} = \frac{1}{2} \log \frac{(1-\beta) \cdot e^{-2\alpha^2}}{N_{l-1} \cdot c_{f_{l-1}}}$$

We choose $\alpha = 0.001$ and $\beta = 0.999$ in all experiments.

A note on the relation to initialisation methods for deterministic neural networks. Our result is similar to the initialisation methods from Glorot & Yoshua Bengio (2010) and He et al. (2015). The difference is in the correction factor $c_{f_{l-1}}$. Whereas Glorot & Yoshua Bengio (2010) considers linear functions (or tanh in the linear regime), both methods base their derivation on the assumption that every data sample $\mathbf{x}_0$ is processed by a different, random neural network with independent weights, drawn from the initialisation distribution. The assumption is made explicit in (He et al., 2015) by the use of the variance of products of independent variables rule. We note that this assumption is false for both the initialisation of deterministic neural networks, as well as the graphical model assumption in Bayesian neural networks. Consequently, (He et al., 2015) obtains different correction factors (in their case for relu and leaky relu), taking into account the mean after the forward-pass through the non-linearity.

7 On the detrimental effect of invariances in the likelihood for variational inference

During the research conducted for Ch. 6, it became evident that the main bottleneck limiting the Bayesian continual-learning approach is the poor approximation quality of variational BNNs on small dataset sizes, even though Bayesian methods should perform well especially in the low-data regime. This is problematic, because it is natural that only few data points are observed per step in online/continual learning. This chapter explains a key aspect of this problem: invariances in the likelihood function—which occur in over-parametrised models—have a detrimental effect on variational-Bayesian inference. For instance, the function computed by NNs is invariant w.r.t. *permutations* of hidden nodes/neurons in each layer, corresponding to the simultaneous permutation of weight matrices from preceding and subsequent layers. Furthermore, the parameter space of NNs has subspaces that exhibit *translation* invariance. This theoretical work shows that invariances lead to an additional and quantifiable (though intractable) gap in the ELBO objective, compared to a hypothetical approximation that accounts for the known invariances. The detrimental effect of *translation* invariance on the mean-field Gaussian approximation is shown for over-parametrised Bayesian linear regression models: as the over-parametrisation increases, the optimal (in terms of the ELBO objective) mean-field posterior approximation collapses to the prior. However, if the objective is corrected by the invariance gap, the optimal solution is indeed the true posterior. While this work does not propose a solution to this long-standing problem, it explains *why* variational Bayes tends to collapse, and provides a framework for future work to develop approximations that correct for the detrimental invariances.

Authors Richard Kurle
Ralf Herbrich
Tim Januschowski
Yuyang Wang
Jan Gasthaus

Conference Advances in Neural Information Processing Systems, NeurIPS 2022

Contribution

Problem definition	significantly contributed
Literature survey	significantly contributed
Algorithm development	contributed
Method implementation	significantly contributed
Experimental evaluation	significantly contributed
Preparation of the manuscript	significantly contributed

On the detrimental effect of invariances in the likelihood for variational inference

Richard Kurle *
AWS AI Labs
kurler@amazon.com

Ralf Herbrich
Hasso-Plattner Institut
ralf.herbrich@hpi.de

Tim Januschowski †
Zalando SE
tim.januschowski@zalando.de

Yuyang Wang
AWS AI Labs
yuyawang@amazon.com

Jan Gasthaus
AWS AI Labs
gasthaus@amazon.com

Abstract

Variational Bayesian posterior inference often requires simplifying approximations such as mean-field parametrisation to ensure tractability. However, prior work has associated the variational mean-field approximation for Bayesian neural networks with underfitting in the case of small datasets or large model sizes. In this work, we show that invariances in the likelihood function of over-parametrised models contribute to this phenomenon because these invariances complicate the structure of the posterior by introducing discrete and/or continuous modes which cannot be well approximated by Gaussian mean-field distributions. In particular, we show that the mean-field approximation has an additional gap in the evidence lower bound compared to a purpose-built posterior that takes into account the known invariances. Importantly, this invariance gap is not constant; it vanishes as the approximation reverts to the prior. We proceed by first considering translation invariances in a linear model with a single data point in detail. We show that, while the true posterior can be constructed from a mean-field parametrisation, this is achieved only if the objective function takes into account the invariance gap. Then, we transfer our analysis of the linear model to neural networks. Our analysis provides a framework for future work to explore solutions to the invariance problem.

1 Introduction

Bayesian neural networks (BNNs) have several appealing advantages compared to deterministic neural networks (NN) such as improving generalization [36], capturing epistemic uncertainty [16], and providing a framework for continual learning methods [17, 22]. Unfortunately, reaping these theoretical benefits has so far been impeded [35]. In particular, several practical issues with variational Bayesian inference methods—which are the de-facto standard technique for scaling inference in BNNs to large datasets [14, 1]—have been identified to be partially responsible for a performance gap compared to deterministic NNs[15]. The most common variational approximation of the posterior is a product of independent Normal distributions, commonly referred to as the *mean-field* approximation. It has been observed that mean-field variational BNNs (VBNNs) suffer from severe underfitting for large models and small dataset sizes [12]. Recent work [4] showed that under certain assumptions such as an odd Lipschitz activation function and a finite dataset with bounded likelihood, the optimal mean-field approximation *collapses* to the prior as the NN width increases, *ignoring the data*.

*Correspondence to kurler@amazon.com.
†Work done while at AWS AI Labs.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

The goal of this work is to shed light on *why* the mean-field variational approximation collapses and to provide a new angle for future works to address this shortcoming. To this end, we study invariances in the likelihood function of *overparametrised models* and their detrimental effect on variational inference. For instance, it is well known that NNs exhibit several invariances *with respect to their parameters*, including node permutation invariance [18], sign-flip invariance (in the case of odd activation functions) [27, 2], scaling invariance (in the case of piece-wise linear activations) [24], and as we will show in Sec. 5, the parameter space of BNNs additionally has subspaces with *translation invariance*.

Invariance in the likelihood function does not necessarily pose a challenge if maximum likelihood point estimation through stochastic gradient descent is used, because convergence to any of the equivalent optima (resulting from the invariance) suffices for prediction. However, these invariances are detrimental to the variational Bayesian approach. As a first step to show this, we isolate the impact of the invariances in Sec. 3. To this end, we construct both a *mean-field* as well as an *invariance-abiding* approximation which explicitly models the invariances by integrating over all transformations that leave the likelihood invariant. Notably, both aforementioned posterior approximations are constructed from the same (mean-field) *likelihood* approximation. We then prove that, under the conditions outlined in Sec. 3, the mean-field approximation induces the same posterior predictive distribution as the invariance-abiding approximation. However, we also prove that the ELBO objective corresponding to these two approximations differs by the KL divergence between both posterior approximations; we refer to this difference as *invariance gap*. Importantly, the gap vanishes if both distributions are identical, which is the case if the mean-field approximation reverts to the prior.

We then demonstrate the detrimental effect of invariances in the likelihood function of an overparametrised Bayesian linear regression model in Sec. 4. This model is purposely selected as the canonical model exhibiting (only) *translation invariance*. We provide a detailed analysis of this model, including a tractable solution for the invariance-abiding approximation. It turns out that the optimal parameters w.r.t. the ELBO objective with the invariance-abiding distribution result in the true posterior. In contrast, posterior approximations with parameters that are optimal w.r.t. the mean-field ELBO revert to the prior as the number of dimensions increases.

Finally, we transfer our analysis of the linear model to VBNNs in Sec. 5, showing that subspaces in BNNs exist that exhibit translation invariance. Combined with a previous analysis of the node permutation invariance (cf. Kurle et al. [18] or App. E of the supplementary material), this provides the basis for future work to approximate the (generally intractable) invariance gap and thereby optimise for a tighter and favorable ELBO objective. We start in Sec. 2 by introducing the basic concepts of VBNNs and recent results on which we build our contribution.

2 Background

2.1 Variational Bayesian Neural Networks

Neural network functional model. Deep NNs are layered models that progressively transform their inputs in each layer. More formally, an L–layer NN computes algebraically

$$f(\mathbf{x}) = h_L\left(\mathbf{w}_{L,1}^{\mathrm{T}}\mathbf{z}_{L-1}\right)\,, \quad z_{1,i} = h_1\left(\mathbf{w}_{1,i}^{\mathrm{T}}\mathbf{x}\right)\,, \quad z_{l,i} = h_l\left(\mathbf{w}_{l,i}^{\mathrm{T}}\mathbf{z}_{l-1}\right)\,,$$

where the $h_l : \mathbb{R} \to \mathbb{R}$ are monotonic transfer functions that introduce non-linearities. Such a network has $n_1 + n_2 + \cdots + n_L$ many nodes $z_{l,i}$, where i indexes the layer-wise vectors $\mathbf{z}_l$. These nodes are each a weighted linear combination of either the input vector $\mathbf{x}$ (first hidden layer) or the value of the hidden units $\mathbf{z}_l$ (all other hidden layers and the output $f(\mathbf{x})$). We denote all learnable parameters of the model by the stacked weight vectors of each layer and node, $\mathbf{w} = [\mathbf{w}_{1,1}^{\mathrm{T}}, \ldots, \mathbf{w}_{L,n_L}^{\mathrm{T}}]^{\mathrm{T}}$.

Variational Bayesian treatment. If the weights and biases $\mathbf{w}$ are treated as random variables with a prior distribution $p(\mathbf{w})$, then the posterior $p(\mathbf{w} \,|\, \mathcal{D})$—induced by the dataset $\mathcal{D}$ through the likelihood function $\ell(\mathbf{w}; \mathcal{D}) := p(\mathcal{D} \,|\, \mathbf{w})$ that is defined via $f(\mathbf{x})$—is referred to as a Bayesian neural network (BNN). The variational Bayesian method approximates the posterior by a distribution $q_{\boldsymbol{\theta}}(\mathbf{w}) \approx p(\mathbf{w} \,|\, \mathcal{D})$ with variational parameters $\boldsymbol{\theta}$, casting inference as an optimization problem

$$q^*_{\boldsymbol{\theta}}(\mathbf{w}) = \underset{q_{\boldsymbol{\theta}} \,\in\, \mathcal{Q}}{\operatorname{argmin}} \; \mathrm{KL}\left[q_{\boldsymbol{\theta}} \,||\, p(\cdot \,|\, \mathcal{D})\right], \tag{1}$$

where $\mathrm{KL}\left[q_{\boldsymbol{\theta}} \,||\, p(\cdot \,|\, \mathcal{D})\right] := \mathbb{E}_{\mathbf{w}\sim q_{\boldsymbol{\theta}}}\left[\ln q_{\boldsymbol{\theta}}(\mathbf{w}) - \ln p(\mathbf{w}|\mathcal{D})\right]$ and $\mathcal{Q}$ is a family of distributions over $\mathbf{w}$. The optimization of (1) is achieved by maximising a lower bound to the (log) model evidence $\ln p(\mathcal{D})$,

$$\begin{aligned}\mathcal{L}_{\mathrm{ELBO}}\left(q_{\boldsymbol{\theta}}, \mathcal{D}\right) = \ln p(\mathcal{D}) - \mathrm{KL}\left[q_{\boldsymbol{\theta}} \,||\, p(\cdot \,|\, \mathcal{D})\right] &= \mathbb{E}_{\mathbf{w}\sim q_{\boldsymbol{\theta}}}\left[\ln \frac{\ell(\mathbf{w};\mathcal{D})\, p(\mathbf{w})}{q_{\boldsymbol{\theta}}(\mathbf{w})}\right] \\ &= \mathrm{ELL}\left(q_{\boldsymbol{\theta}},\, \mathcal{D}\right) - \mathrm{KL}\left[q_{\boldsymbol{\theta}} \,||\, p\right],\end{aligned} \tag{2}$$

where $\mathrm{ELL}\left(q_{\boldsymbol{\theta}},\, \mathcal{D}\right) := \mathbb{E}_{\mathbf{w}\sim q_{\boldsymbol{\theta}}}\left[\ln \ell(\mathbf{w};\mathcal{D})\right]$ is the expected log-likelihood.

Definition 1 (Mean-field variational BNN)**.** A *mean-field* variational BNN (VBNN) is the minimizer of (1) where $\mathcal{Q}$ is the family of Gaussians with diagonal covariance matrix.

2.2 Data-related bound on the KL divergence

The term $\mathrm{KL}\left[q_{\boldsymbol{\theta}} \,||\, p\right]$ in (2) admits a data-dependent upper bound that naturally occurs as a consequence of the finite information provided by a finite dataset in the presence of noise. This result has been shown in [4] and we recall the result for a Gaussian likelihood with homogeneous noise (for other likelihood functions, we refer to App. F in [4]).

Assume a VBNN with Gaussian observation noise defined through $y = f_{\mathbf{w}}(\mathbf{x}) + \epsilon$, with $\epsilon \sim \mathcal{N}(0, \sigma_y^2)$, an isotropic Gaussian prior $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \mathbf{I})$, and a mean-field variational approximation $q_{\boldsymbol{\theta}}(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{m}, \mathrm{Diag}(\mathbf{v}))$. Define the NN output variance $\sigma_L^2\left(\mathbf{x}^{(n)}\right) := \mathbb{V}_{\mathbf{w}\sim p}\left[f(\mathbf{x}^{(n)};\, \mathbf{w})\right]$ for some fixed input $\mathbf{x}^{(n)}$, where L indicates the last layer. Then,

$$\mathrm{KL}\left[q_{\boldsymbol{\theta}} \,||\, p\right] \leq \mathrm{ELL}\left(q^*, \mathcal{D}\right) - \mathrm{ELL}\left(p, \mathcal{D}\right) = \sum_{n=1}^{N} \frac{\sigma_L^2\left(\mathbf{x}^{(n)}\right) + \left(y^{(n)}\right)^2}{2\sigma_y^2}, \tag{3}$$

where q^* is a hypothetical approximation that predicts the data perfectly up to the noise variance σ_y^2 (see App. F for details). Prior work [4] has used the data-related bound to prove *that* the predictive distribution of mean-field BNNs converges to the prior predictive distribution if the network width is large and the activation function is odd, ultimately resulting in posterior collapse to the prior. In contrast, we address the question *why* the mean-field approximation cannot approximate the posterior by relating this data-related bound to invariances in neural network functions in the following section.

3 Invariance-abiding variational approximation

We develop a framework that enables modelling the invariances in the likelihood and understanding their impact on the ELBO objective. To achieve this, we approximate the likelihood by a Gaussian function with variational parameters and marginalise over all transformations to which the true likelihood is invariant. By taking the product with the prior, we construct an *invariance-abiding* posterior approximation q_{mix} which can be related to a *mean-field* approximation q_0 that does not model invariances. We then describe conditions under which both approximations yield an identical posterior predictive, while the KL regularisation term of q_0 is lower bounded by the KL of q_{mix}.

Variational likelihood and posterior approximations. Non-identifiability implies that the posterior does not concentrate on a single set of parameters irrespective of the dataset size, because the same likelihood is assigned to different parameter values [11]. We model this invariance through transformations $t(\cdot, \mathbf{r})$ to which the likelihood $\ell(\mathbf{w};\mathcal{D}) = p(\mathcal{D} \,|\, \mathbf{w})$ is invariant via variables $\mathbf{r}$:

$$\forall \mathbf{r} \sim p(\mathbf{r}): \quad \ell(t(\mathbf{w}, \mathbf{r}); \mathcal{D})) = \ell(\mathbf{w}; \mathcal{D}). \tag{4}$$

From (4) it follows that the likelihood is also invariant w.r.t. the marginalisation

$$\ell\left(\mathbf{w}; \mathcal{D}\right) = \mathbb{E}_{\mathbf{r}\sim p}\left[\ell\left(t\left(\mathbf{w}, \mathbf{r}\right); \mathcal{D}\right)\right].$$

We assume that each of the equivalent parametrisations $\mathbf{w}' = t(\mathbf{w}, \mathbf{r})$ of the likelihood has the same probability a priori. For discontinuous transformations such as node permutations (see App. E), $p(\mathbf{r})$ is a uniform distribution over discrete variables indexing these transformations. For the continuous translation invariance (see Sec. 4), we model the uniform distribution as a Gaussian with infinite variance. It is conceivable that the likelihood can be constructed by marginalising over these transformations of a simpler function $\ell_0(\mathbf{w};\mathcal{D})$ such that $\ell(\mathbf{w};\mathcal{D}) = \mathbb{E}_{\mathbf{r}\sim p}\left[\ell_0(t(\mathbf{w}, \mathbf{r}); \mathcal{D})\right]$. For

instance, permutation invariance induces a factorial number of discontinuous modes that are each equivalent (cf. [18], App. E); and as we show in Sec. 4, the full covariance Gaussian likelihood and posterior of an over-parametrised Bayesian linear regression model can be constructed from a product of independent Gaussians. Curiously, it may be sufficient to approximate ℓ_0 while taking into account the known invariances. To this end, we define a Gaussian *variational likelihood approximation* $g_0(\mathbf{w};\boldsymbol{\theta}) \approx \ell_0(\mathbf{w},\mathcal{D})$, with variational parameters $\boldsymbol{\theta}$. From this single mode approximation, we model an *invariance-abiding likelihood* approximation using the same parametrisation through

$$g_{\text{mix}}(\mathbf{w};\boldsymbol{\theta}) := \mathbb{E}_{\mathbf{r}\sim p}\left[g_0(t(\mathbf{w},\mathbf{r});\boldsymbol{\theta})\right] \approx \mathbb{E}_{\mathbf{r}\sim p}\left[\ell_0(t(\mathbf{w},\mathbf{r}),\mathcal{D})\right] = \ell(\mathbf{w};\mathcal{D}). \tag{5}$$

We consider mean-field Gaussians for the prior $p(\mathbf{w})$ and likelihood approximation $g_0(\mathbf{w};\boldsymbol{\theta})$, where $\boldsymbol{\theta} = \{\mathbf{m},\boldsymbol{\lambda}\}$ are means and variances of g_0. We then define a *mean-field posterior* as the product

$$q_0(\mathbf{w};\boldsymbol{\theta}) := Z_0^{-1}\, p(\mathbf{w}) \cdot g_0(\mathbf{w};\boldsymbol{\theta}), \qquad Z_0 = \int p(\mathbf{w}) \cdot g_0(\mathbf{w};\boldsymbol{\theta}) d\mathbf{w}. \tag{6a}$$

Similarly, we define an *invariance-abiding posterior* as the product of prior and invariant likelihood:

$$q_{\text{mix}}(\mathbf{w};\boldsymbol{\theta}) := Z_{\text{mix}}^{-1}\, p(\mathbf{w}) \cdot g_{\text{mix}}(\mathbf{w};\boldsymbol{\theta}), \qquad Z_{\text{mix}} = \int p(\mathbf{w}) \cdot g_{\text{mix}}(\mathbf{w};\boldsymbol{\theta}) d\mathbf{w}, \tag{6b}$$

where Z_0 and Z_{mix} are normalisation constants. While $q_0(\mathbf{w};\boldsymbol{\theta})$ is a mean-field approximation, the invariance-abiding approximation $q_{\text{mix}}(\mathbf{w};\boldsymbol{\theta})$ is a mixture with infinite continuous or finite discrete modes depending on the type of invariance. We will describe continuous translation invariance in Sec. 4; for the discrete node permutation invariance, see App. E of the supplementary material.

Posterior predictive equivalence. Next, we discuss conditions under which we can construct approximations $q_{\text{mix}}(\mathbf{w};\boldsymbol{\theta})$ and $q_0(\mathbf{w};\boldsymbol{\theta})$ that yield the same predictive distribution. We first write the density q_{mix} as an expectation of product densities,

$$q_{\text{mix}}(\mathbf{w};\boldsymbol{\theta}) = Z_{\text{mix}}^{-1}\, p(\mathbf{w}) \cdot \int p(\mathbf{r})\, g_0(t(\mathbf{w},\mathbf{r});\boldsymbol{\theta}) d\mathbf{r} = \int p(\mathbf{r})\, \frac{p(\mathbf{w}) \cdot g_0(t(\mathbf{w},\mathbf{r});\boldsymbol{\theta})}{Z_{\text{mix}}} d\mathbf{r}.$$

Then, we assume that there exists a mapping $\mathbf{r}' = \varphi(\mathbf{r})$ such that

$$\forall \mathbf{r} \sim p(\mathbf{r}) : p(\mathbf{w}) \cdot g_0(t(\mathbf{w},\mathbf{r});\boldsymbol{\theta}) = p(t(\mathbf{w},\varphi(\mathbf{r}))) \cdot g_0(t(\mathbf{w},\varphi(\mathbf{r}));\boldsymbol{\theta}). \tag{7}$$

The condition in (7) allows us to exploit the invariance property of the likelihood (approximation) also for each product density $q_0(t(\mathbf{w},\varphi(\mathbf{r}));\boldsymbol{\theta}) = p(t(\mathbf{w},\varphi(\mathbf{r}))) \cdot g_0(t(\mathbf{w},\varphi(\mathbf{r}));\boldsymbol{\theta})$, because then

$$q_{\text{mix}}(\mathbf{w};\boldsymbol{\theta}) = \int \frac{Z_0(\mathbf{r})}{Z_{\text{mix}}} p(\mathbf{r}) \cdot q_0(t(\mathbf{w},\varphi(\mathbf{r}));\boldsymbol{\theta}) d\mathbf{r},$$

where the normalisation constants are $Z_0(\mathbf{r}) = \int p(\mathbf{w}) \cdot g_0(t(\mathbf{w},\varphi(\mathbf{r}));\boldsymbol{\theta}) d\mathbf{w}$, and Z_{mix} is defined in (6b). We show that the condition in (7) holds for translation and permutation invariance in App. C.

The second condition is that all invariance transformations $t(\cdot,\mathbf{r})$ must be volume-preserving, i.e.

$$\forall \mathbf{r} \sim p(\mathbf{r}) : \left|\det \frac{\partial t(\mathbf{w},\mathbf{r})}{\partial \mathbf{w}}\right|^{-1} = 1. \tag{8}$$

Although (7) and (8) are fairly restricting, common invariances such as translation and permutation invariance fulfil these conditions for Gaussian priors and likelihood approximations (see App. C). With (7) and (8), we can show the posterior predictive equivalence (see Lemma 1 in App. C)

$$\mathbb{E}_{\mathbf{w}\sim q_{\text{mix}}}\left[\ln p(\mathcal{D}\,|\,\mathbf{w})\right] = \mathbb{E}_{\mathbf{w}\sim q_0}\left[\ln p(\mathcal{D}\,|\,\mathbf{w})\right]. \tag{9}$$

Invariance gap. If the conditions in (7) and (8) are met, we also show (see Lemma 2 in App. C)

$$\mathcal{L}_{\text{ELBO}}\left(q_0,\mathcal{D}\right) - \mathcal{L}_{\text{ELBO}}\left(q_{\text{mix}},\mathcal{D}\right) = \text{KL}\left[q_0\,||\,p\right] - \text{KL}\left[q_{\text{mix}}\,||\,p\right] = \text{KL}\left[q_0\,||\,q_{\text{mix}}\right]. \tag{10}$$

Interestingly, the gap between the two respective ELBO objectives is given exactly by the relative entropy between the mean-field and invariance-abiding approximation; we therefore refer to it as the *invariance gap*. We make the following observation about the detrimental effect of the invariances: when maximising the standard ELBO $\mathcal{L}_{\text{ELBO}}\left(q_0,\mathcal{D}\right)$ instead of the tighter objective $\mathcal{L}_{\text{ELBO}}\left(q_{\text{mix}},\mathcal{D}\right)$ w.r.t. the parameters of the variational *likelihood* approximation $g_0(\mathbf{w};\boldsymbol{\theta})$ (used to construct both q_0 and q_{mix}), we see that the former objective favors solutions where q_0 and q_{mix} coincide. This suboptimal solution is obtained if the mean-field posterior $q_0(\mathbf{w})$ and thus also the invariance-abiding posterior $q_{\text{mix}}(\mathbf{w})$ revert to the prior. This is the case if $g_0(\mathbf{w})$ is uniform, because then $g_{\text{mix}}(\mathbf{w})$ is uniform as well, and because both posteriors are constructed as the product of prior and likelihood approximation (cf. (6a)).

Implication of data-related bound. Since $\mathrm{KL}\left[q_0 \,||\, p\right]$ is upper bounded by the best- and worst-case ELL as described in Sec. 2.2, the invariance gap is also bounded (using (10) and (3)):

$$\mathrm{KL}\left[q_0 \,||\, q_{\mathrm{mix}}\right] \leq \mathrm{KL}\left[q_0 \,||\, p\right] \leq \mathrm{ELL}\left(q^*, \mathcal{D}\right) - \mathrm{ELL}\left(p, \mathcal{D}\right). \tag{11}$$

Consequently, this bound puts a constraint on the achievable solutions to the maximisation of the ELBO objective w.r.t. the variational parameters of the mean-field approximation $q_0(\mathbf{w}; \boldsymbol{\theta})$. As discussed above, one specific parametrisation for which the invariance gap vanishes is the mean-field posterior collapse, $q_0(\mathbf{w}; \boldsymbol{\theta}) \approx p(\mathbf{w})$. However, to show that the invariance gap not only admits posterior collapse as a potential parametrisation but indeed incentivises it, we next tackle the question how the invariance gap behaves for non-collapsed approximations. One particularly relevant variational parametrisation is the optimal solution w.r.t. the ELBO objective with the invariance-abiding approximation. In order to tackle this question, we now consider a simple model for which the relevant distributions and the invariance gap can be computed exactly.

4 Translation invariance in linear models

We study an over-parametrised Bayesian linear regression model as the canonical model that exhibits *translation invariance*. The model serves as a useful tool to understand the detrimental effect of translation invariance since all interesting quantities can be computed analytically, incl. the mean-field and invariance-abiding distribution defined in (6) and the invariance gap from (10). We show how to lift results from this canonical model to the more general case of NNs in Sec. 5.

Likelihood model. Consider a linear model with K latent variables $\mathbf{w} = [w_1, \ldots, w_K]^{\mathrm{T}}$ and assume that we have N observations of variables y given dependent inputs $\mathbf{x}$. To simplify the setting, we will assume that $\mathbf{x} = \mathbf{1}$ (see App. D for the more general case). We further assume that the observation depends only on the inner product with the inputs and additive Gaussian noise. That is, $y = \frac{1}{K}\mathbf{1}^{\mathrm{T}}\mathbf{w} + \epsilon, \quad \epsilon \sim \mathcal{N}\left(0, \sigma_y^2\right)$. Note that any change in $\mathbf{w}$ which leaves the sum of the elements unaffected does not change the likelihood. In general, we can model this translation invariance using a $K-1$ dimensional vector $\boldsymbol{\Delta} \in \mathbb{R}^{K-1}$ and observing that

$$\mathbf{1}^{\mathrm{T}}\mathbf{w} = \mathbf{1}^{\mathrm{T}}\left(\mathbf{w} + \mathbf{B}\boldsymbol{\Delta}\right), \quad \mathbf{B} := \begin{bmatrix} \mathbf{I} \\ -\mathbf{1}^{\mathrm{T}} \end{bmatrix}, \tag{12}$$

since $\mathbf{1}^{\mathrm{T}}\mathbf{B}\boldsymbol{\Delta} = \mathbf{0}$. This over-parametrised model exhibits translation invariance $t(\mathbf{w}, \Delta) = \mathbf{w} - \mathbf{B}\Delta$.

Prior. We assume a Gaussian prior $p\left(\mathbf{w}\right) = \mathcal{N}\left(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$ with diagonal covariance $\boldsymbol{\Sigma} = \mathrm{Diag}\left(\boldsymbol{\sigma}^2\right)$, where we consider $\boldsymbol{\sigma}^2 = K \cdot \sigma_0^2 \cdot \mathbf{1}$ proportional to the number of dimensions K, such that the predictive variance $\mathbb{V}\left[\frac{1}{K}\sum_k w_k + \epsilon\right]$ is constant w.r.t. K. Another way to view this is to model a prior over parameters that induces the same prior over functions for different K.

Posterior. The posterior of this Gaussian linear model can be computed as (see App. D.3)

$$p(\mathbf{w} \,|\, \mathbf{y}) = \mathcal{N}\left(\mathbf{w}; \mathbf{m}_p^*, \mathbf{V}_p^*\right), \quad \mathbf{V}_p^* = \left(\frac{N}{K^2\sigma_y^2}\mathbf{1}\mathbf{1}^{\mathrm{T}} + \boldsymbol{\Sigma}^{-1}\right)^{-1}, \quad \mathbf{m}_p^* = \mathbf{V}_p^*\frac{\sum_i y_i}{K\sigma_y^2}\mathbf{1}. \tag{13}$$

As can be seen, the resulting posterior has full covariance with a diagonal plus rank-1 matrix. We will now show that we can construct this posterior from the prior and a mean-field Gaussian approximation of the likelihood by marginalising over all translations $\Delta \sim p(\Delta)$ to which the likelihood is invariant.

4.1 Mean-field parametrisation of the likelihood function

Next, we construct the mean-field and invariance-abiding posterior approximations defined in (6) from the prior defined above and the mean-field likelihood approximation with locations $\mathbf{m}$ and variances $\boldsymbol{\lambda}$. We model that the likelihood is translation invariant by computing the marginal from (5) and considering $p(\Delta) = \mathcal{N}\left(\boldsymbol{\Delta}; \mathbf{0}, \beta^2\mathbf{I}\right)$ with $\beta \to \infty$ as the uniform distribution over all translations:

$$\begin{aligned} g_{\mathrm{mix}}\left(\mathbf{w}; \boldsymbol{\theta}\right) &= \int g_0\left(t(\mathbf{w}, \Delta); \boldsymbol{\theta}\right) \cdot p\left(\boldsymbol{\Delta}\right) \, d\boldsymbol{\Delta} \\ &= \lim_{\beta \to \infty} \int \mathcal{N}\left(\mathbf{w}; \mathbf{m} + \mathbf{B}\boldsymbol{\Delta}, \mathrm{Diag}\left(\boldsymbol{\lambda}\right)\right) \cdot \mathcal{N}\left(\boldsymbol{\Delta}; \mathbf{0}, \beta^2\mathbf{I}\right) \, d\boldsymbol{\Delta} \;=:\; \mathcal{N}\left(\mathbf{w}; \mathbf{m}_{\mathrm{mix}}, \mathbf{V}_{\mathrm{mix}}\right). \end{aligned}$$

Writing the uniform distribution as the limiting case of a Gaussian with infinite variance allows us to compute the integral analytically. As shown in App. D, the resulting function is a multivariate Gaussian with a degenerate rank-1 covariance matrix and the same location parameter as $g_0(\mathbf{w})$:

$$\mathbf{m}_{\text{mix}} = \mathbf{m}, \quad \mathbf{V}_{\text{mix}}^{-1} = \frac{1}{\mathbf{1}^{\text{T}}\boldsymbol{\lambda}}\mathbf{1}\mathbf{1}^{\text{T}}. \tag{14}$$

However, the invariance-abiding posterior is non-degenerate, and it can be computed efficiently as

$$q_{\text{mix}}(\mathbf{w};\, \boldsymbol{\theta}) = \mathcal{N}\left(\mathbf{w};\, \boldsymbol{\mu} + \frac{\mathbf{1}^{\text{T}}(\mathbf{m}-\boldsymbol{\mu})}{\mathbf{1}^{\text{T}}(\boldsymbol{\lambda}+\boldsymbol{\sigma}^2)}\boldsymbol{\sigma}^2,\, \text{Diag}\left(\boldsymbol{\sigma}^2\right) - \frac{1}{\mathbf{1}^{\text{T}}(\boldsymbol{\lambda}+\boldsymbol{\sigma}^2)} \cdot \boldsymbol{\sigma}^2\left(\boldsymbol{\sigma}^2\right)^{\text{T}}\right). \tag{15}$$

As can be seen, this covariance matrix is not diagonal; it has an additive rank-1 term that vanishes as $\lambda \to \infty$. Interestingly, the location parameter of q_{mix} is translated from the *prior* location $\boldsymbol{\mu}$ along the direction of the *prior* variance vector $\boldsymbol{\sigma}^2$. This is because g_{mix} is uniform along the $K-1$ dimensions hyper-plane determined by its normal vector $\mathbf{1}$. Taking the Gaussian product then translates the location in the direction $\text{Diag}(\boldsymbol{\sigma}^2)\mathbf{1} = \boldsymbol{\sigma}^2$. Note also that the invariance-abiding posterior can be written in the same form as the true posterior in (13) (see App. D.3). The optimal invariance-abiding posterior is thus the true posterior. We visualise the two respective posterior approximations and the corresponding likelihood in Fig. 3 of App. D with two different parametrisations (cf. Sec. 4.3).

4.2 Invariance gap

To quantify the detrimental effect of the translation invariance in the considered linear model we now compute the invariance gap in (11). Note again that we assume a scaled standard normal prior with variance $\boldsymbol{\sigma}^2 = K\sigma_0^2 \cdot \mathbf{1}$. We simplify the form of the KL divergence by assuming that all variances and means take the same value in the likelihood (and thus also in the posterior) approximation, i.e.

$$\forall k : \lambda_k = \hat{\lambda},\, m_k = \hat{m},\, \sigma_k = \hat{\sigma},\, \mu_k = \hat{\mu}.$$

This choice is motivated by the fact that the posterior approximation is a function of the sum $\mathbf{1}^{\text{T}}\boldsymbol{\lambda}$ only (cf. (15)), and, hence, it makes no difference for q_{mix}. However, since the prior variance is proportional to $\mathbf{1}$, the Gaussian likelihood resulting in the highest ELBO for the approximation q_0 also has a variance vector proportional to $\mathbf{1}$. Using this assumption, the invariance gap is

$$\text{KL}\left[q_0 \,||\, q_{\text{mix}}\right] = \frac{K-1}{2}\left[\ln\left(\frac{\hat{\sigma}^2+\hat{\lambda}}{\hat{\lambda}}\right) + \frac{\hat{\lambda}}{\hat{\sigma}^2+\hat{\lambda}} - 1\right]. \tag{16}$$

This is a convex function in $\phi = \frac{\hat{\sigma}^2+\hat{\lambda}}{\hat{\lambda}}$ with a minimum at $\phi = 1$; it is minimised as $\hat{\lambda} \to \infty$. It is however not evident whether the gap has a large magnitude compared to the rest of the regularisation term and over-regularises in practice. In the next section, we therefore analyse this term at the optima for the mean-field and the invariance-abiding approximations defined in (17). The corresponding invariance gaps are visualised in Fig. 1 where it can be seen that the invariance gap grows linearly when using the optimal parameters of the invariance-abiding distribution and the unattainable data-related bound is reached quickly, thus preventing this optimum if (17b) is optimised instead of (17a).

4.3 Optimal mean-field and invariance-abiding parametrisations

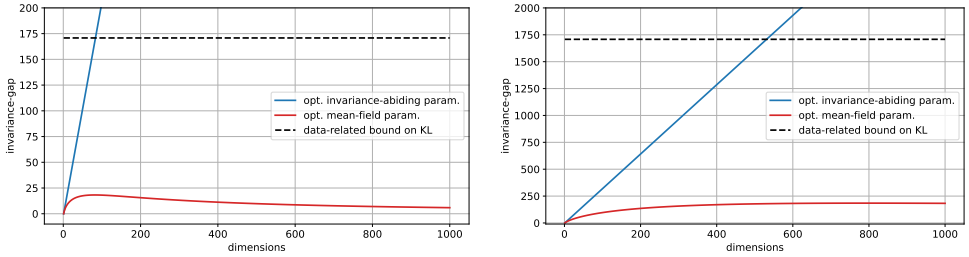
To better understand the detrimental effect of the invariance gap, we now analyse the ELL and KL terms of the ELBO objective for the mean-field and invariance-abiding variational approximations, respectively. We compare these terms for both distributions with the parameters optimized for both posterior distributions, giving 2×2 combinations. We denote the optimal parameters w.r.t. the invariance-abiding approximation and the mean-field approximation, respectively, as

$$\boldsymbol{\theta}_{\text{mix}}^* = \underset{\boldsymbol{\theta}}{\text{argmax}}\; \mathcal{L}_{\text{ELBO}}\left(q_{\text{mix}}(\cdot\,;\boldsymbol{\theta}), \mathcal{D}\right), \tag{17a}$$

$$\boldsymbol{\theta}_0^* = \underset{\boldsymbol{\theta}}{\text{argmax}}\; \mathcal{L}_{\text{ELBO}}\left(q_0(\cdot\,;\boldsymbol{\theta}), \mathcal{D}\right). \tag{17b}$$

Perhaps the simplest way compute these optimal parameters is to notice that q_{mix} in (15) can be written in the same form as the true posterior and then simply read out the optimal parameters. This is shown in App. D.3; the resulting optimal variational parameters are

$$g_0(\mathbf{w}\,;\boldsymbol{\theta}_{\text{mix}}^*) = \mathcal{N}\left(\mathbf{w}; \mathbf{m}_{\text{mix}}^*, \text{Diag}\left(\boldsymbol{\lambda}_{\text{mix}}^*\right)\right), \quad \mathbf{m}_{\text{mix}}^* = \frac{1}{N}\sum_{n=1}^{N} y^{(n)} \cdot \mathbf{1}, \quad \boldsymbol{\lambda}_{\text{mix}}^* = \frac{K\sigma_y^2}{N} \cdot \mathbf{1}. \tag{18}$$



(a) $N = 10$ observations. (b) $N = 100$ observations.

Figure 1: Invariance gap evaluated at different optima (cf. (17)), i.e. $\mathrm{KL}\left[q_0(\cdot;\, \boldsymbol{\theta}^*_{\mathrm{mix}}) \,||\, q_{\mathrm{mix}}(\cdot;\, \boldsymbol{\theta}^*_{\mathrm{mix}})\right]$ and $\mathrm{KL}\left[q_0(\cdot;\, \boldsymbol{\theta}^*_0) \,||\, q_{\mathrm{mix}}(\cdot;\, \boldsymbol{\theta}^*_0)\right]$. Prior variances are $\sigma^2 = K \cdot \mathbf{1}$, where K are the dimensions; the noise variance is $\sigma_y^2 = (2\pi \mathrm{e})^{-1}$; all observations $y = 1$ and $\mathbf{x} = \mathbf{1}$ are identical. As K increases, the invariance gap vanishes in case of the optimal parameters $\boldsymbol{\theta}^*_0$. In contrast, the gap for $\boldsymbol{\theta}^*_{\mathrm{mix}}$, which induces the true posterior predictive, grows linearly. As the data-related bound (cf. Sec. 2.2) can not be exceeded by the optimal parameters, $\boldsymbol{\theta}^*_0$ can not coincide with the optimal $\boldsymbol{\theta}^*_{\mathrm{mix}}$.

The optimal parameters for the mean-field *posterior* can also be computed analytically, since the Gaussian mean-field distribution that minimises the KL divergence to the multivariate Gaussian true posterior is known [32]. The resulting optimal parameters of the mean-field *likelihood* are

$$g_0(\mathbf{w}\,;\, \boldsymbol{\theta}^*_0) = \mathcal{N}\left(\mathbf{w}; \mathbf{m}^*_0, \mathrm{Diag}\left(\boldsymbol{\lambda}^*_0\right)\right), \quad \mathbf{m}^*_0 = \mathbf{m}^*_{\mathrm{mix}}, \quad \boldsymbol{\lambda}^*_0 = \frac{K^2 \sigma_y^2}{N}\, \mathbf{1}. \tag{19}$$

As can be seen, the two respective optimal parameters differ by the factor K in the likelihood variance. We then construct the two respective posterior approximations q_0 and q_{mix} with these two optimal parameters and compare the 2×2 combinations in Fig. 2, where we visualise the ELBO terms.

Note again that we model prior variances $\boldsymbol{\sigma}^2 = K\sigma_0^2 \cdot \mathbf{1}$ such that prior and posterior over functions are identical for any K. Due to this choice and since $q_{\mathrm{mix}}(\mathbf{w}; \boldsymbol{\theta}^*_{\mathrm{mix}})$ approximates the true posterior exactly, it does not suffer from over-parametrisation. This can be seen by the loss terms in Fig. 2 being constant in K. In contrast, since $\boldsymbol{\lambda}^*_0$ depends quadratically on K and $\boldsymbol{\sigma}^2$ only linearly, $q_{\mathrm{mix}}(\mathbf{w}; \boldsymbol{\theta}^*_0)$ *collapses* to the prior as $K \to \infty$. This can be seen e.g. by the shrinking KL regularisation (Fig. 2a) and ELL (in 2b) term. As a consequence, and in line with Coker et al. [4], the posterior predictive variance of the optimal mean-field approximation reverts to the prior predictive variance (Fig. 2d).

We have shown that—in the simplified setting of a linear model with dependent observations—the consequence of not handling translation invariance is indeed posterior collapse as $K \to \infty$, while modelling the invariance coincides with the true posterior. An important observation and key takeaway is that, while we need to optimise for (17a), the mean-field posterior approximation $q_0(\mathbf{w}; \boldsymbol{\theta}^*_{\mathrm{mix}})$ is sufficient for prediction. Indeed, for the linear model, we could compute the invariance gap exactly and thereby correct the ELBO objective for the invariances of this model. For more complex non-linear models such as BNNs, this section may serve as a direction for approximating the invariance gap. To this end, the next section describes the layer-wise translation invariance exhibited by BNNs.

5 Translation invariance in Bayesian neural networks

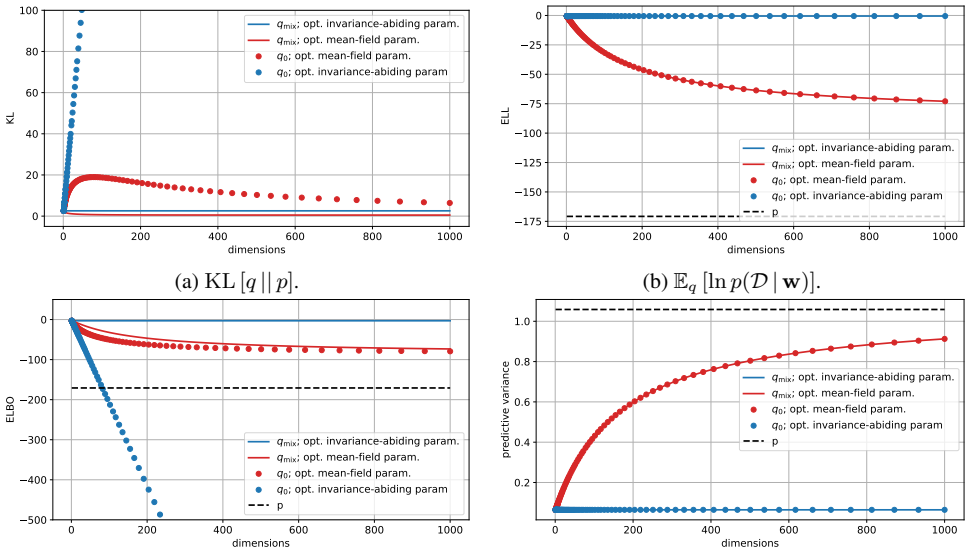
Let us now go back to the general NN model in Sec. 2.1. In order to describe the set of all invariances, note that for each node $z_{l,j}$ in layer l (or the output node y), there is a subspace that keeps the value $z_{l,j}$ (or y) invariant when using the translation-invariance model in (12), because the value itself only depends on the *actual* activation $\mathbf{z}_{l-1}$ (or the input $\mathbf{x}$).

More formally, the following equations model all translation invariances of a NN at a data point $\mathbf{x}$:

$$\forall \boldsymbol{\Delta}_{L,1}: \quad f(\mathbf{x}) = h_L\left(\mathbf{w}_{L,1}^{\mathrm{T}} \mathbf{z}_{L-1}\right) = h_L\left(\left(\mathbf{w}_{L,1} + \mathbf{B}_{\mathbf{z}_{L-1}} \boldsymbol{\Delta}_{L,1}\right)^{\mathrm{T}} \mathbf{z}_{L-1}\right), \tag{20}$$

$$\forall j \in \{1, \ldots, n_l\}, \boldsymbol{\Delta}_{l,j}: \quad z_{l,j} = h_l\left(\mathbf{w}_{l,j}^{\mathrm{T}} \mathbf{z}_{l-1}\right) h_l\left(\left(\mathbf{w}_{l,j} + \mathbf{B}_{\mathbf{z}_{l-1}} \boldsymbol{\Delta}_{l,j}\right)^{\mathrm{T}} \mathbf{z}_{l-1}\right), \tag{21}$$

$$\forall j \in \{1, \ldots, n_1\}, \boldsymbol{\Delta}_{1,j}: \quad z_{1,j} = h_1\left(\mathbf{w}_{1,j}^{\mathrm{T}} \mathbf{x}\right) = h_1\left(\left(\mathbf{w}_{1,j} + \mathbf{B}_{\mathbf{x}} \boldsymbol{\Delta}_{1,j}\right)^{\mathrm{T}} \mathbf{x}\right), \tag{22}$$



(a) $\mathrm{KL}\left[q \,||\, p\right]$.

(b) $\mathbb{E}_q\left[\ln p(\mathcal{D} \,|\, \mathbf{w})\right]$.

(c) $\mathcal{L}_{\mathrm{ELBO}}(q,\, \mathcal{D}) = \mathbb{E}_q\left[\ln p(\mathcal{D} \,|\, \mathbf{w})\right] - \mathrm{KL}\left[q \,||\, p\right]$.

(d) $\mathbb{V}_{\mathbf{w}\sim q}\left[\frac{1}{K}\mathbf{1}^{\mathrm{T}}\mathbf{w}\right] + \sigma_y^2$.

Figure 2: ELBO loss terms and predictive variance for different posterior approximations. Lines/dots indicate the invariance-abiding/mean-field posterior, respectively. Red/blue indicates the optimal invariance-abiding/mean-field parameters (cf. (17)). Prior variance is $\boldsymbol{\sigma}^2 = K \cdot \mathbf{1}$, and $\sigma_y^2 = 1/\left(2\pi\mathrm{e}\right)$; we used $N = 10$ identical observations with value $y = 1$ and inputs $\mathbf{x} = \mathbf{1}$. Notably, both posterior approximations yield the same predictive distribution as can be seen by the ELL (b) and predictive variance (d). The variance of the mean-field approximation reverts to the prior variance as $K \to \infty$.

where the layer index l ranges from $2, \ldots, L-1$ and $\mathbf{B}_{\mathbf{z}}$ is given by

$$\mathbf{B}_{\mathbf{z}} := \begin{bmatrix} & \mathbf{I} & \\ -\frac{z_1}{z_k} & \cdots & -\frac{z_{k-1}}{z_k} \end{bmatrix}.$$

In order to efficiently compute the invariance-abiding likelihood g_{mix}, note that it also decomposes over the NN layers and the associated parameters as follows: For the n_1 nodes in the first layer we have

$$q_{\mathrm{mix}}(\mathbf{w}_{1,j}) = \mathbb{E}_{\mathbf{x}}\left[\mathcal{N}\left(\mathbf{w}_{1,j}; \boldsymbol{\mu} + \frac{\mathbf{x}^{\mathrm{T}}\left(\mathbf{m} - \boldsymbol{\mu}\right)}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}}\left(\boldsymbol{\Sigma}\mathbf{x}\right), \boldsymbol{\Sigma} - \frac{1}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}}\left(\boldsymbol{\Sigma}\mathbf{x}\right)\left(\boldsymbol{\Sigma}\mathbf{x}\right)^{\mathrm{T}}\right)\right], \tag{23}$$

$$P(z_{1,j}) = \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathbf{w}\sim q_{\mathrm{mix}}(\mathbf{w}_{1,j})}\left[h_1\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}\right)\right]\right], \tag{24}$$

where the outer expectation is taken over $\mathbf{x} \sim p(\mathcal{D})$ and $p(\mathcal{D})$ is the empirical distribution over the training set $\mathcal{D}$, and where $\mathbf{m}$, $\mathbf{V}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are constrained to the parts of the overall weight vector that correspond to $\mathbf{w}_{1,j}$. Now, since the invariance-abiding mechanism for nodes in layer l only depends on the *value* $\mathbf{z}_{l-1}$ of the hidden units in layer $n-1$ (see (20) and (21)), we have

$$q_{\mathrm{mix}}(\mathbf{w}_{l,j}) = \mathbb{E}_{\mathbf{z}}\left[\mathcal{N}\left(\mathbf{w}_{l,j}; \boldsymbol{\mu} + \frac{\mathbf{z}^{\mathrm{T}}\left(\mathbf{m} - \boldsymbol{\mu}\right)}{\mathbf{z}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{z}}\left(\boldsymbol{\Sigma}\mathbf{z}\right), \boldsymbol{\Sigma} - \frac{1}{\mathbf{z}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{z}}\left(\boldsymbol{\Sigma}\mathbf{z}\right)\left(\boldsymbol{\Sigma}\mathbf{z}\right)^{\mathrm{T}}\right)\right], \tag{25}$$

$$P(z_{l,j}) = \mathbb{E}_{\mathbf{z}}\left[\mathbb{E}_{\mathbf{w}\sim q_{\mathrm{mix}}(\mathbf{w}_{l,j})}\left[h_l\left(\mathbf{w}^{\mathrm{T}}\mathbf{z}_{l-1}\right)\right]\right], \tag{26}$$

where again $\mathbf{m}$, $\mathbf{V}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are constrained to the parts of the overall weight vector that correspond to $\mathbf{w}_{l,j}$ and the expectation over $\mathbf{z}$ is taken with respect to $\mathbf{z} \sim P(\mathbf{z}_{l-1})$.

Thus, the invariance-abiding approximation could e.g. be computed with a layer-by-layer iterative optimisation: First, given data $\mathbf{x} \sim p(\mathcal{D})$ and prior $p(\mathbf{w})$, use (23) and point estimates for all weight vectors in later layers to optimise the mean $\mathbf{m}$ and diagonal covariance $\mathbf{V}$ of the weights in the first layer, i.e., $\{\mathbf{w}_{1,j} | j = 1, \ldots, n_1\}$. Then, we can use (24) to generate P samples $\mathbf{z}_{1,n}$ of the activations of the first layer under the (now fitted) optimal $q_{\mathrm{mix}}(\{\mathbf{w}_{1,j}\})$. Next, for each of these samples $\mathbf{z}_{1,n}$, we can use (25) and point estimates for all weight vectors in later layers to optimise the mean $\mathbf{m}$ and diagonal covariance $\mathbf{V}$ of the weights in the second layer and average them for the optimal q_{mix} of the weights in the second layer. Finally, we can use (26) to generate samples $\mathbf{z}_{2,i}$ for the second layer.

This process is repeated until the last layer, and iterated until $q_{\text{mix}}(\mathbf{w})$ converges. The complexity of this procedure is no larger than optimising the weight matrices of a single layer, because all previous layers are fully characterised by the distribution of the latent activations $\mathbf{z}_l$ of the hidden layers.

6 Related work

Poor empirical performance of VBNNs has been reported in several previous works, e.g., [35, 15, 31]. For instance, it has been shown that single-layer mean-field VBNNs with ReLU activations can not have large predictive uncertainty between regions of low uncertainty [9]. In contrast, Farquhar et al. [7] argue that mean-field approximations are expressive enough for deep BNNs; they prove a universality result that the predictive distribution of mean-field approximations of BNNs with at least 2 layers of hidden units *can* approximate any true posterior distribution over function values arbitrarily closely. This result seems in conflict with our work and [4], who show that the predictive distribution of mean-field VBNNs reverts to the prior predictive distribution as the network width increases. The discrepancy between these two results is resolved by noting that Farquhar et al. [7] only shows that the expressive power of the model is large enough but not that the approximate inference algorithms will converge to this solution. Our work sheds light on why the mean-field approximation fails to approximate expressive posteriors via the invariance gap in the ELBO. Our framework to model the invariances is similar to and extends previous preliminary works [18, 21].

Other attempts have proposed to approximate the posterior predictive directly, thereby circumventing the non-identifiability issue [26, 19, 34]. While these approaches have shown promising empirical performance, estimating the KL regularisation term in function space is more complicated and can even be ill-defined due to regions of zero prior probability mass [3]. In a similar vein, previous work also proposed to map the BNN prior to a Gaussian process (functional) prior [8, 30].

Surprisingly, restricting the parametrisation of the variational posterior results in competitive or even better performance [28, 20, 29, 6]. For example, Dusenberry et al. [6] proposes a variational approximation only for rank-1 factors, resulting in inference of a lower-dimensional subspace. The outer product of these low-rank factors perturb a weight matrix that is treated deterministically through maximum a posteriori (MAP) estimation. This approach avoids the invariance problem associated with variational Bayesian inference since the MAP estimate is not impeded by this problem and the subspace of the variational weights do not possess the same invariances.

More broadly, (non-)identifiability of parameters and latent variables in probabilistic models is a widely-studied topic from frequentist [25] and Bayesian perspectives [5, 23, 10]. Recently, Wang et al. [33] attributed posterior collapse in variational auto-encoders to non-identifiability of latent variables.

7 Conclusion

We have associated the posterior collapse phenomenon of mean-field VBNNs with invariances in the likelihood function, in particular translation invariance. While the invariance does not affect the predictive distribution, the approximations of the posterior—which abide and do not abide the invariance—differ in the KL regularisation term and consequently in the tightness of the ELBO objective. We proved that the objectives of the two approximations differ by the relative entropy (KL) between the standard mean-field approximation and the invariance-abiding distribution. We related this to a data-related bound on the KL regularisation, which prevents fits for which the gap is large.

We studied over-parametrised Bayesian linear regression as the canonical model that exhibits *translation invariance* and for which the relevant terms can be computed exactly. A detailed analysis of this model confirms our hypothesis that the invariance leads to a significant additional gap in the ELBO objective compared to approximations that model the invariance. It is this very gap which prevents mean-field approximation to achieve the same fit as an invariance-abiding approximation and instead leads to a collapse of the posterior variances to the prior variance.

While we can compute the invariance gap for the over-parametrised linear model, we have not yet identified a computationally efficient procedure for layered models or for other invariances. However, our work provides the mathematical tools to address over-regularisation due to invariances. Future work will focus on approximations of the invariance gap in order to correct the ELBO objective for translation and permutation invariances in general neural network functions.

References

[1] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 1613–1622. JMLR.org, 2015.

[2] J. Brea, B. Simsek, B. Illing, and W. Gerstner. Weight-space symmetry in deep networks gives rise to permutation saddles, connected by equal-loss valleys across the loss landscape. *ArXiv*, abs/1907.02911, 2019.

[3] D. R. Burt, S. W. Ober, A. Garriga-Alonso, and M. van der Wilk. Understanding variational inference in function-space. In *Third Symposium on Advances in Approximate Bayesian Inference*, 2021.

[4] B. Coker, W. P. Bruinsma, D. R. Burt, W. Pan, and F. Doshi-Velez. Wide mean-field variational Bayesian neural networks ignore the data. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2022.

[5] A. P. Dawid. Conditional independence in statistical theory. *Journal of the Royal Statistical Society. Series B (Methodological)*, 41(1):1–31, 1979. ISSN 00359246.

[6] M. W. Dusenberry, G. Jerfel, Y. Wen, Y.-A. Ma, J. Snoek, K. Heller, B. Lakshminarayanan, and D. Tran. Efficient and scalable Bayesian neural nets with rank-1 factors. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

[7] S. Farquhar, L. Smith, and Y. Gal. Liberty or depth: Deep Bayesian neural nets do not need complex weight posterior approximations. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4346–4357. Curran Associates, Inc., 2020.

[8] D. Flam-Shepherd. Mapping gaussian process priors to Bayesian neural networks. In *Bayesian Deep Learning NeurIPS workshop*, 2017.

[9] A. Foong, D. Burt, Y. Li, and R. Turner. On the expressiveness of approximate inference in Bayesian neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15897–15908. Curran Associates, Inc., 2020.

[10] A. E. Gelfand and S. K. Sahu. Identifiability, improper priors, and gibbs sampling for generalized linear models. *Journal of the American Statistical Association*, 94(445):247–253, 1999.

[11] A. Gelman, D. Simpson, and M. Betancourt. The prior can often only be understood in the context of the likelihood. *Entropy*, 19(10), 2017. ISSN 1099-4300. doi: 10.3390/e19100555.

[12] S. Ghosh, J. Yao, and F. Doshi-Velez. Structured variational learning of Bayesian neural networks with horseshoe priors. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1744–1753. PMLR, 10–15 Jul 2018.

[13] R. Herbrich. *Learning Kernel Classifiers: Theory and Algorithms*. The MIT Press, 2nd edition edition, 2002.

[14] G. E. Hinton and D. van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, page 5–13. Association for Computing Machinery, 1993. ISBN 0897916115. doi: 10.1145/168304.168306.

[15] P. Izmailov, S. Vikram, M. D. Hoffman, and A. G. G. Wilson. What are Bayesian neural network posteriors really like? In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4629–4640. PMLR, 18–24 Jul 2021.

[16] A. Kendall and Y. Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[17] R. Kurle, B. Cseke, A. Klushyn, P. van der Smagt, and S. Günnemann. Continual learning with Bayesian neural networks for non-stationary data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[18] R. Kurle, T. Januschowski, J. Gasthaus, and B. Wang. On symmetries in variational Bayesian neural nets. In *Bayesian Deep Learning NeurIPS workshop*, 2021.

[19] C. Ma, Y. Li, and J. M. Hernandez-Lobato. Variational implicit processes. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4222–4233. PMLR, 09–15 Jun 2019.

[20] A. Mishkin, F. Kunstner, D. Nielsen, M. W. Schmidt, and M. E. Khan. Slang: Fast structured covariance approximations for Bayesian deep learning with natural gradient. In *NeurIPS*, pages 6248–6258, 2018.

[21] D. A. Moore. Symmetrized variational inference. *NIPS Workshop on Advances in Approximate Bayesian Inference*, December 2016.

[22] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

[23] D. J. Poirier. Revising beliefs in nonidentified models. *Econometric Theory*, 14(4):483–509, 1998.

[24] A. Pourzanjani, R. M. Jiang, and L. Petzold. Improving the identifiability of neural networks for Bayesian inference. In *Bayesian Deep Learning NeurIPS workshop*, 2017.

[25] B. L. S. Prakasa Rao. *Identifiability in Stochastic Models*. Academic Press, Oxford, 1992.

[26] S. Sun, G. Zhang, J. Shi, and R. B. Grosse. Functional variational Bayesian neural networks. *ArXiv*, abs/1903.05779, 2019.

[27] H. J. Sussmann. Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural Networks*, 5(4):589–593, 1992. ISSN 0893-6080. doi: https://doi.org/10.1016/S0893-6080(05)80037-1.

[28] J. Swiatkowski, K. Roth, B. S. Veeling, L. Tran, J. V. Dillon, J. Snoek, S. Mandt, T. Salimans, R. Jenatton, and S. Nowozin. The k-tied normal distribution: A compact parameterization of gaussian mean field posteriors in Bayesian neural networks. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

[29] M. Tomczak, S. Swaroop, and R. Turner. Efficient low rank gaussian variational inference for neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4610–4622. Curran Associates, Inc., 2020.

[30] B.-H. Tran, S. Rossi, D. Milios, and M. Filippone. All you need is a good functional prior for Bayesian deep learning. *Journal of Machine Learning Research*, 23(74):1–56, 2022.

[31] B. Trippe and R. Turner. Overpruning in variational Bayesian neural networks. *arXiv preprint arXiv:1801.06230*, 2018.

[32] M. J. Wainwright, M. I. Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.

[33] Y. Wang, D. Blei, and J. P. Cunningham. Posterior collapse and latent variable non-identifiability. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 5443–5455. Curran Associates, Inc., 2021.

[34] Z. Wang, T. Ren, J. Zhu, and B. Zhang. Function space particle optimization for Bayesian neural networks. In *International Conference on Learning Representations*, 2019.

[35] F. Wenzel, K. Roth, B. Veeling, J. Swiatkowski, L. Tran, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, and S. Nowozin. How good is the Bayes posterior in deep neural networks really? In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 10248–10259. PMLR, 13–18 Jul 2020.

[36] A. G. Wilson and P. Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4697–4708. Curran Associates, Inc., 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We point out limitations and future work throughout the paper.
   (c) Did you discuss any potential negative societal impacts of your work? [No] This is purely theoretical work without direct negative societal impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...[No] No experiments
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets... [No] No existing assets used.
5. If you used crowdsourcing or conducted research with human subjects... [No] No crowdsourcing/human subjects.

A Notation

We denote matrices and vectors with bold upper- and lower-case letters, e.g. $\mathbf{a}$ and $\mathbf{A}$. In order to address an element of a vector or matrix, we will use non-bold letters, e.g. $\mathbf{x} = [x_1, x_2, \dots, x_N]^{\mathrm{T}}$. We will use the superscript $^{\mathrm{T}}$ to denote a transpose of a vector and $|\mathbf{A}|$ to denote the determinant of the matrix $\mathbf{A}$. We use $\mathrm{Diag}(\mathbf{a})$ to denote the diagonal matrix constructed from a vector, and $\mathrm{diag}(\mathbf{A})$ to denote the diagonal vector of a matrix. We use the notation $\mathcal{N}(\mathbf{x}; \mathbf{m}, \mathbf{V})$ to denote the density of the Gaussian distribution given by

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) := |2\pi\boldsymbol{\Sigma}|^{-\frac{1}{2}} \cdot \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) .$$

An alternative representation of the Gaussian distribution is in terms of its canonical parameters $\boldsymbol{\eta} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$ and $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$:

$$\mathcal{G}(\mathbf{x}; \boldsymbol{\eta}, \boldsymbol{\Lambda}) := \exp\left(\mathbf{x}^{\mathrm{T}}\boldsymbol{\eta} - \frac{1}{2}\mathbf{x}^{\mathrm{T}}\boldsymbol{\Lambda}\mathbf{x} - \frac{1}{2}\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{\Lambda}^{-1}\boldsymbol{\eta} - \frac{1}{2}\ln\left(\left|2\pi\boldsymbol{\Lambda}^{-1}\right|\right)\right) .$$

Note that $\mathcal{G}(\mathbf{x}; \boldsymbol{\eta}_1, \boldsymbol{\Lambda}_1) \cdot \mathcal{G}(\mathbf{x}; \boldsymbol{\eta}_2, \boldsymbol{\Lambda}_2) \propto \mathcal{G}(\mathbf{x}; \boldsymbol{\eta}_1 + \boldsymbol{\eta}_2, \boldsymbol{\Lambda}_1 + \boldsymbol{\Lambda}_2)$ which renders this canonical parameterisation very useful when considering the multiplication of Gaussian densities. Furthermore, we use $\mathbb{E}_{\mathbf{w}\sim p}[f(\mathbf{w})]$ to denote the expectation of the function $f(\cdot)$ when $\mathbf{w}$ is drawn from the distribution p. Also, we use $\mathbb{V}_{\mathbf{w}\sim p}[f(\mathbf{w})]$ to denote the variance of the function $f(\cdot)$ when $\mathbf{w}$ is drawn from the distribution p defined by

$$\mathbb{V}_{\mathbf{w}\sim p}[f(\mathbf{w})] := \mathbb{E}_{\mathbf{w}\sim p}\left[\left(f(\mathbf{w}) - \mathbb{E}_{\mathbf{w}'\sim p}[f(\mathbf{w}')]\right)^2\right] .$$

B Convolution of Gaussian Measures

Theorem 1 (Convolution of Gaussian Measures)**.** *For any* $\mathbf{A} \in \mathbb{R}^{n\times k}$ *and* $\mathbf{b} \in \mathbb{R}^n$ *it holds that*

$$p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}; \mathbf{A}\boldsymbol{\theta} + \mathbf{b}, \mathbf{V}) \ , \ p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \ ,$$

implies that

$$p(\boldsymbol{\theta}|\mathbf{x}) = \mathcal{N}\left(\boldsymbol{\theta}; \mathbf{C}^{-1}\left(\mathbf{A}^T\mathbf{V}^{-1}(\mathbf{x}-\mathbf{b}) + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right), \mathbf{C}^{-1}\right) , \tag{27}$$

$$p(\mathbf{x}) = \mathcal{N}\left(\mathbf{x}; \mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{V} + \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T\right) , \tag{28}$$

$$\mathbf{C} = \mathbf{A}^T\mathbf{V}^{-1}\mathbf{A} + \boldsymbol{\Sigma}^{-1} .$$

Proof. Let's start by proving (27). First, we note that

$$p\left(\boldsymbol{\theta}|\mathbf{x}\right) = \frac{p\left(\mathbf{x}|\boldsymbol{\theta}\right) \cdot p\left(\boldsymbol{\theta}\right)}{\int p\left(\mathbf{x}|\widetilde{\boldsymbol{\theta}}\right) \cdot p\left(\widetilde{\boldsymbol{\theta}}\right) \, d\widetilde{\boldsymbol{\theta}}} = \frac{\mathcal{N}\left(\mathbf{x}; \mathbf{A}\boldsymbol{\theta} + \mathbf{b}, \mathbf{V}\right) \cdot \mathcal{N}\left(\boldsymbol{\theta}; \boldsymbol{\mu}, \boldsymbol{\Sigma}\right)}{\int \mathcal{N}\left(\mathbf{x}; \mathbf{A}\widetilde{\boldsymbol{\theta}} + \mathbf{b}, \mathbf{V}\right) \cdot \mathcal{N}\left(\widetilde{\boldsymbol{\theta}}; \boldsymbol{\mu}, \boldsymbol{\Sigma}\right) \, d\widetilde{\boldsymbol{\theta}}} \,,$$

where only the numerator depends on $\boldsymbol{\theta}$. Thus, the numerator is given by

$$c \cdot \exp\left(-\frac{1}{2}\left[\left(\left(\mathbf{x} - \mathbf{b}\right) - \mathbf{A}\boldsymbol{\theta}\right)^{\mathrm{T}} \mathbf{V}^{-1} \left(\left(\mathbf{x} - \mathbf{b}\right) - \mathbf{A}\boldsymbol{\theta}\right) + \left(\boldsymbol{\theta} - \boldsymbol{\mu}\right)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{\theta} - \boldsymbol{\mu}\right)\right]\right) \,,$$

where $c = |2\pi\mathbf{V}|^{-\frac{1}{2}} \cdot |2\pi\boldsymbol{\Sigma}|^{-\frac{1}{2}}$ is independent of $\mathbf{x}$ and $\boldsymbol{\theta}$. Using Theorem A.86 in [13], this quadratic form in $\boldsymbol{\theta}$ can be rewritten as

$$c \cdot \exp\left(-\frac{1}{2}\left[\left(\boldsymbol{\theta} - \mathbf{c}\right)^{\mathrm{T}} \mathbf{C} \left(\boldsymbol{\theta} - \mathbf{c}\right) + d\left(\mathbf{x}\right)\right]\right) \,,$$

where

$$\begin{aligned} \mathbf{C} &= \mathbf{A}^{\mathrm{T}}\mathbf{V}^{-1}\mathbf{A} + \boldsymbol{\Sigma}^{-1} \,, \\ \mathbf{C}\mathbf{c} &= \mathbf{A}^{\mathrm{T}}\mathbf{V}^{-1}\left(\mathbf{x} - \mathbf{b}\right) + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \,, \\ d\left(\mathbf{x}\right) &= \left(\mathbf{x} - \mathbf{b} - \mathbf{A}\boldsymbol{\mu}\right)^{\mathrm{T}} \left(\mathbf{V} + \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^{\mathrm{T}}\right)^{-1} \left(\mathbf{x} - \mathbf{b} - \mathbf{A}\boldsymbol{\mu}\right) \,. \end{aligned} \tag{29}$$

Since $d(\mathbf{x})$ does not depend on $\boldsymbol{\theta}$, it get incorporated into the normalization constant which proves (27).

In order to prove (28), note that

$$\begin{aligned} p\left(\mathbf{x}\right) &= \int p\left(\mathbf{x}|\widetilde{\boldsymbol{\theta}}\right) \cdot p\left(\widetilde{\boldsymbol{\theta}}\right) \, d\widetilde{\boldsymbol{\theta}} \\ &= \int \mathcal{N}\left(\mathbf{x}; \mathbf{A}\widetilde{\boldsymbol{\theta}} + \mathbf{b}, \mathbf{V}\right) \cdot \mathcal{N}\left(\widetilde{\boldsymbol{\theta}}; \boldsymbol{\mu}, \boldsymbol{\Sigma}\right) \, d\widetilde{\boldsymbol{\theta}} \\ &= \int c \cdot \exp\left(-\frac{1}{2}\left[\left(\widetilde{\boldsymbol{\theta}} - \mathbf{c}\right)^{\mathrm{T}} \mathbf{C} \left(\widetilde{\boldsymbol{\theta}} - \mathbf{c}\right) + d\left(\mathbf{x}\right)\right]\right) \, d\widetilde{\boldsymbol{\theta}} \\ &= c \cdot \exp\left(-\frac{1}{2} d\left(\mathbf{x}\right)\right) \cdot \int \exp\left(-\frac{1}{2}\left[\left(\widetilde{\boldsymbol{\theta}} - \mathbf{c}\right)^{\mathrm{T}} \mathbf{C} \left(\widetilde{\boldsymbol{\theta}} - \mathbf{c}\right)\right]\right) \, d\widetilde{\boldsymbol{\theta}} \\ &= c \cdot \exp\left(-\frac{1}{2} d\left(\mathbf{x}\right)\right) \cdot \left|2\pi\mathbf{C}^{-1}\right|^{\frac{1}{2}} \\ &= \tilde{c} \cdot \exp\left(-\frac{1}{2}\left(\left(\mathbf{x} - \left(\mathbf{A}\boldsymbol{\mu} + \mathbf{b}\right)\right)^{\mathrm{T}} \left(\mathbf{V} + \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^{\mathrm{T}}\right)^{-1} \left(\mathbf{x} - \left(\mathbf{A}\boldsymbol{\mu} + \mathbf{b}\right)\right)\right)\right) \\ &= \mathcal{N}\left(\mathbf{x}; \mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{V} + \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^{\mathrm{T}}\right) \,, \end{aligned}$$

where the penultimate line follows again from Theorem A.86 in [13] with $d(\mathbf{x})$ defined in (29). □

Theorem 2 (Woodbury formula)**.** *Let* $\mathbf{C}$ *be an invertible* $n \times n$ *matrix. Then, for any* $\mathbf{A} \in \mathbb{R}^{n \times k}$ *and* $\mathbf{B} \in \mathbb{R}^{k \times n}$*,*

$$\left(\mathbf{C} + \mathbf{A}\mathbf{B}\right)^{-1} = \mathbf{C}^{-1} - \mathbf{C}^{-1}\mathbf{A}\left(\mathbf{I} + \mathbf{B}\mathbf{C}^{-1}\mathbf{A}\right)^{-1}\mathbf{B}\mathbf{C}^{-1} \,.$$

C Invariance gap and posterior-predictive equivalence

In this section, we show that the posterior predictive distribution of the mean-field and invariance-abiding distribution are identical (for identical parametrisation of the likelihood approximation) as stated in (9), and that the difference in the respective KL regularisation terms can be quantified by the KL defined in (10).

We first discuss the conditions from (7) and (8). The goal is to use the invariance property of the likelihood function (4). Unfortunately, the prior does not generally have the same invariance. Yet, the mean-field approximate posterior defined as the product between prior and likelihood approximation

can still have the invariance property, as we show for permutation and translation invariance with mean-field Gaussian likelihood approximation and prior. This condition is defined in (7) for each mean-field posterior in the integral over $\mathbf{r}$:

$$\begin{aligned} q_{\text{mix}}(\mathbf{w};\boldsymbol{\theta}) &= Z_{\text{mix}}^{-1}\, p(\mathbf{w}) \cdot \int p(\mathbf{r})\, g_0(t(\mathbf{w},\mathbf{r});\boldsymbol{\theta}) d\mathbf{r} = \int p(\mathbf{r})\, \frac{1}{Z_{\text{mix}}}\, p(\mathbf{w}) \cdot g_0(t(\mathbf{w},\mathbf{r});\boldsymbol{\theta}) d\mathbf{r} \\ &= \int p(\mathbf{r})\, \frac{1}{Z_{\text{mix}}}\, p(t(\mathbf{w},\varphi(\mathbf{r}))) \cdot g_0(t(\mathbf{w},\varphi(\mathbf{r}));\boldsymbol{\theta}) d\mathbf{r} \\ &= \int p(\mathbf{r})\, \frac{Z_0(\mathbf{r})}{Z_{\text{mix}}}\, q_0(t(\mathbf{w},\varphi(\mathbf{r}));\boldsymbol{\theta}) d\mathbf{r}, \end{aligned}$$

where Z_{mix} and $Z_0(\mathbf{r})$ are normalisation constants. The second line introduced the assumption that there exists a surjective mapping $\mathbf{r}' = \varphi(\mathbf{r})$ such that the product of the untransformed prior $p(\mathbf{w})$ and the transformed likelihood approximation $g_0(t(\mathbf{w},\varphi(\mathbf{r}));\boldsymbol{\theta})$ are identical to the product of the transformed prior and likelihood approximation with $\mathbf{r}'$, i.e. as stated in the main text,

$$\forall \mathbf{r} \sim p(\mathbf{r}) : p(\mathbf{w}) \cdot g_0(t(\mathbf{w},\mathbf{r});\boldsymbol{\theta}) = p(t(\mathbf{w},\varphi(\mathbf{r}))) \cdot g_0(t(\mathbf{w},\varphi(\mathbf{r}));\boldsymbol{\theta}).$$

This condition may seem quite limiting, however, we show that this condition holds for *permutation* invariance with the isotropic Gaussian prior and for *translation* invariance with Gaussian priors and Gaussian likelihood approximation.

Condition (7) **and** (8) **for permutation invariance.** Consider first permutation invariance (cf. App. E). It is easy to see that for the isotropic Gaussian prior: $p(\mathbf{w}) = p(\mathbf{P_r w})$. Also, note that $\mathcal{G}(\mathbf{P}\mathbf{w};\, \boldsymbol{\eta}, \boldsymbol{\Lambda}) = \mathcal{G}(\mathbf{w}; \mathbf{P}^{\mathrm{T}}\boldsymbol{\eta}, \mathbf{P}^{\mathrm{T}}\boldsymbol{\Lambda}\mathbf{P})$. Thus, the product between the untransformed Gaussian prior and the permuted Gaussian likelihood approximation is

$$\begin{aligned} p(\mathbf{w}) \cdot g_0(\mathbf{P}\mathbf{w}) = p(\mathbf{P}\mathbf{w}) \cdot g_0(\mathbf{P}\mathbf{w}) &= \mathcal{G}(\mathbf{P}\mathbf{w}; \boldsymbol{\eta}_{\text{p}}, \boldsymbol{\Lambda}_{\text{p}}) \cdot \mathcal{G}(\mathbf{P}\mathbf{w}; \boldsymbol{\eta}_g, \boldsymbol{\Lambda}_g) && (30)\\ &= \mathcal{G}(\mathbf{w}; \mathbf{P}^{\mathrm{T}}\boldsymbol{\eta}_{\text{p}}, \mathbf{P}^{\mathrm{T}}\boldsymbol{\Lambda}_{\text{p}}\mathbf{P}) \cdot \mathcal{G}(\mathbf{w}; \mathbf{P}^{\mathrm{T}}\boldsymbol{\eta}_g, \mathbf{P}^{\mathrm{T}}\boldsymbol{\Lambda}_g\mathbf{P}) && (31)\\ &\propto \mathcal{G}(\mathbf{w}; \mathbf{P}^{\mathrm{T}}(\boldsymbol{\eta}_{\text{p}} + \boldsymbol{\eta}_g), \mathbf{P}^{\mathrm{T}}(\boldsymbol{\Lambda}_{\text{p}} + \boldsymbol{\Lambda}_g)\mathbf{P}) && (32)\\ &= \mathcal{G}(\mathbf{P}\mathbf{w}; \boldsymbol{\eta}_{\text{p}} + \boldsymbol{\eta}_g, \boldsymbol{\Lambda}_{\text{p}} + \boldsymbol{\Lambda}_g) = q_0(\mathbf{P}\mathbf{w}). && (33) \end{aligned}$$

Hence, for permutation invariance with the isotropic Gaussian prior and Gaussian likelihood approximation, we have

$$\forall \mathbf{r} \sim p(\mathbf{r}) : p(\mathbf{w}) \cdot g_0(\mathbf{P_r w}) = p(\mathbf{P_r w}) \cdot g_0(\mathbf{P_r w}) \propto q_0(\mathbf{P_r w}). \tag{34}$$

The invariance transformation is thus simply $t(\mathbf{w}, \varphi(\mathbf{r})) = t(\mathbf{w}, \mathbf{r}) = \mathbf{P_r w}$ (i.e. we do not need the mapping φ). This is because the prior has no preference over the permutation-induced modes of the likelihood. Thus, we have

$$\forall \mathbf{r} \sim p(\mathbf{r}) : \left| \det \frac{\partial t(\mathbf{w},\mathbf{r})}{\partial \mathbf{w}} \right|^{-1} = \left| \det \frac{\partial \mathbf{P_r w}}{\partial \mathbf{w}} \right|^{-1} = |\det \mathbf{P_r}|^{-1} = 1\,.$$

Condition (7) **and** (8) **for translation invariance.** Next, consider translation invariance, and note that $\mathcal{N}(\mathbf{w} - \mathbf{v}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu} + \mathbf{v}, \boldsymbol{\Sigma})$, and, consequently, $\mathcal{G}(\mathbf{w} - \mathbf{v}; \boldsymbol{\eta}, \boldsymbol{\Lambda}) = \mathcal{G}(\mathbf{w}; \boldsymbol{\eta} + \boldsymbol{\Lambda}\mathbf{v}, \boldsymbol{\Lambda})$. It is easier to show directly that the product between the untransformed Gaussian prior and translated Gaussian likelihood approximation can be written as a Gaussian posterior translated, because:

$$\begin{aligned} p(\mathbf{w}) \cdot g_0(\mathbf{w} - \mathbf{v}) &= \mathcal{G}(\mathbf{w}; \boldsymbol{\eta}_{\text{p}}, \boldsymbol{\Lambda}_{\text{p}}) \cdot \mathcal{G}(\mathbf{w} - \mathbf{v}; \boldsymbol{\eta}_g, \boldsymbol{\Lambda}_g) && (35)\\ &= \mathcal{G}(\mathbf{w}; \boldsymbol{\eta}_{\text{p}}, \boldsymbol{\Lambda}_{\text{p}}) \cdot \mathcal{G}(\mathbf{w}; \boldsymbol{\eta}_g + \boldsymbol{\Lambda}_g\mathbf{v}, \boldsymbol{\Lambda}_g) && (36)\\ &\propto \mathcal{G}(\mathbf{w}; \boldsymbol{\eta}_{\text{p}} + \boldsymbol{\eta}_g + \boldsymbol{\Lambda}_g\mathbf{v}, \boldsymbol{\Lambda}_{\text{p}} + \boldsymbol{\Lambda}_g) && (37)\\ &= \mathcal{G}\Big(\mathbf{w}; \boldsymbol{\eta}_{\text{p}} + \boldsymbol{\eta}_g + \overbrace{(\boldsymbol{\Lambda}_{\text{p}} + \boldsymbol{\Lambda}_g)\,(\boldsymbol{\Lambda}_{\text{p}} + \boldsymbol{\Lambda}_g)^{-1}}^{\mathbf{I}} \boldsymbol{\Lambda}_g\mathbf{v}, \boldsymbol{\Lambda}_{\text{p}} + \boldsymbol{\Lambda}_g\Big) && (38)\\ &= \mathcal{G}\left(\mathbf{w} - (\boldsymbol{\Lambda}_{\text{p}} + \boldsymbol{\Lambda}_g)^{-1} \boldsymbol{\Lambda}_g\mathbf{v}; \boldsymbol{\eta}_{\text{p}} + \boldsymbol{\eta}_g, \boldsymbol{\Lambda}_{\text{p}} + \boldsymbol{\Lambda}_g\right) && (39)\\ &= q_0\left(\mathbf{w} - (\boldsymbol{\Lambda}_{\text{p}} + \boldsymbol{\Lambda}_g)^{-1} \boldsymbol{\Lambda}_g\mathbf{v}\right). && (40) \end{aligned}$$

Since the precision matrices are diagonal, $\left(\left(\mathbf{\Lambda}_\mathrm{p}+\mathbf{\Lambda}_g\right)^{-1}\mathbf{\Lambda}_g\right)\mathbf{B}\mathbf{r} = \mathbf{B}\left(\left(\mathbf{\Lambda}_\mathrm{p}+\mathbf{\Lambda}_g\right)^{-1}\mathbf{\Lambda}_g\right)\mathbf{r}$. Consequently, for the translation invariance $g_0(\mathbf{w}) = g_0(t(\mathbf{w},\mathbf{r})) = g_0(\mathbf{w}-\mathbf{B}\mathbf{r})$, we have

$$\forall \mathbf{r}\sim p(\mathbf{r}) : p(\mathbf{w})\cdot g_0(\mathbf{w}-\mathbf{B}\mathbf{r}) = p\left(\mathbf{w}-\mathbf{B}\mathbf{r}'\right)\cdot g_0\left(\mathbf{w}-\mathbf{B}\mathbf{r}'\right) \propto q_0\left(\mathbf{w}-\mathbf{B}\mathbf{r}'\right), \tag{41}$$

where $\mathbf{r}' = \varphi(\mathbf{r}) = \left(\left(\mathbf{\Lambda}_\mathrm{p}+\mathbf{\Lambda}_g\right)^{-1}\mathbf{\Lambda}_g\right)\mathbf{r}$. Thus, we have

$$\forall \mathbf{r}\sim p(\mathbf{r}) : \left|\det\frac{\partial t(\mathbf{w},\mathbf{r})}{\partial \mathbf{w}}\right|^{-1} = \left|\det\frac{\partial(\mathbf{w}-\mathbf{B}\mathbf{r})}{\partial \mathbf{w}}\right|^{-1} = \left|\det \mathbf{I}\right|^{-1} = 1\,.$$

Lemma 1. *For any distribution $p(\mathbf{w})$, likelihood approximation $g_0(\mathbf{w};\boldsymbol{\theta})$, $q_0(\mathbf{w};\boldsymbol{\theta})$ as defined in* (6)*, $q_\mathrm{mix}(\mathbf{w};\boldsymbol{\theta})$ as defined in* (6b) *and $p(\mathbf{r})$, assume there exists a mapping $\varphi : \mathbf{r}\mapsto\mathbf{r}'$ such that*

$$\forall \mathbf{r}\sim p(\mathbf{r}) : p(\mathbf{w})\cdot g_0(t(\mathbf{w},\mathbf{r});\boldsymbol{\theta}) = p(t(\mathbf{w},\varphi(\mathbf{r})))\cdot g_0(t(\mathbf{w},\varphi(\mathbf{r}));\boldsymbol{\theta}), \tag{42}$$

as well as

$$\forall \mathbf{r}\sim p(\mathbf{r}) : \left|\det\frac{\partial t(\mathbf{w},\mathbf{r})}{\partial \mathbf{w}}\right|^{-1} = 1\,. \tag{43}$$

Then,

$$\mathbb{E}_{\mathbf{w}\sim q_\mathrm{mix}(\mathbf{w};\boldsymbol{\theta})}\left[\ln p(\mathcal{D}\,|\,\mathbf{w})\right] = \mathbb{E}_{\mathbf{w}\sim q_0(\mathbf{w};\boldsymbol{\theta})}\left[\ln p(\mathcal{D}\,|\,\mathbf{w})\right]\,. \tag{44}$$

Proof. The lemma can be proven by applying the change-of-variables formula and using the invariance property (42) of $g(\cdot;\boldsymbol{\theta})$:

$$\begin{aligned}
&\mathbb{E}_{\mathbf{w}\sim q_\mathrm{mix}(\mathbf{w};\boldsymbol{\theta})}\left[\ln p(\mathcal{D}\,|\,\mathbf{w})\right] && (45)\\
&= \int \overbrace{\int p(\mathbf{r})\frac{Z_0(\mathbf{r})}{Z_\mathrm{mix}} q_0(t(\mathbf{w},\varphi(\mathbf{r}));\,\boldsymbol{\theta})\,d\mathbf{r}}^{q_\mathrm{mix}(\mathbf{w})} \ln\left[p(\mathcal{D}\,|\,\mathbf{w})\right]\,d\mathbf{w} && (46)\\
&= \int p(\mathbf{r})\frac{Z_0(\mathbf{r})}{Z_\mathrm{mix}}\int q_0(t(\mathbf{w},\varphi(\mathbf{r}));\,\boldsymbol{\theta})\,\ln\left[p(\mathcal{D}\,|\,\mathbf{w})\right]\,d\mathbf{w}\,d\mathbf{r} && (47)\\
&= \int p(\mathbf{r})\frac{Z_0(\mathbf{r})}{Z_\mathrm{mix}}\int q_0(t(\mathbf{w},\varphi(\mathbf{r}));\,\boldsymbol{\theta})\,\ln\left[p(\mathcal{D}\,|\,t(\mathbf{w},\varphi(\mathbf{r})))\right]\,d\mathbf{w}\,d\mathbf{r} && (48)\\
&= \int p(\mathbf{r})\frac{Z_0(\mathbf{r})}{Z_\mathrm{mix}}\int q_0(\mathbf{w};\,\boldsymbol{\theta})\overbrace{\left|\det\frac{\partial t(\mathbf{w},\varphi(\mathbf{r}))}{\partial \mathbf{w}}\right|^{-1}}^{1}\ln\left[p(\mathcal{D}\,|\,\mathbf{w})\right]\,d\mathbf{w}\,d\mathbf{r} && (49)\\
&= \overbrace{\int p(\mathbf{r})\frac{Z_0(\mathbf{r})}{Z_\mathrm{mix}}d\mathbf{r}}^{1}\int q_0(\mathbf{w};\,\boldsymbol{\theta})\ln\left[p(\mathcal{D}\,|\,\mathbf{w})\right]\,d\mathbf{w} && (50)\\
&= \mathbb{E}_{\mathbf{w}\sim q_0}\left[\ln p(\mathcal{D}\,|\,\mathbf{w})\right]. && (51)
\end{aligned}$$

where the second line uses (42), the third line changes the order of integration (Fubini's theorem), the fourth line uses the invariance property $\ln p(\mathcal{D}\,|\,\mathbf{w}) = \ln p(\mathcal{D}\,|\,t(\mathbf{w},\varphi(\mathbf{r})))$, the fifth line then applies the change of variables theorem together with (43), the sixth line then uses again the invariance property of the log likelihood, and the seventh line notes that the integral over the normalisation constants $Z_0(\mathbf{r})$ cancels with the normalisation constant Z_mix of the invariance-abiding posterior, since

$$\begin{aligned}
\int p(\mathbf{r})Z_0(\mathbf{r})d\mathbf{r} &= \int\int p(\mathbf{r})\,p(\mathbf{w})\cdot g_0(t(\mathbf{w},\mathbf{r}))\,d\mathbf{w}d\mathbf{r} && (52)\\
&= \int p(\mathbf{w})\cdot\int p(\mathbf{r})\,g_0(t(\mathbf{w},\mathbf{r}))\,d\mathbf{r}d\mathbf{w} && (53)\\
&= \int p(\mathbf{w})\cdot g_\mathrm{mix}(\mathbf{w})\,d\mathbf{w} = Z_\mathrm{mix}, && (54)
\end{aligned}$$

where we changed the integration order, resulting in the normalisation constant of the invariance-abiding likelihood approximation. □

Lemma 2. *For any distribution $p(\mathbf{w})$, likelihood approximation $g_0(\mathbf{w};\boldsymbol{\theta})$, $q_0(\mathbf{w};\boldsymbol{\theta})$ as defined in* (6)*, $q_{\text{mix}}(\mathbf{w};\boldsymbol{\theta})$ as defined in* (6b) *and $p(\mathbf{r})$, assume there exist two mappings $t : \mathbf{w} \times \mathbf{r} \mapsto \mathbf{w}'$ and $\varphi : \mathbf{r} \mapsto \mathbf{r}'$ such that* (42) *and* (43) *hold. Then,*

$$\mathrm{KL}\left[q_0 \,||\, p\right] - \mathrm{KL}\left[q_{\text{mix}} \,||\, p\right] = \mathrm{KL}\left[q_0 \,||\, q_{\text{mix}}\right] . \tag{55}$$

Proof. First, note that

$$\mathrm{KL}\left[q_0 \,||\, p\right] = \mathbb{E}_{\mathbf{w}\sim q_0}\left[\ln\left[Z_0\, g_0(\mathbf{w})\right]\right] , \tag{56}$$
$$\mathrm{KL}\left[q_{\text{mix}} \,||\, p\right] = \mathbb{E}_{\mathbf{w}\sim q_{\text{mix}}}\left[\ln\left[Z_{\text{mix}}\, g_{\text{mix}}(\mathbf{w})\right]\right] . \tag{57}$$

Now, in the latter KL, we expand the distribution $q_{\text{mix}}(\mathbf{w})$ as in (42),

$$\mathrm{KL}\left[q_{\text{mix}} \,||\, p\right] = \int \overbrace{\int p(\mathbf{r}) \frac{Z_0(\mathbf{r})}{Z_{\text{mix}}} q_0(t(\mathbf{w},\varphi(\mathbf{r}));\,\boldsymbol{\theta}) d\mathbf{r}}^{q_{\text{mix}}(\mathbf{w})} \ln\left[Z_{\text{mix}}\, g_{\text{mix}}(\mathbf{w})\right]\, d\mathbf{w} \tag{58}$$
$$= \int\int p(\mathbf{r}) \frac{Z_0(\mathbf{r})}{Z_{\text{mix}}} q_0(t(\mathbf{w},\varphi(\mathbf{r}));\,\boldsymbol{\theta})\, \ln\left[Z_{\text{mix}}\, g_{\text{mix}}(\mathbf{w})\right]\, d\mathbf{r} d\mathbf{w} \tag{59}$$
$$= \int\int p(\mathbf{r}) \frac{Z_0(\mathbf{r})}{Z_{\text{mix}}} q_0(t(\mathbf{w},\varphi(\mathbf{r}));\,\boldsymbol{\theta})\, \ln\left[Z_{\text{mix}}\, g_{\text{mix}}(t(\mathbf{w},\varphi(\mathbf{r})))\right]\, d\mathbf{r} d\mathbf{w} \tag{60}$$
$$= \int\int p(\mathbf{r}) \frac{Z_0(\mathbf{r})}{Z_{\text{mix}}} q_0(\mathbf{w};\,\boldsymbol{\theta}) \overbrace{\left|\det \frac{\partial t(\mathbf{w},\varphi(\mathbf{r}))}{\partial \mathbf{w}}\right|^{-1}}^{1} \ln\left[Z_{\text{mix}}\, g_{\text{mix}}(\mathbf{w})\right]\, d\mathbf{r} d\mathbf{w} \tag{61}$$
$$= \int q_0(\mathbf{w};\,\boldsymbol{\theta}) \overbrace{\int p(\mathbf{r}) \frac{Z_0(\mathbf{r})}{Z_{\text{mix}}}\, d\mathbf{r}}^{1} \ln\left[Z_{\text{mix}}\, g_{\text{mix}}(\mathbf{w})\right] d\mathbf{w} \tag{62}$$
$$= \int q_0(\mathbf{w};\,\boldsymbol{\theta})\, \ln\left[Z_{\text{mix}}\, g_{\text{mix}}(\mathbf{w})\right]\, d\mathbf{w}' , \tag{63}$$

where the third line uses the invariance property of the invariance-abiding likelihood approximation, $\forall \mathbf{r} : g_{\text{mix}}(t(\mathbf{w},\varphi(\mathbf{r}))) = g_{\text{mix}}(\mathbf{w})$, the change of variables formula is applied to the fourth line for the volume-preserving transformation, and the fifth line re-arranges the integration, noting that the integral over normalisation constants equals one.

The proposition follows by taking the difference between the regularisation terms corresponding to the respective ELBO objectives:

$$\mathrm{KL}\left[q_0 \,||\, p\right] - \mathrm{KL}\left[q_{\text{mix}} \,||\, p\right] = \mathbb{E}_{\mathbf{w}\sim q_0}\left[\ln \frac{Z_0\, g_0(\mathbf{w})}{Z_{\text{mix}} g_{\text{mix}}(\mathbf{w})}\right] \tag{64}$$
$$= \mathbb{E}_{\mathbf{w}\sim q_0}\left[\ln \frac{Z_0\, p(\mathbf{w}) g_0(\mathbf{w})}{Z_{\text{mix}} p(\mathbf{w}) g_{\text{mix}}(\mathbf{w})}\right] = \mathrm{KL}\left[q_0 \,||\, q_{\text{mix}}\right] . \tag{65}$$

□

D Translation invariance in linear models

In this section, we derive the results from Sec. 4 for a Bayesian linear regression with a single input vector $\mathbf{x}$ and corresponding target observation y. The result stated in Sec. 4 for $\mathbf{x} = \mathbf{1}$ follows as a special case.

We assume we have K latent variables $\mathbf{w} = [w_1, \ldots, w_K]^{\mathrm{T}}$ and one observation $\mathbf{x} = [x_1, \ldots, x_K]^{\mathrm{T}}$. We further assume that the likelihood $p\left(y, |\mathbf{w}, \mathbf{x}\right)$ only depend on their inner product, that is, $\mathbf{x}^{\mathrm{T}}\mathbf{w}$. Then we know that any change in $\mathbf{w}$, which leaves the sum of the elements weighted by $\mathbf{x}$ unaffected, does not change the likelihood. For example $\mathbf{w}' = [w_1 + \Delta, w_2 - \Delta \cdot \frac{x_1}{x_2}, w_3, \ldots, w_K]^{\mathrm{T}}$ has the exact same likelihood than $\mathbf{w} = [w_1, w_2, w_3, \ldots, w_K]^{\mathrm{T}}$. In general, we can model this translation

invariance using an $K-1$ dimensional vector $\mathbf{\Delta} \in \mathbb{R}^{K-1}$ and noting that

$$\mathbf{x}^{\mathrm{T}}\mathbf{w} = \mathbf{x}^{\mathrm{T}}\left(\mathbf{w} + \underbrace{\begin{bmatrix} \mathbf{I} \\ -x_K^{-1}\mathbf{x}_{K-1}^{\mathrm{T}} \end{bmatrix}}_{\mathbf{B}} \mathbf{\Delta}\right),$$

where we used $\mathbf{x}_{K-1} := [x_1, \ldots, x_{K-1}]^{\mathrm{T}}$ because

$$\mathbf{x}^{\mathrm{T}}\mathbf{B}\mathbf{\Delta} = \begin{bmatrix} \mathbf{x}_{K-1}^{\mathrm{T}} & x_K \end{bmatrix} \begin{bmatrix} \mathbf{\Delta} \\ -x_K^{-1}\mathbf{x}_{K-1}^{\mathrm{T}}\mathbf{\Delta} \end{bmatrix} = 0\,.$$

D.1 Likelihood Model

Now let us assume that we approximate the likelihood by a function that has Gaussian shape, that is $p\left(y, |\mathbf{w}, \mathbf{x}\right) \approx q\left(\mathbf{w}\right) \propto \mathcal{N}\left(\mathbf{w}; \mathbf{m}, \mathbf{V}\right)$. In order to model that the likelihood is translation invariant, we compute the marginal $q\left(\mathbf{w} - \mathbf{B}\mathbf{\Delta}\right)$ over $p\left(\mathbf{\Delta}\right) = \mathcal{N}\left(\mathbf{\Delta}; \mathbf{0}, \beta^2\mathbf{I}\right)$ and considering the case of $\beta \to \infty$, that is

$$\begin{aligned} q_\beta\left(\mathbf{w}\right) &:= \int q\left(\mathbf{w} - \mathbf{B}\mathbf{\Delta}\right) \cdot p\left(\mathbf{\Delta}\right)\, d\mathbf{\Delta} \\ &= \int \mathcal{N}\left(\mathbf{w}; \mathbf{m} + \mathbf{B}\mathbf{\Delta}, \mathbf{V}\right) \cdot \mathcal{N}\left(\mathbf{\Delta}; \mathbf{0}, \beta^2\mathbf{I}\right)\, d\mathbf{\Delta}\,. \end{aligned}$$

According to Theorem 1, for any $\beta \in \mathbb{R}^+$ this is another Gaussian given by

$$\begin{aligned} q_\beta\left(\mathbf{w}\right) &= \mathcal{N}\left(\mathbf{w}; \mathbf{m} + \mathbf{B}\mathbf{0}, \underbrace{\mathbf{V} + \beta\mathbf{B} \cdot \beta\mathbf{B}^{\mathrm{T}}}_{\mathbf{V}_\beta}\right) \\ &= \mathcal{N}\left(\mathbf{w}; \mathbf{m}, \mathbf{V} + \beta^2 \cdot \begin{bmatrix} \mathbf{I} & -x_K^{-1}\mathbf{x}_{K-1} \\ -x_K^{-1}\mathbf{x}_{K-1}^{\mathrm{T}} & x_K^{-2}\mathbf{x}_{K-1}^{\mathrm{T}}\mathbf{x}_{K-1} \end{bmatrix}\right). \end{aligned}$$

Note that $\lim_{\beta\to\infty} q_\beta\left(\mathbf{w}\right) = g_{\mathrm{mix}}(\mathbf{w})$ as defined in Subsection 4.1. Using the Woodbury formula in Theorem 2, the inverse of the covariance can be re-written as

$$\begin{aligned} \mathbf{V}_\beta^{-1} &:= \left(\mathbf{V} + \beta\mathbf{B} \cdot \beta\mathbf{B}^{\mathrm{T}}\right)^{-1} \\ &= \mathbf{V}^{-1} - \beta^2 \cdot \mathbf{V}^{-1}\mathbf{B}\left(\mathbf{I} + \beta^2\mathbf{B}^{\mathrm{T}}\mathbf{V}^{-1}\mathbf{B}\right)^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{V}^{-1} \\ &= \mathbf{V}^{-1} - \beta^2 \cdot \mathbf{V}^{-1}\mathbf{B}\left(\beta^2\left(\beta^{-2}\mathbf{I} + \mathbf{B}^{\mathrm{T}}\mathbf{V}^{-1}\mathbf{B}\right)\right)^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{V}^{-1} \\ &= \mathbf{V}^{-1} - \mathbf{V}^{-1}\mathbf{B}\left(\beta^{-2}\mathbf{I} + \mathbf{B}^{\mathrm{T}}\mathbf{V}^{-1}\mathbf{B}\right)^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{V}^{-1}\,. \end{aligned}$$

D.2 Posterior Model

If we assume a prior $p\left(\mathbf{w}\right) = \mathcal{N}\left(\mathbf{w}; \boldsymbol{\mu}, \mathbf{\Sigma}\right)$, then the posterior for the Gaussian approximation is given by

$$\begin{aligned} p\left(\mathbf{w}|\mathbf{x}\right) &\propto q\left(\mathbf{w}\right) \cdot p\left(\mathbf{w}\right) \\ &\propto \mathcal{G}\left(\mathbf{w}; \mathbf{V}^{-1}\mathbf{m}, \mathbf{V}^{-1}\right) \cdot \mathcal{G}\left(\mathbf{w}; \mathbf{\Sigma}^{-1}\boldsymbol{\mu}, \mathbf{\Sigma}^{-1}\right) \\ &= \mathcal{G}\left(\mathbf{w}; \mathbf{V}^{-1}\mathbf{m} + \mathbf{\Sigma}^{-1}\boldsymbol{\mu}, \mathbf{V}^{-1} + \mathbf{\Sigma}^{-1}\right) \\ &= \mathcal{N}\left(\mathbf{w}; \mathbf{\Sigma}\left(\mathbf{V} + \mathbf{\Sigma}\right)^{-1}\mathbf{m} + \mathbf{V}\left(\mathbf{V} + \mathbf{\Sigma}\right)^{-1}\boldsymbol{\mu}, \mathbf{V}\left(\mathbf{V} + \mathbf{\Sigma}\right)^{-1}\mathbf{\Sigma}\right), \end{aligned}$$

where we used the identity $\left(\mathbf{\Sigma}^{-1} + \mathbf{V}^{-1}\right)^{-1} = \mathbf{V}\left(\mathbf{V} + \mathbf{\Sigma}\right)^{-1}\mathbf{\Sigma} = \mathbf{\Sigma}\left(\mathbf{V} + \mathbf{\Sigma}\right)^{-1}\mathbf{V}$ in the last line. Similarly, if we use the likelihood approximation which has incorporated the translation invariance,

we get

$$\begin{aligned}
p_\beta(\mathbf{w}|\mathbf{x}) &\propto q_\beta(\mathbf{w}) \cdot p(\mathbf{w}) \\
&\propto \mathcal{G}\left(\mathbf{w}; \mathbf{V}_\beta^{-1}\mathbf{m}, \mathbf{V}_\beta^{-1}\right) \cdot \mathcal{G}\left(\mathbf{w}; \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}\right) \\
&= \mathcal{G}\left(\mathbf{w}; \mathbf{V}_\beta^{-1}\mathbf{m} + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}, \mathbf{V}_\beta^{-1} + \boldsymbol{\Sigma}^{-1}\right) \\
&= \mathcal{N}\left(\mathbf{w}; \left(\mathbf{V}_\beta^{-1} + \boldsymbol{\Sigma}^{-1}\right)^{-1}\left(\mathbf{V}_\beta^{-1}\mathbf{m} + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right), \left(\mathbf{V}_\beta^{-1} + \boldsymbol{\Sigma}^{-1}\right)^{-1}\right)
\end{aligned}$$

Observing that $\lim_{\beta\to\infty} \beta^{-2}\mathbf{I} = \mathbf{0}$ we thus see that

$$p_\infty(\mathbf{w}|\mathbf{x}) = \mathcal{N}\left(\mathbf{w}; \left(\mathbf{V}_\mathbf{B}^{-1} + \boldsymbol{\Sigma}^{-1}\right)^{-1}\left(\mathbf{V}_\mathbf{B}^{-1}\mathbf{m} + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right), \left(\mathbf{V}_\mathbf{B}^{-1} + \boldsymbol{\Sigma}^{-1}\right)^{-1}\right), \tag{66}$$

$$\mathbf{V}_\mathbf{B}^{-1} := \mathbf{V}^{-1} - \mathbf{V}^{-1}\mathbf{B}\left(\mathbf{B}^\mathrm{T}\mathbf{V}^{-1}\mathbf{B}\right)^{-1}\mathbf{B}^\mathrm{T}\mathbf{V}^{-1}. \tag{67}$$

Note again that $p_\infty(\mathbf{w}) = q_{\mathrm{mix}}(\mathbf{w}; \boldsymbol{\theta})$ as defined in (15).

D.2.1 Special Case of Diagonal Likelihood Covariance

Now let us consider the special case where the covariance matrix of the Gaussian likelihood approximation is a diagonal matrix, that is $\mathbf{V} = \mathrm{Diag}(\boldsymbol{\lambda})$. Then, observe that

$$\mathbf{V}^{-1}\mathbf{B} = \begin{bmatrix} \mathbf{V}_{K-1}^{-1} & \mathbf{0} \\ \mathbf{0} & \lambda_K^{-1} \end{bmatrix}\begin{bmatrix} \mathbf{I} \\ -x_K^{-1}\mathbf{x}_{K-1}^\mathrm{T} \end{bmatrix} = \begin{bmatrix} \mathbf{V}_{K-1}^{-1} \\ -\lambda_K^{-1}x_K^{-1}\mathbf{x}_{K-1}^\mathrm{T} \end{bmatrix}, \tag{68}$$

$$\mathbf{V}^{-1}\mathbf{B}\mathbf{V}_{K-1}\mathbf{x}_{K-1} = \left(\mathbf{V}^{-1}\mathbf{B}\right)\cdot\mathbf{V}_{K-1}\mathbf{x}_{K-1} = \begin{bmatrix} \mathbf{x}_{K-1} \\ -\lambda_K^{-1}x_K^{-1}\mathbf{x}_{K-1}^\mathrm{T}\mathbf{V}_{K-1}\mathbf{x}_{K-1} \end{bmatrix}, \tag{69}$$

$$\mathbf{B}^\mathrm{T}\mathbf{V}^{-1}\mathbf{B} = \begin{bmatrix} \mathbf{I} & -x_K^{-1}\mathbf{x}_{K-1} \end{bmatrix}\begin{bmatrix} \mathbf{V}_{K-1}^{-1} \\ -\lambda_K^{-1}x_K^{-1}\mathbf{x}_{K-1}^\mathrm{T} \end{bmatrix} = \mathbf{V}_{K-1}^{-1} + \lambda_K^{-1}x_K^{-2}\mathbf{x}_{K-1}\mathbf{x}_{K-1}^\mathrm{T},$$

where we used the notation $\mathbf{V}_{K-1}^{-1}$ to denote the diagonal $(K-1)\times(K-1)$ matrix with all

$$\boldsymbol{\lambda}_{K-1} := \left[\lambda_1^{-1}, \lambda_2^{-1}, \ldots, \lambda_{K-1}^{-1}\right]^\mathrm{T}$$

on the diagonal. Thus, using Theorem 2 with $\mathbf{C} = \mathbf{V}_{K-1}^{-1}$, $\mathbf{A} = x_K^{-2}\lambda_K^{-1}\mathbf{x}_{K-1}$ and $\mathbf{B} = \mathbf{x}_{K-1}^\mathrm{T}$, we see that

$$\begin{aligned}
\left(\mathbf{B}^\mathrm{T}\mathbf{V}^{-1}\mathbf{B}\right)^{-1} &= \mathbf{V}_{K-1} - \frac{1}{\lambda_K x_K^2 + \mathbf{x}_{K-1}^\mathrm{T}\mathbf{V}_{K-1}\mathbf{x}_{K-1}}\mathbf{V}_{K-1}\mathbf{x}_{K-1}\mathbf{x}_{K-1}^\mathrm{T}\mathbf{V}_{K-1} \\
&= \mathbf{V}_{K-1} - \frac{1}{\mathbf{x}^\mathrm{T}\mathbf{V}\mathbf{x}}\mathbf{V}_{K-1}\mathbf{x}_{K-1}\mathbf{x}_{K-1}^\mathrm{T}\mathbf{V}_{K-1},
\end{aligned}$$

Covariance $\left(\mathbf{V}_\mathbf{B}^{-1} + \boldsymbol{\Sigma}^{-1}\right)$. Thus, for the covariance (67) of the posterior we have

$$\begin{aligned}
&\mathbf{V}^{-1} - \mathbf{V}^{-1}\mathbf{B}\left(\mathbf{B}^\mathrm{T}\mathbf{V}^{-1}\mathbf{B}\right)^{-1}\mathbf{B}^\mathrm{T}\mathbf{V}^{-1} \\
&\quad= \mathbf{V}^{-1} - \mathbf{V}^{-1}\mathbf{B}\left[\mathbf{V}_{K-1} - \frac{1}{\mathbf{x}^\mathrm{T}\mathbf{V}\mathbf{x}}\mathbf{V}_{K-1}\mathbf{x}_{K-1}\mathbf{x}_{K-1}^\mathrm{T}\mathbf{V}_{K-1}\right]\mathbf{B}^\mathrm{T}\mathbf{V}^{-1} \\
&\quad= \mathbf{V}^{-1} - \underbrace{\mathbf{V}^{-1}\mathbf{B}\mathbf{V}_{K-1}\mathbf{B}^\mathrm{T}\mathbf{V}^{-1}}_{S} + \underbrace{\frac{1}{\mathbf{x}^\mathrm{T}\mathbf{V}\mathbf{x}}\mathbf{V}^{-1}\mathbf{B}\mathbf{V}_{K-1}\mathbf{x}_{K-1}\mathbf{x}_{K-1}^\mathrm{T}\mathbf{V}_{K-1}\mathbf{B}^\mathrm{T}\mathbf{V}^{-1}}_{T}.
\end{aligned}$$

Let's focus on the expression S first. Using (68) we have

$$\begin{aligned}
S &= \begin{bmatrix} \mathbf{V}_{K-1}^{-1} \\ -\lambda_K^{-1}x_K^{-1}\mathbf{x}_{K-1}^\mathrm{T} \end{bmatrix}\mathbf{V}_{K-1}\begin{bmatrix} \mathbf{V}_{K-1}^{-1} & -\lambda_K^{-1}x_K^{-1}\mathbf{x}_{K-1} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{V}_{K-1}^{-1} & -\lambda_K^{-1}x_K^{-1}\mathbf{x}_{K-1} \\ -\lambda_K^{-1}x_K^{-1}\mathbf{x}_{K-1}^\mathrm{T} & \lambda_K^{-2}x_K^{-2}\mathbf{x}_{K-1}^\mathrm{T}\mathbf{V}_{K-1}\mathbf{x}_{K-1} \end{bmatrix}.
\end{aligned} \tag{70}$$

Similarly, for the expression Tusing (69) we have

$$
\begin{aligned}
T &= \frac{1}{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}} \begin{bmatrix} \mathbf{x}_{K-1} \\ -\lambda_K^{-1} x_K^{-1} \mathbf{x}_{K-1}^{\mathrm{T}} \mathbf{V}_{K-1}\mathbf{x}_{K-1} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{K-1} & -\lambda_K^{-1} x_K^{-1} \mathbf{x}_{K-1}^{\mathrm{T}} \mathbf{V}_{K-1}\mathbf{x}_{K-1} \end{bmatrix} \\
&= \frac{1}{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}} \begin{bmatrix} \mathbf{x}_{K-1}\mathbf{x}_{K-1}^{\mathrm{T}} & -\frac{\mathbf{x}_{K-1}^{\mathrm{T}}\mathbf{V}_{K-1}\mathbf{x}_{K-1}}{\lambda_K x_K}\mathbf{x}_{K-1} \\ -\frac{\mathbf{x}_{K-1}^{\mathrm{T}}\mathbf{V}_{K-1}\mathbf{x}_{K-1}}{\lambda_K x_K}\mathbf{x}_{K-1}^{\mathrm{T}} & \frac{\left(\mathbf{x}_{K-1}^{\mathrm{T}}\mathbf{V}_{K-1}\mathbf{x}_{K-1}\right)^2}{\lambda_K^2 x_K^2} \end{bmatrix} . \qquad (71)
\end{aligned}
$$

Putting (70) and (71) together, we get

$$
\begin{aligned}
\mathbf{V}_{\mathbf{B}}^{-1} &= \frac{1}{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}} \begin{bmatrix} \mathbf{x}_{K-1}\mathbf{x}_{K-1}^{\mathrm{T}} & \left(\frac{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}}{\lambda_K x_K} - \frac{\mathbf{x}_{K-1}^{\mathrm{T}}\mathbf{V}_{K-1}\mathbf{x}_{K-1}}{\lambda_K x_K}\right)\mathbf{x}_{K-1} \\ \left(\frac{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}}{\lambda_K x_K} - \frac{\mathbf{x}_{K-1}^{\mathrm{T}}\mathbf{V}_{K-1}\mathbf{x}_{K-1}}{\lambda_K x_K}\right)\mathbf{x}_{K-1}^{\mathrm{T}} & \frac{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}}{\lambda_K} - \frac{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}\cdot\mathbf{x}_{K-1}^{\mathrm{T}}\mathbf{V}_{K-1}\mathbf{x}_{K-1}}{\lambda_K^2 x_K^2} + \frac{\left(\mathbf{x}_{K-1}^{\mathrm{T}}\mathbf{V}_{K-1}\mathbf{x}_{K-1}\right)^2}{\lambda_K^2 x_K^2} \end{bmatrix} \\
&= \frac{1}{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}} \begin{bmatrix} \mathbf{x}_{K-1}\mathbf{x}_{K-1}^{\mathrm{T}} & x_K\mathbf{x}_{K-1} \\ x_K\mathbf{x}_{K-1}^{\mathrm{T}} & x_K^2 \end{bmatrix} \\
&= \frac{1}{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}}\mathbf{x}\mathbf{x}^{\mathrm{T}} , \qquad (72)
\end{aligned}
$$

where we repeatedly used that $\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x} - \mathbf{x}_{K-1}^{\mathrm{T}}\mathbf{V}_{K-1}\mathbf{x}_{K-1} = \lambda_K x_K^2$. Thus, the covariance in (66) can be written as

$$
\begin{aligned}
\left(\mathbf{V}_{\mathbf{B}}^{-1} + \boldsymbol{\Sigma}^{-1}\right)^{-1} &= \left(\boldsymbol{\Sigma}^{-1} + \frac{1}{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}}\mathbf{x}\mathbf{x}^{\mathrm{T}}\right)^{-1} \\
&= \boldsymbol{\Sigma} - \frac{1}{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}} \cdot \frac{1}{1 + \left(\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}\right)^{-1}\mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{x}} \cdot \boldsymbol{\Sigma}\mathbf{x}\mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma} \\
&= \boldsymbol{\Sigma} - \frac{1}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}} \cdot \left(\boldsymbol{\Sigma}\mathbf{x}\right)\left(\boldsymbol{\Sigma}\mathbf{x}\right)^{\mathrm{T}} , \qquad (73)
\end{aligned}
$$

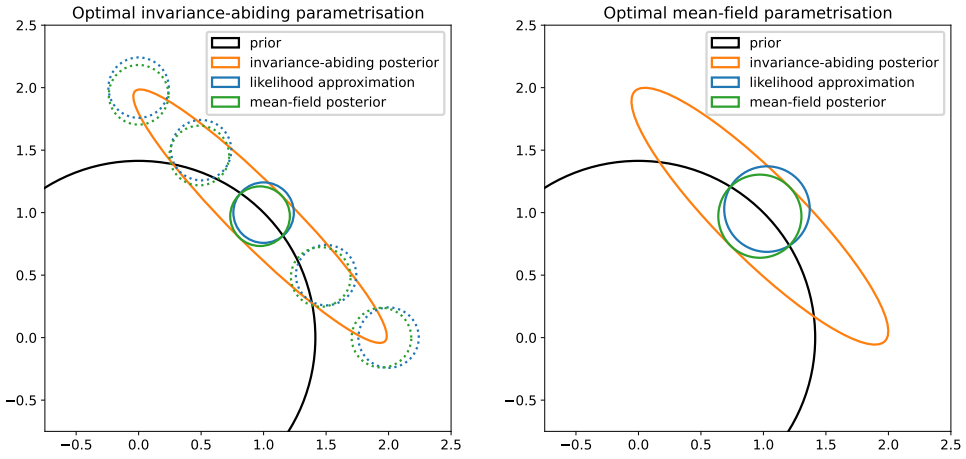
where we used Theorem 2 in the third step. Note that (14) is a special case of (72) when using $\mathbf{x} = \mathbf{1}$ and observing that $\mathbf{V}\mathbf{1} = \boldsymbol{\lambda}$. Similarly, the covariance in (15) is a special case of (73) when further noticing that $\boldsymbol{\Sigma}\mathbf{1} = \boldsymbol{\sigma}^2$.

Mean $\left(\mathbf{V}_{\mathbf{B}}^{-1} + \boldsymbol{\Sigma}^{-1}\right)^{-1}\left(\mathbf{V}_{\mathbf{B}}^{-1}\mathbf{m} + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right)$. In order to derive an efficient update for the mean of the posterior, please note that by virtue of (72), $\mathbf{V}_{\mathbf{B}}^{-1}$ can be written as $\mathbf{d}\mathbf{d}^{\mathrm{T}}$with $\mathbf{d} = \left(\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}\right)^{-\frac{1}{2}} \cdot \mathbf{x}$. Thus, using (73) we have

$$
\begin{aligned}
&\left(\mathbf{V}_{\mathbf{B}}^{-1} + \boldsymbol{\Sigma}^{-1}\right)^{-1}\left(\mathbf{V}_{\mathbf{B}}^{-1}\mathbf{m} + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right) \\
&= \left(\boldsymbol{\Sigma} - \frac{1}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}} \cdot \left(\boldsymbol{\Sigma}\mathbf{x}\right)\left(\boldsymbol{\Sigma}\mathbf{x}\right)^{\mathrm{T}}\right)\left(\mathbf{d}\mathbf{d}^{\mathrm{T}}\mathbf{m} + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right) \\
&= \boldsymbol{\Sigma}\mathbf{d}\mathbf{d}^{\mathrm{T}}\mathbf{m} + \boldsymbol{\mu} - \frac{1}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}} \cdot \left(\boldsymbol{\Sigma}\mathbf{x}\right)\left(\boldsymbol{\Sigma}\mathbf{x}\right)^{\mathrm{T}}\mathbf{d}\mathbf{d}^{\mathrm{T}}\mathbf{m} - \frac{1}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}} \cdot \left(\boldsymbol{\Sigma}\mathbf{x}\right)\left(\boldsymbol{\Sigma}\mathbf{x}\right)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \\
&= \boldsymbol{\mu} + \left(\frac{\mathbf{x}^{\mathrm{T}}\mathbf{m}}{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}}\right) \cdot \left(\boldsymbol{\Sigma}\mathbf{x}\right) - \frac{\left(\boldsymbol{\Sigma}\mathbf{x}\right)^{\mathrm{T}}\mathbf{d}\mathbf{d}^{\mathrm{T}}\mathbf{m}}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}} \cdot \left(\boldsymbol{\Sigma}\mathbf{x}\right) - \frac{\left(\boldsymbol{\Sigma}\mathbf{x}\right)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}} \cdot \left(\boldsymbol{\Sigma}\mathbf{x}\right) \\
&= \boldsymbol{\mu} + \left[\frac{\mathbf{x}^{\mathrm{T}}\mathbf{m}}{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}} - \frac{\mathbf{x}^{\mathrm{T}}\mathbf{m}}{\mathbf{x}^{\mathrm{T}}\mathbf{V}\mathbf{x}} \cdot \frac{\mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{x}}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}} - \frac{\mathbf{x}^{\mathrm{T}}\boldsymbol{\mu}}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}}\right] \cdot \left(\boldsymbol{\Sigma}\mathbf{x}\right) \\
&= \boldsymbol{\mu} + \left(\frac{\mathbf{x}^{\mathrm{T}}\left(\mathbf{m} - \boldsymbol{\mu}\right)}{\mathbf{x}^{\mathrm{T}}\left(\mathbf{V} + \boldsymbol{\Sigma}\right)\mathbf{x}}\right) \cdot \left(\boldsymbol{\Sigma}\mathbf{x}\right) .
\end{aligned}
$$

Thus, the location parameter in (15) is a special case of this more general result when using $\mathbf{x} = \mathbf{1}$ and noticing again that $\boldsymbol{\Sigma}\mathbf{1} = \boldsymbol{\sigma}^2$ and $\mathbf{V}\mathbf{1} = \boldsymbol{\lambda}$, respectively. It can be seen that the Gaussian product updates the prior location in the direction $\boldsymbol{\Sigma}\mathbf{x}$. This is because the likelihood is translation invariant wrt. all directions perpendicular to $\mathbf{x}$ (i.e. the hyper plane determined by the normal vector $\mathbf{x}$).

The two posterior approximations q_0 and q_{mix} as well as the corresponding likelihood approximation is visualised in Fig. 3 for two different parametrisations (cf. Sec. 4.3).



(a) Optimal parameters $\boldsymbol{\theta}^*_{\text{mix}}$, cf. (17a)

(b) Optimal parameters $\boldsymbol{\theta}^*_0$, cf. (17b)

Figure 3: Gaussian likelihood and posterior approximations q_0 and q_{mix} (see (15)) with different parameter optima (cf. (17)). The dotted circles show alternative parameter values that induce the same predictive distribution but do not correspond to one of the optima in (17).

D.3 True posterior and optimal invariance-abiding parameters

For the linear model with a single observation y and inputs $\mathbf{x}$, the true posterior $p(\mathbf{w} \mid \mathbf{x}, y)$ follows from the standard Bayesian update equation for Gaussian linear models (cf. (27) in App. B with $\mathbf{A} = \frac{1}{K} \cdot \mathbf{x}^\mathrm{T}$):

$$p(\mathbf{w} \mid y, \mathbf{x}) = \mathcal{N}\left(\mathbf{w}; \mathbf{m}_p^*, \mathbf{V}_p^*\right), \quad \mathbf{V}_p^* = \left(\frac{1}{K^2\sigma_y^2}\mathbf{x}\mathbf{x}^\mathrm{T} + \boldsymbol{\Sigma}^{-1}\right)^{-1}, \quad \mathbf{m}_p^* = \mathbf{V}_p^* \frac{y}{K\sigma_y^2}\mathbf{x}. \tag{74}$$

For N observations $Y := \{y^{(n)}\}_{n=1}^N$ with identical input $\mathbf{x}$, the posterior is

$$p\left(\mathbf{w} \mid Y, \mathbf{x}\right) = \mathcal{N}\left(\mathbf{w}; \mathbf{m}_p^*, \mathbf{V}_p^*\right), \mathbf{V}_p^* = \left(\frac{N}{K^2\sigma_y^2}\mathbf{x}\mathbf{x}^\mathrm{T} + \boldsymbol{\Sigma}^{-1}\right)^{-1}, \mathbf{m}_p^* = \mathbf{V}_p^* \frac{\sum_{n=1}^N y^{(n)}}{K\sigma_y^2}\mathbf{x}. \tag{75}$$

We want to relate the true posterior (75) to the parameters of the invariance-abiding posterior $q_{\text{mix}}(\mathbf{w}; \boldsymbol{\theta})$. It suffices to consider the special case $\mathbf{x} = \mathbf{1}$ from the main text. We will use the form

$$q_{\text{mix}}(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}; \left(\boldsymbol{\Sigma}^{-1} + \mathbf{V}_{\text{mix}}^{-1}\right)^{-1}\left(\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \mathbf{V}_{\text{mix}}^{-1}\mathbf{m}_{\text{mix}}\right), \boldsymbol{\Sigma}^{-1} + \mathbf{V}_{\text{mix}}^{-1}\right) \tag{76}$$

For the precision matrix $\boldsymbol{\Sigma}^{-1} + \mathbf{V}_{\text{mix}}^{-1}$, using the form in (72) with $\mathbf{x} = \mathbf{1}$ gives

$$\mathbf{V}_{\text{mix}}^{-1} = \frac{1}{\mathbf{x}^\mathrm{T}\mathbf{V}\mathbf{x}}\mathbf{x}\mathbf{x}^\mathrm{T} = \frac{1}{\mathbf{1}^\mathrm{T}\boldsymbol{\lambda}}\mathbf{1}\mathbf{1}^\mathrm{T}, \tag{77}$$

since $\mathbf{V}\mathbf{1} = \boldsymbol{\lambda}$. Comparing this to (75), we see that $\boldsymbol{\Sigma}^{-1} + \mathbf{V}_{\text{mix}}^{-1}$ has the same structure as the precision matrix of the true posterior. By setting the optimal variances as $\mathbf{1}^\mathrm{T}\boldsymbol{\lambda}^* = \frac{K^2\sigma_y^2}{N}$, and using $\frac{K^2\sigma_y^2}{N} = \frac{K\sigma_y^2}{N} \cdot \mathbf{1}^\mathrm{T}\mathbf{1}$, we see that one possible choice for the optimal variance parameters is

$$\boldsymbol{\lambda}^* = \frac{K\sigma_y^2}{N} \cdot \mathbf{1}. \tag{78}$$

We note that other vectors that are not proportional to $\mathbf{1}$ and also sum to $\frac{K^2\sigma_y^2}{N}$ are also valid optima.

Similarly, with $\boldsymbol{\mu} = \mathbf{0}$, and by noting that $\boldsymbol{\Sigma}^{-1} + \mathbf{V}_{\text{mix}}^{-1} = \mathbf{V}_p^{-1}$, we see that

$$\mathbf{m}^* = \frac{1}{N}\sum_{n=1}^N y^{(n)} \cdot \mathbf{1}. \tag{79}$$

E Permutation invariance in Bayesian neural networks

Here we analyse the *permutation* invariance in BNNs. This invariance is independent of the data, persisting for any dataset size.

We first describe the set of permutation matrices corresponding to the transformations that leave the likelihood invariant, i.e. $t(\mathbf{w}, \mathbf{r}) = \mathbf{P}_\mathbf{r}\mathbf{w}$. We ignore the biases for simplicity and describe these permutations first on a node/neuron level, then layer-wise for the weight matrices, and finally on the weight vector that is obtained by stacking all weight matrices. We will denote layers with indices l and the number of layers by L. Layer-wise permutation matrices are then denoted as $\tilde{\mathbf{P}}_l$ and the corresponding weight matrices are denoted as $\mathbf{W}_l$. The permutation matrix that results when stacking all weights matrices into a vector $\mathbf{w}$ is denoted as $\mathbf{P}$. When necessary, one particular permutation matrix from the set of all possible permutation matrices for a given architecture is indexed with superscript $^{(i)}$, i.e. $\mathcal{P} = \{\mathbf{P}^{(i)}\}_i$ and $|\mathcal{P}| = \prod_{l=1}^{L-1} k_l!$, where k_l is the number of nodes in layer l.

Node permutations. Each hidden layer $\mathbf{z}_l \in \mathbf{z}_1, \ldots, \mathbf{z}_{L-1}$ is a set of nodes that can be permuted in $k_l!$ possible ways, relabelling the corresponding parameters attached to these nodes. Each of these $k_l!$ permutations per layer can be combined with any of the permutations of another layer. Each permutation matrix $\tilde{\mathbf{P}}_l$ for a layer l corresponds to one of the unique orderings of nodes $\mathbf{z}_l$. For instance, the permutation matrix that reorders the first 3 nodes as $(z_{l,3}, z_{l,1}, z_{l,2})$ is

$$\tilde{\mathbf{P}}_l = \begin{pmatrix} 0 & 0 & 1 & \ldots \\ 1 & 0 & 0 & \ldots \\ 0 & 1 & 0 & \ldots \\ \ldots & \ldots & \ldots & \ldots \end{pmatrix}. \tag{80}$$

The remaining entries in the matrix are ones on the diagonal and zeros elsewhere.

Layer-wise permutation of weight matrices. Next, we describe how node permutations correspond to permutations of weight *matrices* in terms of permutations to the in- and outgoing weights. For one particular instance of permutations (omitting superscript $^{(i)}$) to each of the hidden layers, the corresponding weight matrices can be permuted as follows:

$$\forall l \in 1, \ldots, L: \quad \mathbf{w}'_l = \tilde{\mathbf{P}}_l \mathbf{W}_l \tilde{\mathbf{P}}_{l-1}^\mathsf{T}, \quad \tilde{\mathbf{P}}_0 = \tilde{\mathbf{P}}_L = \mathbf{I}. \tag{81}$$

The identities corresponding to the first and last layers is because only hidden layer nodes can be permuted but not the data itself. Note also that each permutation matrix is applied to two weight matrices, since permutations to nodes of a particular layer correspond to the simultaneous permutation of the weight matrices from the preceding and subsequent layer.

Permutation of stacked weight vectors. The layer-wise formulation can be written in terms of the stacked weight vector $\mathbf{w} = [\mathrm{vec}(\mathbf{W}_1), \ldots, \mathrm{vec}(\mathbf{W}_{L+1})]^\mathsf{T}$, using $\mathrm{vec}(ABC) = \left(C^\mathsf{T} \otimes A\right) \mathrm{vec}(B)$:

$$\mathrm{vec}(\mathbf{W}'_l) = \left(\tilde{\mathbf{P}}_{l-1} \otimes \tilde{\mathbf{P}}_l\right) \mathrm{vec}(\mathbf{W}_l) =: \mathbf{P}_{l,l-1} \mathrm{vec}(\mathbf{W}_l), \tag{82}$$

where we denote the new permutation matrix that is applied to the vectorised weights with a bar and the subscripts correspond to the two successive layers. The overall permutation matrix corresponding to the entire weight vector $\mathbf{w}$ is given by forming the block-diagonal matrix

$$\mathbf{P} = \begin{pmatrix} \mathbf{P}_{1,0} & \mathbf{0} & \mathbf{0} & \ldots \\ \mathbf{0} & \mathbf{P}_{2,1} & \mathbf{0} & \ldots \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_{3,2} & \ldots \\ \ldots & \ldots & \ldots & \ldots \end{pmatrix}, \tag{83}$$

where each denotes block-matrices of zeros with the respective dimensions and the remaining parts of the matrix are the identity on the diagonal and zeros elsewhere.

Invariance gap. Note again that the permutation invariance for BNNs is independent of the data. The invariance gap $\mathrm{KL}\left[q_0 \,||\, q_{\mathrm{mix}}\right]$ takes values in the range $[\,0,\, \ln|\mathcal{P}|\,]$, where $|\mathcal{P}|$ is the factorial number of modes. The gap takes the maximal value when each mode is completely separated from the other modes, and the gap is zero when both $q_0(\mathbf{w})$ and $q_{\mathrm{mix}}(\mathbf{w})$ are identical. This is the case e.g. if $q_0(\mathbf{w})$ reverts to the prior $p(\mathbf{w})$ since all modes are then identical, i.e. $q_0(\mathbf{P}\mathbf{w}) = q(\mathbf{w})$.

F Data-related bound on the mean-field KL divergence

Assume the regression setting described in Sec. 2.2, where we assume a finite dataset $\mathcal{D} = \{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^N$ and a regression setting with fixed homogeneous noise variance σ_y^2. We can further assume that the prior is chosen such that it induces a reasonably bounded output variance of the neural network, $\sigma_L^2\big(\mathbf{x}^{(n)}\big) := \mathbb{V}_{\mathbf{w}\sim p}\big[f(\mathbf{x}^{(n)};\, \mathbf{w})\big]$, for some fixed input $\mathbf{x}^{(n)}$. We will then have a finite expected log-likelihood and the ELBO for a mean-field variational approximation $q_{\boldsymbol{\theta}}$ is

$$\mathcal{L}_{\mathrm{ELBO}}\left(q_{\boldsymbol{\theta}}, \mathcal{D}\right) = \underbrace{\sum_{n=1}^{N} \mathbb{E}_{\mathbf{w}\sim q_{\boldsymbol{\theta}}}\left[\ln p(y^{(n)} \,|\, \mathbf{x}^{(n)}, w)\right]}_{\mathrm{ELL}(q_{\boldsymbol{\theta}}, \mathcal{D})} - \mathrm{KL}\left[q_{\boldsymbol{\theta}} \,||\, p\right]. \tag{84}$$

Let us now consider a hypothetical *worst-case* fit in terms of the ELBO objective. This is when the data is completely ignored with $q_{\boldsymbol{\theta}}(\mathbf{w}) = p(\mathbf{w})$, as any worse fit could be trivially improved by setting the approximation to the prior. This gives then the inequality (using (84))

$$\forall q_{\boldsymbol{\theta}}: \quad \mathcal{L}_{\mathrm{ELBO}}\left(q_{\boldsymbol{\theta}}, \mathcal{D}\right) \geq \mathcal{L}_{\mathrm{ELBO}}\left(p, \mathcal{D}\right) \;\Leftrightarrow\; \mathrm{ELL}\left(q_{\boldsymbol{\theta}}, \mathcal{D}\right) - \mathrm{KL}\left[q_{\boldsymbol{\theta}} \,||\, p\right] \geq \mathrm{ELL}\left(p, \mathcal{D}\right). \tag{85}$$

Although we can not compute the expected log-likelihood for an arbitrary fit $q_{\boldsymbol{\theta}}$, we can quantify the upper bound given above, by considering the *best-case* fit q^*, i.e. a hypothetical optimum where the data is perfectly predicted up to the known observation noise variance σ_y^2. The resulting bound is then

$$\mathrm{KL}\left[q_{\boldsymbol{\theta}} \,||\, p\right] \leq \mathrm{ELL}\left(q^*, \mathcal{D}\right) - \mathrm{ELL}\left(p, \mathcal{D}\right). \tag{86}$$

Next, we compute the two expected log-likelihood terms in (86). We first consider the worst-case, where we have

$$\mathrm{ELL}(p, \mathcal{D}) = \sum_{n=1}^{N} \mathbb{E}_{\mathbf{w}\sim p}\left[\ln p(y^{(n)}|\mathbf{x}^{(n)}, \mathbf{w})\right] \tag{87}$$

$$= \sum_{n=1}^{N} -\frac{1}{2\sigma_y^2} \mathbb{E}_{\mathbf{w}\sim p}\left[\left(y^{(n)} - z_L\right)^2\right] - \frac{N}{2}\ln\left[2\pi\sigma_y^2\right], \tag{88}$$

where $\mathbf{z}_L$ is the noisy output of the neural network given by

$$z_L = f(\mathbf{x}^{(n)};\, \mathbf{w}) + \epsilon, \quad \mathbf{w} \sim p(\mathbf{w}), \quad \epsilon \sim \mathcal{N}\left(0,\, \sigma_y^2\right). \tag{89}$$

Splitting the expectation of the quadratic term into variance and square of the expectation, we have

$$\mathbb{E}_{\mathbf{w}\sim p}\left[\left(y^{(n)} - z_L\right)^2\right] = \mathbb{E}_{\mathbf{w}\sim p}\left[y^{(n)} - z_L\right]^2 + \mathbb{V}_{\mathbf{w}\sim p}\left[y^{(n)} - z_L\right] \tag{90}$$

Since the last layer computes $f(\mathbf{x}^{(n)};\, \mathbf{w}) = \mathbf{w}_{L,1}^{\mathrm{T}} \mathbf{z}_{L-1}$ in the regression setting and the prior weights are independent of $\mathbf{z}_{L-1}$ and centred at zero, i.e. $\mathbb{E}_{\mathbf{w}\sim p}\left[\mathbf{w}_{L,1}\right] = \mathbf{0}$, it follows that

$$\mathbb{E}_{\mathbf{w}\sim p}\left[y^{(n)} - z_L\right] = y^{(n)}. \tag{91}$$

$$\mathbb{V}_{\mathbf{w}\sim p}\left[y^{(n)} - z_L\right] = \mathbb{E}_{\mathbf{w}\sim p}\left[z_L^2\right] = \sigma_L^2\big(\mathbf{x}^{(n)}\big) + \sigma_y^2, \tag{92}$$

where we denote the variance of the neural network outputs by $\sigma_L^2\big(\mathbf{x}^{(n)}\big) := \mathbb{V}_{\mathbf{w}\sim p}\big[f(\mathbf{x}^{(n)};\, \mathbf{w})\big]$.

Hence, the expected log likelihood under the prior is

$$\mathrm{ELL}(p, \mathcal{D}) = \sum_{n=1}^{N} -\frac{1}{2\sigma_y^2}\left[\sigma_L^2\big(\mathbf{x}^{(n)}\big) + \sigma_y^2 + \big(y^{(n)}\big)^2\right] - \frac{N}{2}\ln\left[2\pi\sigma_y^2\right] \tag{93}$$

$$= -\frac{1}{2}\sum_{n=1}^{N} \frac{\sigma_L^2\big(\mathbf{x}^{(n)}\big) + \big(y^{(n)}\big)^2}{\sigma_y^2} - \frac{N}{2}\left(1 + \ln\left[2\pi\sigma_y^2\right]\right). \tag{94}$$

For the best-case fit in terms of the ELBO, we assume that we completely overfit and perfectly predict the data by putting the mean of the Gaussian exactly on the data, i.e.

$$p(y^{(n)}|\mathbf{x}^{(n)}, \mathbf{w}) = \mathcal{N}(y; y^{(n)}, \sigma_y^2).$$

Then, we have

$$\mathbb{E}_{\mathbf{w}\sim q^*}\left[y^{(n)} - z_L\right] = 0, \tag{95}$$

$$\mathbb{V}_{\mathbf{w}\sim q^*}\left[y^{(n)} - z_L\right] = \sigma_y^2. \tag{96}$$

Hence, the expected log likelihood of the best case fit is

$$\mathrm{ELL}(q^*, \mathcal{D}) = \sum_{n=1}^{N} \mathbb{E}_{\mathbf{w}\sim q^*}\left[\ln p(y^{(n)} \mid \mathbf{x}^{(n)}, \mathbf{w})\right] \tag{97}$$

$$= \sum_{n=1}^{N} -\frac{1}{2\sigma_y^2}\left[\sigma_y^2\right] - \frac{N}{2}\ln\left[2\pi\sigma_y^2\right] \tag{98}$$

$$= -\frac{N}{2}\left(1 + \ln\left[2\pi\sigma_y^2\right]\right). \tag{99}$$

Taking the difference between (93) and (97), we obtain the result in (3),

$$\mathrm{KL}\left[q_{\boldsymbol{\theta}} \,||\, p\right] \leq \sum_{n=1}^{N} \frac{\sigma_L^2\left(\mathbf{x}^{(n)}\right) + \left(y^{(n)}\right)^2}{2\sigma_y^2}.$$

8 Multi-source Neural Variational Inference

When making inferences from multiple sources (modalities or views), a key challenge is to learn probabilistic representations that allow for comparing and combining information from different sources in a principled manner, while dealing with missing or anomalous data. This problem is addressed in this chapter by using a modular amortised variational inference approach (akin to variational autoencoders [26, 27]) and a generative model with conditionally independent observations: individual posterior approximations (beliefs)—each informed by a single source—are learned jointly. The resulting beliefs can be fused with and related to each other through a shared latent variable that represents the state underlying all observed data sources.

In contrast to the previous chapters, neural network weights are here treated as parameters rather than random variables. The relevant background is provided in Sec. 2.3 and2.4, and Ch. 4. Note that in the background sections (for consistency reasons), latent variables and observed variables are denoted by $\mathbf{x}$ and $\mathbf{y}$, respectively, whereas in the following publication, latent variables are denoted by $\mathbf{z}$ and observations by $\mathbf{x}$.

Authors	Richard Kurle Stephan Günnemann Patrick Van der Smagt	
Conference	Association for the Advancement of Artificial Intelligence , AAAI 2019	
Contribution	Problem definition	significantly contributed
	Literature survey	significantly contributed
	Algorithm development	significantly contributed
	Method implementation	significantly contributed
	Experimental evaluation	significantly contributed
	Preparation of the manuscript	significantly contributed

The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Multi-Source Neural Variational Inference

Richard Kurle
Department of Informatics
Technical University of Munich,
Data:Lab, Volkswagen Group
80805 Munich, Germany
richard.kurle@tum.de

Stephan Günnemann
Department of Informatics
Technical University of Munich
guennemann@in.tum.de

Patrick van der Smagt
Data:Lab, Volkswagen Group
80805 Munich, Germany

Abstract

Learning from multiple sources of information is an important problem in machine-learning research. The key challenges are learning representations and formulating inference methods that take into account the complementarity and redundancy of various information sources. In this paper we formulate a variational autoencoder based multi-source learning framework in which each encoder is conditioned on a different information source. This allows us to relate the sources via the shared latent variables by computing divergence measures between individual source's posterior approximations. We explore a variety of options to learn these encoders and to integrate the beliefs they compute into a consistent posterior approximation. We visualise learned beliefs on a toy dataset and evaluate our methods for learning shared representations and structured output prediction, showing trade-offs of learning separate encoders for each information source. Furthermore, we demonstrate how conflict detection and redundancy can increase robustness of inference in a multi-source setting.

1 Introduction

An essential feature of most living organisms is the ability to process, relate, and integrate information coming from a vast number of sensors and eventually from memories and predictions (Stein and Meredith 1993). While integrating information from complementary sources enables a coherent and unified description of the environment, redundant sources are beneficial for reducing uncertainty and ambiguity. Furthermore, when sources provide conflicting information, it can be inferred that some sources must be unreliable.

Replicating this feature is an important goal of multimodal machine learning (Baltrušaitis, Ahuja, and Morency 2017). Learning joint representations of multiple modalities has been attempted using various methods, including neural networks (Ngiam et al. 2011), probabilistic graphical models (Srivastava and Salakhutdinov 2014), and canonical correlation analysis (Andrew et al. 2013). These methods focus on learning joint representations and multimodal sensor fusion. However, it is challenging to relate information extracted from different modalities. In this work, we aim at learning probabilistic representations that can be related to each other by statistical divergence measures as well as translated from one modality to another. We make no assumptions about the nature of the data (i.e. multimodal or multi-view) and therefore adopt a more general problem formulation, namely learning from multiple *information sources*.

Probabilistic graphical models are a common choice to address the difficulties of learning from multiple sources by modelling relationships between information sources—i.e., observed random variables—via unobserved, random variables. Inferring the hidden variables is usually only tractable for simple linear models. For nonlinear models, one has to resort to approximate Bayesian methods. The variational autoencoder (VAE) (Kingma and Welling 2013; Rezende, Mohamed, and Wierstra 2014) is one such method, combining neural networks and variational inference for latent-variable models (LVM).

We build on the VAE framework, jointly learning the generative and inference models from multiple information sources. In contrast to the VAE, we encapsulate individual inference models into separate "modules". As a result, we obtain multiple posterior approximations, each informed by a different source. These posteriors represent the belief over the *same* latent variables of the LVM, conditioned on the available information in the respective source.

Modelling beliefs individually—but coupled by the generative model—enables computing meaningful quantities such as measures of surprise, redundancy, or conflict between beliefs. Exploiting these measures can in turn increase the robustness of the inference models. Furthermore, we explore different methods to integrate arbitrary subsets of these beliefs, to approximate the posterior for the respective subset of observations. We essentially modularise neural variational inference in the sense that information sources and their associated encoders can be flexibly interchanged and combined after training.

2 Background—Neural variational inference

Consider a dataset $\mathbf{X} = \{\mathbf{x}^{(n)}\}_{n=1}^N$ of N i.i.d. samples of some random variable $\mathbf{x}$ and the following generative model:

$$p_\theta(\mathbf{x}^{(n)}) = \int p_\theta(\mathbf{x}^{(n)} \mid \mathbf{z}^{(n)})\, p(\mathbf{z}^{(n)})\, d\mathbf{z}^{(n)},$$

where θ are the parameters of a neural network, defining the conditional distribution between latent and observable random variables $\mathbf{z}$ and $\mathbf{x}$ respectively. The variational autoen-

coder (Kingma and Welling 2013; Rezende, Mohamed, and Wierstra 2014) is an approximate inference method that enables learning the parameters of this model by optimising an evidence lower bound (ELBO) to the log marginal likelihood. A second neural network with parameters ϕ defines the parameters of an approximation $q_\phi(\mathbf{z} \mid \mathbf{x})$ of the posterior distribution. Since the computational cost of inference for each data point is shared by using a recognition model, some authors refer to this form of inference as amortised or neural variational inference (Gershman and Goodman 2014; Mnih and Gregor 2014).

The importance weighted autoencoder (Burda, Grosse, and Salakhutdinov 2015) (IWAE) generalises the VAE by using a multi-sample importance weighting estimate of the log-likelihood. The IWAE ELBO is given as:

$$\ln p_\theta(\mathbf{x}^{(n)}) \geq \mathbb{E}_{\mathbf{z}_{1:K}^{(n)} \sim q_\phi(\mathbf{z}^{(n)} \mid \mathbf{x}^{(n)})} \Big[\ln \frac{1}{K} \sum_{k=1}^{K} w_k^{(n)} \Big],$$

where K is the number of importance samples, and $w_k^{(n)}$ are the importance weights:

$$w_k^{(n)} = \frac{p_\theta(\mathbf{x}^{(n)} \mid \mathbf{z}_k^{(n)})\, p(\mathbf{z}_k^{(n)})}{q_\phi(\mathbf{z}_k^{(n)} \mid \mathbf{x}^{(n)})}.$$

Besides achieving a tighter lower bound, the IWAE was motivated by noticing that a multi-sample estimate does not require all samples from the variational distribution to have a high posterior probability. This enables the training of a generative model using samples from a variational distribution with higher uncertainty. Importantly, this distribution need not be the posterior of all observations in the generative model. It can be a good enough proposal distribution, i.e. the belief from a partially-informed source.

3 Multi-source neural variational inference

We are interested in datasets consisting of tuples $\{\mathbf{x}^{(n)} = (\mathbf{x}_1^{(n)}, \ldots, \mathbf{x}_M^{(n)})\}_{n=1}^N$, we use $m \in \{1, \ldots, M\}$ to denote the index of the source. Each observation $\mathbf{x}_m^{(n)} \in \mathbb{R}^{D_m}$ may be embedded in a different space but is assumed to be generated from the same latent state $\mathbf{z}^{(n)}$. Therefore, each $\mathbf{x}_m^{(n)}$ corresponds to a different, potentially limited source of information about the underlying state $\mathbf{z}^{(n)}$. From now on we will refer to $\mathbf{x}_m$ in the generative model as observations and the same $\mathbf{x}_m$ in the inference model as information sources.

We model each observation $\mathbf{x}_m$ in the generative model with a distinct set of parameters θ_m, although some parameters could be shared. The likelihood function is given as:

$$p_\theta(\mathbf{x}^{(n)} \mid \mathbf{z}^{(n)}) = \prod_{m=1}^{M} p_{\theta_m}\big(\mathbf{x}_m^{(n)} \mid \mathbf{z}^{(n)}\big).$$

For inference, the VAE conditions on all observable data $\mathbf{x}^{(n)}$. However, one can condition (amortize) the approximate posterior distribution on any set of information sources. In this paper we limit ourselves to $\mathbf{x}_S^{(n)},\ S \subset \{1, \ldots, M\}$. An approximate posterior distribution $q_{\phi_S}(\mathbf{z}^{(n)} \mid \mathbf{x}_S^{(n)})$ may then be interpreted as the belief of the respective information sources about the latent variables, underlying the generative process.

In contrast to the VAE, we want to calculate the beliefs from *different* information sources individually, compare them, and eventually integrate them. In the following, we address each of these desiderata.

3.1 Learning individual beliefs

In order to learn individual inference models as in Fig. 1a, we propose an average of M ELBOs, one for each information source and its respective inference model. The resulting objective is an ELBO to the log marginal likelihood itself and referred to as $\mathcal{L}^{(\mathrm{ind})}$:

$$\mathcal{L}^{(\mathrm{ind})} =: \sum_{m=1}^{M} \pi_m \mathbb{E}_{\mathbf{z}_{1:K}^{(n)} \sim q_{\phi_m}\left(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)}\right)} \Big[\ln \frac{1}{K} \sum_{k=1}^{K} w_{m,k}^{(n)} \Big], \tag{1}$$

with

$$w_{m,k}^{(n)} = \frac{p_\theta\big(\mathbf{x}^{(n)} \mid \mathbf{z}_k^{(n)}\big) p\big(\mathbf{z}_k^{(n)}\big)}{q_{\phi_m}\big(\mathbf{z}_k^{(n)} \mid \mathbf{x}_m^{(n)}\big)}.$$

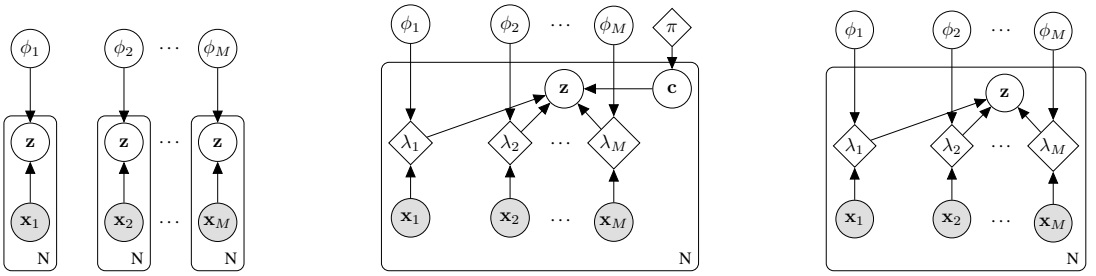
The indices n, m and k refer to the data sample, information source, and importance sample index. The factors π_m are the weights of the ELBOs, satisfying $0 \leq \pi_m \leq 1$ and $\sum_{m=1}^{M} \pi_m = 1$. Although the π_m could be inferred, we set $\pi_m = 1/M,\ \forall m$. This ensures that all parameters ϕ_m are optimised individually to their best possible extent instead of down-weighting less informative sources.

Since we are dealing with partially-informed encoders $q_{\phi_m}(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)})$ instead of $q_\phi(\mathbf{z}^{(n)} \mid \mathbf{x}^{(n)})$, the beliefs can be more uncertain than the posterior of all observations $\mathbf{x}$. This in turn degrades the generative model, as it requires samples from the posterior distribution. We found that the generative model becomes biased towards generating averaged samples rather than samples from a diverse, multimodal distribution. This issue arises in VAE-based objectives, irrespective of the complexity of the variational family, because each Monte-Carlo sample of latent variables must predict all observations. To account for this, we propose to use importance sampling estimates of the log-likelihood (see Sec. 2). The importance weighting and sampling-importance-resampling can be seen as feedback from the observations, allowing to approximate the true posterior even with poorly informed beliefs.

3.2 Comparing beliefs

Encapsulating individual inferences has an appealing advantage compared to an uninterpretable, deterministic combination within a neural network: Having obtained multiple beliefs w.r.t. the same latent variables, each informed by a distinct source, we can calculate meaningful quantities to relate the sources. Examples are measures of redundancy, surprise, or conflict. Here we focus on the latter.

Detecting conflict between beliefs is crucial to avoid false inferences and thus increase robustness of the model. Conflicting beliefs may stem from conflicting data or from unreliable (inference) models. The former is a form of data anomaly, e.g. due to a failing sensor. An unreliable model



(a) Individual inferences (b) Mixture of experts inference. (c) Product of experts inference

Figure 1: Graphical models of inference models. White circles denote hidden random variables, grey-shaded circles—observed random variables, diamonds—deterministic variables. N is the number of i.i.d. samples in the dataset. To better distinguish the mixture or product of expert models from an IWAE with hard-wired integration in a neural-network layer, we explicitly draw the deterministic variables $\lambda_1, \ldots, \lambda_M$, denoting the parameters of the variational distributions.

on the other hand may result from model misspecification or optimisation problems, i.e. due to the approximation or amortisation gap, respectively (Cremer, Li, and Duvenaud 2018). Distinguishing between the two causes of conflict is challenging however and requires evaluating the observed data under the likelihood functions.

Previous work has used the ratio of two KL divergences as a criterion to detect a conflict between a subjective prior and the data (Bousquet 2008). The nominator is the KL between the posterior and the subjective prior, and denominator is the KL between posterior and a non-informative reference prior. The two KL divergences measure the information gain of the posterior—induced by the evidence—w.r.t. the subjective prior and the non-informative prior, respectively. The decision criterion for conflict is a ratio greater than 1.

We propose a similar ratio, replacing the subjective prior with q_{ϕ_m} and taking the prior as reference:

$$\mathrm{c}(m \mid\mid m') = \frac{D_{\mathrm{KL}}\big(q_{\phi_{m'}}(\mathbf{z} \mid \mathbf{x}_{m'}) \mid\mid q_{\phi_m}(\mathbf{z} \mid \mathbf{x}_m)\big)}{D_{\mathrm{KL}}\big(q_{\phi_{m'}}(\mathbf{z} \mid \mathbf{x}_{m'}) \mid\mid p(\mathbf{z})\big)}. \quad (2)$$

This measure has the property that it yields high values if the belief of source m is significantly more certain than that of m'. This is desirable for sources with redundant information. For complementary information sources other conflict measures, e.g. the measure defined in (Dahl, Gåsemyr, and Navig), may be more appropriate.

3.3 Integrating beliefs

So far, we have shown how to learn separate beliefs from different sources and how to relate them. However, we have not readily integrated the information from these sources. This can be seen by noticing that the gap between $\mathcal{L}^{(\mathrm{ind})}$ and the log marginal likelihood is significantly larger compared to an IWAE with an unflexible, hard-wired combination (see supplementary material of our accompanying technical report (Kurle, Günnemann, and Smagt 2018)). Here we propose two methods to integrate the beliefs $q_{\phi_m}(\mathbf{z} \mid x_m)$ to an integrated belief $q_\phi(\mathbf{z} \mid \mathbf{x})$.

Disjunctive integration—Mixture of Experts One approach to combine individual beliefs is by treating them as alternatives, which is justified if some (but not all) sources or their respective models are unreliable or in conflict (Khaleghi et al. 2013). We propose a mixture of experts (MoE) distribution, where each component is the belief, informed by a different source. The corresponding graphical model for inference is shown in Fig. 1b. As in Sec. 3.1, the variational parameters are each predicted from one source individually without communication between them. The difference is that each $q_{\phi_m}(\mathbf{z} \mid \mathbf{x}_m)$ is considered as a mixture component, such that the whole mixture distribution approximates the true posterior.

Instead of learning individual beliefs $q_{\phi_m}(\mathbf{z} \mid \mathbf{x}_m)$ by optimising $\mathcal{L}^{(\mathrm{ind})}$ and integrating them subsequently into a combined $q_\phi(\mathbf{z} \mid \mathbf{x})$, we can design an objective function for learning the MoE posterior directly. We refer to the corresponding ELBO as $\mathcal{L}^{(\mathrm{MoE})}$. It differs from $\mathcal{L}^{(\mathrm{ind})}$ only by the denominator of the importance weights, using the mixture distribution with component weights π_m:

$$w_{m,k}^{(n)} = \frac{p_\theta\big(\mathbf{x}^{(n)} \mid \mathbf{z}_k^{(n)}\big)\, p\big(\mathbf{z}_k^{(n)}\big)}{\sum_{m'=1}^{M} \pi_{m'} q_{\phi_{m'}}\big(\mathbf{z}_k^{(n)} \mid \mathbf{x}_{m'}^{(n)}\big)},$$

Conjunctive integration—Product of Experts Another option for combining beliefs are conjunctive methods, treating each belief as a constraint. These are applicable in the case of equally reliable and independent evidences (Khaleghi et al. 2013). This can be seen by inspecting the mathematical form of the posterior distribution of all observations. Applying Bayes' rule twice reveals that the true posterior of a graphical model with conditionally independent observations can be decomposed as a product of experts (Hinton 2002) (PoE):

$$p(\mathbf{z} \mid \mathbf{x}) = \frac{\prod_{m'=1}^{M} p(\mathbf{x}_{m'})}{p(\mathbf{x})} \cdot p(\mathbf{z}) \cdot \prod_{m=1}^{M} \frac{p(\mathbf{z} \mid \mathbf{x}_m)}{p(\mathbf{z})}. \quad (3)$$

We propose to approximate Eq. (3) by replacing the true posteriors of single observations $p(\mathbf{z} \mid \mathbf{x}_m)$ by the variational distributions $q_{\phi_m}(\mathbf{z} \mid \mathbf{x}_m)$, obtaining the inference model shown in Fig. 1c. In order to make the PoE distribution computable, we further assume that the variational distributions

and the prior are conjugate distributions in the exponential family. Probability distributions in the exponential family have the well-known property that their product is also in the exponential family. Hence, we can calculate the normalisation constant in Eq. (3) from the natural parameters. In this work, we focus on the popular case of normal distributions. For the derivation of the natural parameters and normalisation constant, we refer to the supplementary material of our technical report (Kurle, Günnemann, and Smagt 2018).

Analogous to Sec. 3.3, we can design an objective to learn the PoE distribution directly, rather than integrating individual beliefs. We refer to the corresponding ELBO as $\mathcal{L}^{(\mathrm{PoE})}$:

$$\mathcal{L}^{(\mathrm{PoE})} =: \mathbb{E}_{\mathbf{z}_{1:K}^{(n)} \sim q_\phi\left(\mathbf{z}^{(n)} \mid \mathbf{x}^{(n)}\right)} \left[\ln \frac{1}{K} \sum_{k=1}^{K} w_k^{(n)} \right], \tag{4}$$

where $w_k^{(n)}$ are the standard importance weights as in the IWAE and where $q_\phi(\mathbf{z}^{(n)} \mid \mathbf{x}^{(n)})$ is the PoE inference distribution. However, the natural parameters of the individual normal distributions are not uniquely identifiable by the natural parameters of the integrated normal distribution. Thus, optimising $\mathcal{L}^{(\mathrm{PoE})}$ leads to inseparable individual beliefs. To account for this, we propose a hybrid between individual and integrated inference distribution:

$$\mathcal{L}^{(\mathrm{hybrid})} = \lambda_1 \mathcal{L}^{(\mathrm{ind})} + \lambda_2 \mathcal{L}^{(\mathrm{PoE})}, \tag{5}$$

where we choose $\lambda_1 = \lambda_2 = \frac{1}{2}$ in practice for simplicity.

In Sec. 5 we evaluate the proposed integration methods both as learning objectives, and for integrating the beliefs obtained by optimising $\mathcal{L}^{(\mathrm{ind})}$ or $\mathcal{L}^{(\mathrm{hybrid})}$. Note again however, that $\mathcal{L}^{(\mathrm{PoE})}$ or $\mathcal{L}^{(\mathrm{hybrid})}$ assume conditionally independent observations and equally reliable sources. In contrast, $\mathcal{L}^{(\mathrm{ind})}$ makes no assumptions about the structure of the generative model. This allows for any choice of appropriate integration method after learning.

4 Related Work

Canonical correlation analysis (CCA) (Hotelling 1936) is an early attempt to examine the relationship between two sets of variables. CCA and nonlinear variants (Shon et al. 2005; Andrew et al. 2013; Feng, Li, and Wang 2015) propose projections of pairs of features such that the transformed representations are maximally correlated. CCA variants have been widely used for learning from multiple information sources (Hardoon, Szedmak, and Shawe-taylor 2004; Rasiwasia et al. 2010). These methods have in common with ours, that they learn a common representational space for multimodal data. Furthermore, a connection between linear CCA and probabilistic graphical models has been shown (Bach and Jordan 2005).

Dempster-Shafer theory (Dempster 1967; Shafer 1976) is a widely used framework for integration of uncertain information. Similar to our PoE integration method, Dempster's rule of combination takes the pointwise product of belief functions and normalises subsequently. Due to apparently counterintuitive results obtained when dealing with conflicting information (Zadeh 1986), the research community proposed various measures to detect conflicting belief functions and proposed alternative integration methods. These include disjunctive integration methods (Jiang et al. 2016; Denœux 2008; Deng 2015; Murphy 2000), similar to our MoE integration method.

A closely related line of research is that of multimodal autoencoders (Ngiam et al. 2011) and multimodal Deep Boltzmann machines (DBM) (Srivastava and Salakhutdinov 2014). Multimodal autoencoders use a shared representation for input and reconstructions of different modalities. Since multimodal autoencoders learn only deterministic functions, the interpretability of the representations is limited. Multimodal DBMs on the other hand learn multimodal generative models with a joint representation between the modalities. However, DBMs have only been shown to work on binary latent variables and are notoriously hard to train.

More recently, variational autoencoders were applied to multimodal learning (Suzuki, Nakayama, and Matsuo 2016). Their objective function maximises the ELBO using an encoder with hard-wired sources and additional KL divergence loss terms to train individual encoders. The difference to our methods is that we maximise an ELBO for which we require only M individual encoders. We may then integrate the beliefs of arbitrary subsets of information sources after training. In contrast, the method in (Suzuki, Nakayama, and Matsuo 2016) would require a separate encoder for each possible combination of sources. Similarly, (Vedantam et al. 2017) first trains a generative model with multiple observations, using a fully-informed encoder. In a second training stage, they freeze the generative model parameters and proceed by optimising the parameters of inference models which are informed by a single source. Since the topology of the latent space is fixed in the second stage, finding good weights for the inferenc models may be complicated.

Concurrently to this work, (Wu and Goodman 2018) proposed a method for weakly-supervised learning from multimodal data, which is very similar to our hybrid method discussed in Sec. 3.3. Their method is based on the VAE, whereas we find it crucial to optimise the importance-sampling based ELBO to prevent the generative models from generating averaged conditional samples (see Sec. 3.1).

5 Experiments

We visualise learned beliefs on a 2D toy problem, evaluate our methods for structured prediction and demonstrate how our framework can increase robustness of inference. Model and algorithm hyperparameters are summarised in the supplementary material of our technical report (Kurle, Günnemann, and Smagt 2018).

5.1 Learning beliefs from complementary information sources

We begin our experiments with a toy dataset with complementary sources. As a generative process, we consider a mixture of bi-variate normal distributions with 8 mixture components. The means of each mixture component are located on the unit circle with equidistant angles, and the standard deviations are 0.1. To simulate complementary sources, we allow each source to perceive only one dimension of the data. As with all our experiments, we as-

sume a zero-centred normal prior with unit variance and $\mathbf{z} \in \mathbb{R}^2$. We optimise $\mathcal{L}^{(\text{ind})}$ with two inference models $q_{\phi_1}(\mathbf{z} \mid \mathbf{x}_1)$, $q_{\phi_2}(\mathbf{z} \mid \mathbf{x}_2)$, and two separate likelihood functions $p_{\theta_1}(\mathbf{x}_1 \mid \mathbf{z})$, $p_{\theta_2}(\mathbf{x}_2 \mid \mathbf{z})$. Fig. 2a (right) shows the beliefs of both information sources for 8 test data points. These test points are the means of the 8 mixture components of the observable data, rotated by 2°. The small rotation is only for visualisation purposes, since each source is allowed to perceive only one axis and would therefore produce indistinguishable beliefs for data points with identical values on the perceived axis. We visualise the two beliefs corresponding to the same data point with identical colours. The height and width of the ellipses correspond to the standard deviations of the beliefs. Fig. 2a (left) shows random samples in the observation space, generated from 10 random latent samples $\mathbf{z} \sim q_{\phi_m}(\mathbf{z} \mid \mathbf{x}_m)$ for each belief. The generated samples are colour-coded in correspondence to the figure on the right. The 8 circles in the background visualise the true data distribution with 1 and 2 standard deviations. The two types of markers distinguish the information sources $\mathbf{x}_1$ and $\mathbf{x}_2$ used for inference. As can be seen, the beliefs reflect the ambiguity as a result of perceiving a single dimension x_m. [1]

Next we integrate the two beliefs using Eq. (3). The resulting integrated belief and generated data from random latent samples of the belief are shown in Figs. 2b (right) and 2b (left) respectively. We can see that the integration resolves the ambiguity. In the supplementary material of our accompanying technical report (Kurle, Günnemann, and Smagt 2018), we plot samples from the individual and integrated beliefs, before and after a sampling importance resampling procedure.

5.2 Learning and inference of shared representations for structured prediction

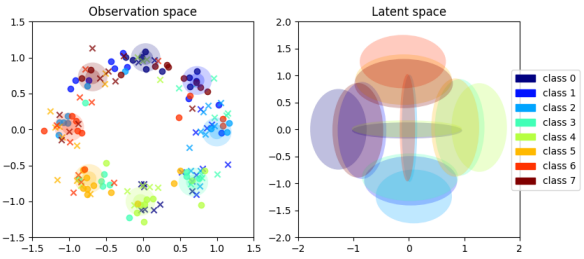
Models trained with $\mathcal{L}^{(\text{ind})}$ or $\mathcal{L}^{(\text{hybrid})}$ can be used to predict structured data of any modality, conditioned on any available information source. Equivalently, we may impute missing data if modelled explicitly as an information source:

$$p(\mathbf{x}_m \mid \mathbf{x}_{m'}) = \mathbb{E}_{\mathbf{z} \sim q_{\phi_{m'}}(\mathbf{z} \mid \mathbf{x}_{m'})} \Big[p_{\theta_m}(\mathbf{x}_m \mid \mathbf{z}) \Big]. \quad (6)$$

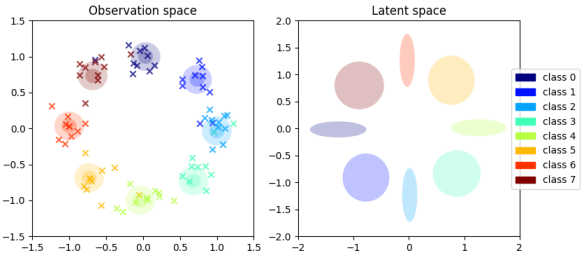
MNIST variants We created 3 variants of MNIST (Lecun et al. 1998), where we simulate multiple information sources as follows:

- MNIST-TB: $\mathbf{x}_1$ perceives the **top** half and $\mathbf{x}_2$ perceives the **bottom** half of the image.
- MNIST-QU: 4 information sources that each perceive **quarters** of the image.
- MNIST-NO: 4 information sources with independent bit-flip **noise** with $p = 0.05$. We use these 4 sources to amortise inference. In the generative model, we use the standard, noise-free digits as observable variables.

[1] The true posterior (of a single source) has two modes for most data points. The uni-modal (Gaussian) proposal distribution learns to cover both modes.



(a) Individual beliefs and their predictions. **Left:** 8 coloured circles are centred at the 8 test inputs from a mixture of Gaussians toy dataset. The radii indicate 1 and 2 standard deviations of the normal distributions. The two types of markers represent generated data from random samples of one of the information sources (data axis 0 or 1). **Right:** Corresponding individual beliefs. Ellipses show 1 standard deviation of the individual approximate posterior distributions.



(b) Integrated belief and its predictions.

Figure 2: Approximate posterior distributions and samples from the predicted likelihood function with and without integration of beliefs

First, we assess how well individual beliefs can be integrated after learning, and whether beliefs can be used individually when learning them as integrated inference distributions. On all MNIST variants, we train 5 different models by optimising the objectives $\mathcal{L}^{(\text{ind})}$, $\mathcal{L}^{(\text{MoE})}$, $\mathcal{L}^{(\text{PoE})}$, and $\mathcal{L}^{(\text{hybrid})}$ with $K = 16$, as well as $\mathcal{L}^{(\text{hybrid})}$ with $K = 1$. All other hyperparameters are identical. We then evaluate each model under the 3 objectives $\mathcal{L}^{(\text{ind})}$, $\mathcal{L}^{(\text{MoE})}$ and $\mathcal{L}^{(\text{PoE})}$. For comparison, we also train a standard IWAE with hard-wired sources on MNIST and on MNIST-NO with a single noisy source. The ELBOs on the test set are estimated using $K = 16$ importance samples. The obtained estimates are summarised in Tab. 1. The results confirm that learning the PoE inference model directly leads to inseparable individual beliefs. As expected, learning individual inference models and integrating them subsequently as a PoE comes with a tradeoff for $\mathcal{L}^{(\text{PoE})}$, which is mostly due to the low entropy of the integrated distribution. On the other hand, optimising the model with $\mathcal{L}^{(\text{hybrid})}$ achieves good results for both individual and integrated beliefs. On MNIST-NO, we can get an improvement of 2.74 nats by integrating the beliefs of redundant sources, compared to the standard IWAE with a single source.

Next, we evaluate our method for conditional (structured)

Table 1: Negative evidence lower bounds on variants of randomly binarised MNIST. Lower is better.

MNIST-TB

	$\mathcal{L}^{(\text{ind})}$	$\mathcal{L}^{(\text{MoE})}$	$\mathcal{L}^{(\text{PoE})}$	$\mathcal{L}^{(\text{hybrid})}$	$\mathcal{L}^{(\text{hybrid})}_{(K=1)}$	IWAE
$\mathcal{L}^{(\text{ind})}$	**102.20**	102.40	265.59	104.03	108.97	-
$\mathcal{L}^{(\text{MoE})}$	**101.51**	101.82	264.48	103.37	108.30	-
$\mathcal{L}^{(\text{PoE})}$	94.38	94.39	**87.59**	90.07	90.81	88.79

MNIST-QU

	$\mathcal{L}^{(\text{ind})}$	$\mathcal{L}^{(\text{MoE})}$	$\mathcal{L}^{(\text{PoE})}$	$\mathcal{L}^{(\text{hybrid})}$	$\mathcal{L}^{(\text{hybrid})}_{(K=1)}$	IWAE
$\mathcal{L}^{(\text{ind})}$	120.46	**120.37**	447.67	129.63	140.61	-
$\mathcal{L}^{(\text{MoE})}$	**119.10**	119.98	446.02	128.16	139.19	-
$\mathcal{L}^{(\text{PoE})}$	108.07	107.85	**87.67**	89.20	90.17	88.79

MNIST-NO

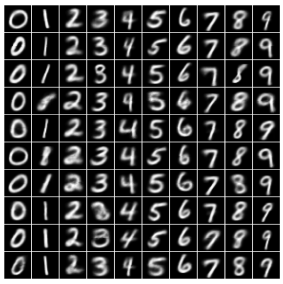
	$\mathcal{L}^{(\text{ind})}$	$\mathcal{L}^{(\text{MoE})}$	$\mathcal{L}^{(\text{PoE})}$	$\mathcal{L}^{(\text{hybrid})}$	$\mathcal{L}^{(\text{hybrid})}_{(K=1)}$	IWAE
$\mathcal{L}^{(\text{ind})}$	**94.81**	94.86	101.20	96.27	95.31	-
$\mathcal{L}^{(\text{MoE})}$	**93.98**	94.03	100.36	95.58	94.55	-
$\mathcal{L}^{(\text{PoE})}$	94.52	94.65	92.27	**92.21**	94.49	94.95

prediction using Eq. (6). Fig. 3a shows the means of the likelihood functions, with latent variables drawn from individual and integrated beliefs. To demonstrate conditional image generation from labels, we add a third encoder that perceives class labels. Fig. 3b shows the means of the likelihood functions, inferred from labels.

We also compare our method to the missing data imputation procedure described in (Rezende, Mohamed, and Wierstra 2014) for MNIST-TB und MNIST-QU. We run the Markov chain for all samples in the test set for 150 steps each and calculate the log likelihood of the imputed data at every step. The results—averaged over the dataset—are compared to our multimodal data generation method in Fig. 4. For large portions of missing data as in MNIST-TB, the Markov chain often fails to converge to the marginal distribution. But even for MNIST-QU with only a quarter of the image missing, our method outperforms the Markov chain procedure by a large margin. Please consult the supplementary material for a visualisation of the stepwise generations during the inference procedure.

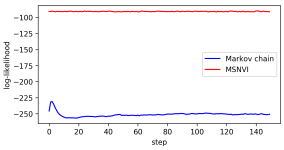
Caltech-UCSD Birds 200 Caltech-UCSD Birds 200 (Welinder et al. 2010) is a dataset with 6033 images of birds with 128×128 resolutions, split into 3000 train and 3033 test images. As a second source, we use segmentation masks provided by (Yang, Safar, and Yang 2014). On this dataset we assess whether learning with multiple modalities can be advantageous in scenarios where we are interested only in one particular modality. Therefore, we evaluate the ELBO for a single source and a single target observation, i.e. encoding images and decoding segmentation masks. We compare models that learned with multiple modalities using $\mathcal{L}^{(\text{ind})}$ and $\mathcal{L}^{(\text{hybrid})}$ with models that learnt from a single modality. Additionally, we evaluate the segmentation accuracy using Eq. (6). The accuracy is estimated with 100 samples, drawn from the belief informed by image data. The results are summarised in Tab. 2. We distinguish between objectives that involve both modalities in the generative model and objectives where we learn only the generative model for the modality

(a) **Row 1:** Original images. **Row 2–4:** Belief informed by top half of the image. **Row 5–7:** Informed by bottom half. **Row 8–10:** Integrated belief.

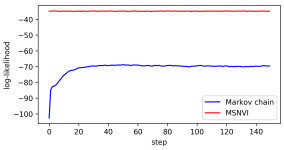


(b) Predictions from 10 random samples of the latent variables, inferred from one-hot class labels.

Figure 3: Predicted images, where latent variables are inferred from the variational distributions of different sources. Sources with partial information generate diverse samples, the integration resolves ambiguities. E.g. in Fig. 3a, the lower half of digit 3 randomly generates digits 5 and 3 and the upper half generates digits 3 and 9. In contrast, the integration resolves ambiguities.



(a) MNIST-TB, where bottom half is missing.



(b) MNIST-QU, where bottom right quarter is missing.

Figure 4: Missing data imputation with Monte Carlo procedure described in (Rezende, Mohamed, and Wierstra 2014) and our method. For the Markov chain procedure, the initial missing data is drawn randomly from $\text{Ber}(0.5)$ and imputed from the previous random generation in subsequent steps. MSNVI was trained with $\mathcal{L}^{(\text{ind})}$. For MNIST-QU, we used the PoE belief of the three observed quarters. The plots show the log-likelihood at every step of the Markov chain, marginalised over the dataset. Higher is better.

Table 2: Negative ELBOs and segmentation accuracy on Caltech-UCSD Birds 200. The IWAE was trained with a single source and target observation. Models trained with $\mathcal{L}^{(\text{ind})}$ and $\mathcal{L}^{(\text{hybrid})}$ use all sources and targets, and $\mathcal{L}^{(\text{ind})*}$ and $\mathcal{L}^{(\text{hybrid})*}$ use all sources for inference, but learn the generative model of a single modality.

	$\mathcal{L}^{(\text{ind})}$	$\mathcal{L}^{(\text{ind})*}$	$\mathcal{L}^{(\text{hybrid})}$	$\mathcal{L}^{(\text{hybrid})*}$	IWAE
img-to-seg	5326	3264	5924	3337	**3228**
img-to-img	-26179	-26663	-29285	-29668	**-30415**
accuracy	0.808	0.870	0.810	**0.872**	0.855

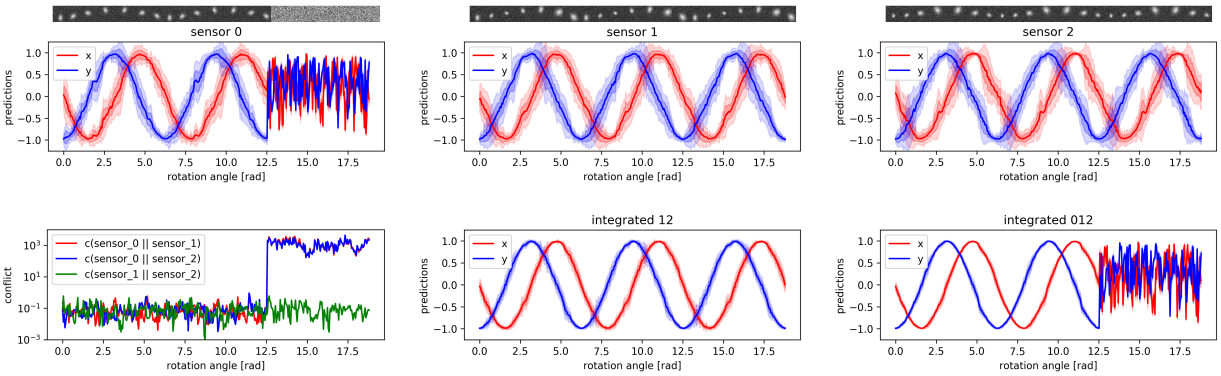


Figure 5: Predictions (x- and y-coordinates) of the pendulum position (figures 1, 2, 3, 5, 6) and conflict measure (figure 4). For the predictions, latent variables are inferred from images of 3 sensors with different views (top row) as well as their integrated beliefs (bottom mid and right). The figures show predictions (of the static model) for different angles of the pendulum, performing 3 rotations. After 2 rotations, failure of sensor 0 is simulated by outputting noise only. Lines show the mean and shaded areas show 1 and 2 standard deviations, estimated using 500 random samples of latent variables. Bottom left: The conflict measure of Eq. (2) for different angles of the pendulum.

of interest (segmentation), denoted with an asterisk. Models that have to learn the generative models for images and segmentations show worse ELBOs and accuracy, when evaluated on one modality. In contrast, the accuracy is slightly increased when we learn the generative model of segmentations only, but use both sources for inference.
We also refer the reader to the supplementary material of our technical report (Kurle, Günnemann, and Smagt 2018), where we visualise conditionally generated images, showing that learning with the importance sampling estimate of the ELBO is crucial to generate diverse samples from partially informed sources.

5.3 Robustness via conflict detection and redundancy

In this experiment we demonstrate how a shared latent representation can increase robustness, by exploiting sensor redundancy and the ability to detect conflicting data. We created a synthetic dataset of perspective images of a pendulum with different views of the same scene. The pendulum rotates along the z-axis and is centred at the origin. We simulate three cameras with 32×32-pixel resolution as information sources for inference and apply independent noise with std 0.1 to all sources. Each sensor is directed towards the origin (centre of rotation) from different view-points: Sensor 0 is aligned with the z-axis, and sensor 1 and 2 are rotated by $45\,\mathrm{deg}$ along the x- and y-axis, respectively. The distance of all sensors to the origin is twice the radius of the pendulum rotation. For the generative model we use the x- and y-coordinate of the pendulum rather than reconstructing the images. The model was trained with $\mathcal{L}^{(\mathrm{ind})}$.

In Fig. 5, we plot the mean and standard deviation of predicted x- and y-coordinates, where latent variables are inferred from a single source as well as from the PoE posteriors of different subsets. As expected, integrating the beliefs from redundant sensors reduces the predictive uncertainty. Additionally, we visualise the three images used as information sources above these plots.

Next, we simulate an anomaly in the form of a defect sensor 0, outputting random noise after 2 rotations of the pendulum. This has a detrimental effect on the integrated beliefs, where sensor 0 is part of the integration. We also plot the conflict measure of Eq. (2). As can be seen, the conflict measures for sensor 0 increases significantly when sensor 0 fails. In this case, one should integrate only the two remaining sensors with low conflict conjunctively.

6 Summary and future research directions

We extended neural variational inference to scenarios where multiple information sources are available. We proposed an objective function to learn individual inference models jointly with a shared generative model. We defined an exemplar measure (of conflict) to compare the beliefs from distinct inference models and their respective information sources. Furthermore, we proposed a disjunctive and a conjunctive integration method to combine arbitrary subsets of beliefs.

We compared the proposed objective functions experimentally, highlighting the advantages and drawbacks of each. Naive integration as a PoE ($\mathcal{L}^{(\mathrm{PoE})}$) leads to inseparable individual beliefs, while optimising the sources only individually ($\mathcal{L}^{(\mathrm{ind})}$) worsens the integration of the sources. On the other hand, a hybrid of the two objectives ($\mathcal{L}^{(\mathrm{hybrid})}$) achieves a good trade-off between both desiderata. Moreover, we showed how our method can be applied to structured output prediction and the benefits of exploiting the comparability of beliefs to increase robustness.

This work offers several future research directions. As an initial step, we considered only static data and a simple latent variable model. However, we have made no assumptions

about the type of information source. Interesting research directions are extensions to sequence models, hierarchical models and different forms of information sources such as external memory. Another important research direction is the combination of disjunctive and conjunctive integration methods, taking into account the conflict between sources.

Acknowledgements

We would like to thank Botond Cseke for valuable suggestions and discussions.

References

Andrew, G.; Arora, R.; Bilmes, J.; and Livescu, K. 2013. Deep canonical correlation analysis. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, III–1247–III–1255. JMLR.org.

Bach, F., and Jordan, M. 2005. A probabilistic interpretation of canonical correlation analysis.

Baltrušaitis, T.; Ahuja, C.; and Morency, L.-P. 2017. Multimodal machine learning: A survey and taxonomy. *arXiv preprint arXiv:1705.09406*.

Bousquet, N. 2008. Diagnostics of prior-data agreement in applied Bayesian analysis. *Journal of Applied Statistics* 35(9):1011–1029.

Burda, Y.; Grosse, R. B.; and Salakhutdinov, R. 2015. Importance weighted autoencoders. *CoRR* abs/1509.00519.

Cremer, C.; Li, X.; and Duvenaud, D. K. 2018. Inference suboptimality in variational autoencoders. *CoRR* abs/1801.03558.

Dahl, F. A.; Gåsemyr, J.; and Navig, B. A robust conflict measure of inconsistencies in Bayesian hierarchical models. *Scandinavian Journal of Statistics* 34(4):816–828.

Dempster, A. P. 1967. Upper and lower probabilities induced by a multivalued mapping. *Ann. Math. Statist.* 38(2):325–339.

Deng, Y. 2015. Generalized evidence theory. *Applied Intelligence* 43(3):530–543.

Denœux, T. 2008. Conjunctive and disjunctive combination of belief functions induced by nondistinct bodies of evidence. *Artificial Intelligence* 172(2):234 – 264.

Feng, F.; Li, R.; and Wang, X. 2015. Deep correspondence restricted boltzmann machine for cross-modal retrieval. *Neurocomputing* 154:50–60.

Gershman, S., and Goodman, N. D. 2014. Amortized inference in probabilistic reasoning. In *Proceedings of the 36th Annual Meeting of the Cognitive Science Society, CogSci 2014, Quebec City, Canada, July 23-26, 2014*.

Hardoon, D. R.; Szedmak, S. R.; and Shawe-taylor, J. R. 2004. Canonical correlation analysis: An overview with application to learning methods. *Neural Comput.* 16(12):2639–2664.

Hinton, G. E. 2002. Training products of experts by minimizing contrastive divergence. *Neural Comput.* 14(8):1771–1800.

Hotelling, H. 1936. Relations between two sets of variates. *Biometrika* 28(3/4):321–377.

Jiang, W.; Xie, C.; Zhuang, M.; Shou, Y.; and Tang, Y. 2016. Sensor data fusion with z-numbers and its application in fault diagnosis. *Sensors* 16(9).

Khaleghi, B.; Khamis, A.; Karray, F.; and Razavi, S. 2013. Multisensor data fusion: A review of the state-of-the-art. 14.

Kingma, D. P., and Welling, M. 2013. Auto-encoding variational Bayes. *CoRR* abs/1312.6114.

Kurle, R.; Günnemann, S.; and Smagt, P. v. d. 2018. Multi-Source Neural Variational Inference. *ArXiv e-prints* abs/1811.04451.

Lecun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

Mnih, A., and Gregor, K. 2014. Neural variational inference and learning in belief networks. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, 1791–1799.

Murphy, C. K. 2000. Combining belief functions when evidence conflicts. *Decis. Support Syst.* 29(1):1–9.

Ngiam, J.; Khosla, A.; Kim, M.; Nam, J.; Lee, H.; and Ng, A. Y. 2011. Multimodal deep learning. In Getoor, L., and Scheffer, T., eds., *ICML*, 689–696. Omnipress.

Rasiwasia, N.; Costa Pereira, J.; Coviello, E.; Doyle, G.; Lanckriet, G. R.; Levy, R.; and Vasconcelos, N. 2010. A new approach to cross-modal multimedia retrieval. In *Proceedings of the 18th ACM International Conference on Multimedia*, MM '10, 251–260. New York, NY, USA: ACM.

Rezende, D. J.; Mohamed, S.; and Wierstra, D. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31th International Conference on Machine Learning (ICML)*, 1278–1286.

Shafer, G. 1976. *A Mathematical Theory of Evidence*. Princeton: Princeton University Press.

Shon, A. P.; Grochow, K.; Hertzmann, A.; and Rao, R. P. N. 2005. Learning shared latent structure for image synthesis and robotic imitation. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, NIPS'05, 1233–1240. Cambridge, MA, USA: MIT Press.

Srivastava, N., and Salakhutdinov, R. 2014. Multimodal learning with deep boltzmann machines. *Journal of Machine Learning Research* 15:2949–2980.

Stein, B. E., and Meredith, M. A. 1993. *The merging of the senses*. Cambridge, MA, US: The MIT Press.

Suzuki, M.; Nakayama, K.; and Matsuo, Y. 2016. Joint multimodal learning with deep generative models.

Vedantam, R.; Fischer, I.; Huang, J.; and Murphy, K. 2017. Generative models of visually grounded imagination. *CoRR* abs/1705.10762.

Welinder, P.; Branson, S.; Mita, T.; Wah, C.; Schroff, F.; Belongie, S.; and Perona, P. 2010. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology.

Wu, M., and Goodman, N. 2018. Multimodal generative models for scalable weakly-supervised learning. *CoRR* abs/1802.05335.

Yang, J.; Safar, S.; and Yang, M.-H. 2014. Max-margin boltzmann machines for object segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition* 320–327.

Zadeh, L. A. 1986. A simple view of the dempster-shafer theory of evidence and its implication for the rule of combination. *AI Mag.* 7(2):85–90.

7 Appendix

A Individual inferences

In this section we derive the $\mathcal{L}^{(\text{ind})}$. Since any proposal distribution yields an ELBO to the log-marginal likelihood, the (weighted) average is also an ELBO.

$$\ln p_\theta(\mathbf{X}) = \sum_{n=1}^{N} \ln p_\theta(\mathbf{x}^{(n)})$$

$$\begin{aligned}
\ln p_\theta(\mathbf{x}^{(n)}) &= \ln \sum_{m=1}^{M} \pi_m \mathbb{E}_{\mathbf{z}_{1:K}^{(n)} \sim q_{\phi_m}\left(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)}\right)} \Big[\frac{1}{K} \sum_{k=1}^{K} w_{m,k}^{(n)}\Big] \\
&\geq \sum_{m=1}^{M} \pi_m \ln \mathbb{E}_{\mathbf{z}_{1:K}^{(n)} \sim q_{\phi_m}\left(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)}\right)} \Big[\frac{1}{K} \sum_{k=1}^{K} w_{m,k}^{(n)}\Big] \\
&\geq \sum_{m=1}^{M} \pi_m \mathbb{E}_{\mathbf{z}_{1:K}^{(n)} \sim q_{\phi_m}\left(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)}\right)} \Big[\ln \frac{1}{K} \sum_{k=1}^{K} w_{m,k}^{(n)}\Big],
\end{aligned}$$

where

$$w_{m,k}^{(n)} = \frac{p_\theta(\mathbf{x}^{(n)} \mid \mathbf{z}_k^{(n)})\, p_\theta(\mathbf{z}_k^{(n)})}{q_{\phi_m}(\mathbf{z}_k^{(n)} \mid \mathbf{x}_m^{(n)})}.$$

The factors π_m are the weights for each ELBO term, satisfying $0 \leq \pi_m \leq 1$ and $\sum_{m=1}^{M} \pi_m = 1$.

When $K = 1$, the gap between $\mathcal{L}^{(\text{ind})}$ and the marginal log-likelihood is the average Kullback-Leibler (KL) divergence between individual approximate posteriors and the true posterior from all sources:

$$\begin{aligned}
&\ln p_\theta(\mathbf{x}^{(n)}) - \mathcal{L}^{(\text{ind})} \\
&= \sum_{m=1}^{M} \pi_m D_{\mathrm{KL}}\big(q_{\phi_m}(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)}) \,||\, p_\theta(\mathbf{z}^{(n)} \mid \mathbf{x}^{(n)})\big)
\end{aligned}$$

This gap can be further decomposed as:

$$\begin{aligned}
&\sum_{m=1}^{M} \pi_m \mathrm{KL}(q_{\phi_m}(\mathbf{z} \mid \mathbf{x}_m) \,||\, p(\mathbf{z} \mid \mathbf{x})) \\
&= \sum_{m=1}^{M} \pi_m \mathrm{KL}(q_{\phi_m}(\mathbf{z} \mid \mathbf{x}_m) \,||\, p(\mathbf{z} \mid \mathbf{x}_m)) \\
&- \sum_{m=1}^{M} \pi_m \mathbb{E}_{\mathbf{z} \sim q_{\theta_m}(\mathbf{z} \mid \mathbf{x}_m)} \big[\ln p_{\theta_{-m}}(\mathbf{x}_{-m} \mid \mathbf{z})\big] \\
&+ \sum_{m=1}^{M} \pi_m \ln p_{\theta_{-m}}(\mathbf{x}_{-m} \mid \mathbf{x}_m) \\
&= \sum_{m=1}^{M} \pi_m \mathrm{KL}(q_{\phi_m}(\mathbf{z} \mid \mathbf{x}_m) \,||\, p(\mathbf{z} \mid \mathbf{x}_m)) \\
&- \sum_{m=1}^{M} \pi_m \mathbb{E}_{\mathbf{z} \sim q_{\theta_m}(\mathbf{z} \mid \mathbf{x}_m)} \big[\ln p_{\theta_{-m}}(\mathbf{x}_{-m} \mid \mathbf{z})\big] \\
&+ \sum_{m=1}^{M} \pi_m \ln \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z} \mid \mathbf{x}_m)} \big[p_{\theta_{-m}}(\mathbf{x}_{-m} \mid \mathbf{z})\big].
\end{aligned}$$

To minimise $\mathcal{L}^{(\text{ind})}$, not only the KL divergence of the individual approximate posterior and the respective true posterior need to be minimised, but also two additional terms which depend on the likelihood of those observations that have not been used as an information source for inference.

B Mixture of experts inference

The ELBO for the mixture distribution $\mathcal{L}^{(\text{MoE})}$ can be derived similarly. We employ a Monte Carlo approximation only w.r.t. each mixture component but not w.r.t. the mixture weights. That is, we enumerate all possible mixture components rather than sampling each from an indicator variable. This reduces variance of the estimate and circumvents the problem of propagating gradients through the sampling process of discrete random variables.

$$\begin{aligned}
&\ln p_\theta\big(\mathbf{x}^{(n)}\big) \\
&= \ln \int p_\theta\big(\mathbf{x}^{(n)}, \mathbf{z}^{(n)}\big) \frac{\sum_{m=1}^{M} \pi_m q_{\phi_m}(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)})}{\sum_{m'=1}^{M} \pi_{m'} q_{\phi_{m'}}(\mathbf{z}^{(n)} \mid \mathbf{x}_{m'}^{(n)})} d\mathbf{z}^{(n)} \\
&= \ln \sum_{m=1}^{M} \pi_m \mathbb{E}_{\mathbf{z}^{(n)} \sim q_{\phi_m}\left(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)}\right)} \left[w_{m,k}^{(n)}\right] \\
&= \ln \sum_{m=1}^{M} \pi_m \mathbb{E}_{\mathbf{z}^{(n)} \sim q_{\phi_m}\left(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)}\right)} \left[\frac{1}{K} \sum_{k=1}^{K} w_{m,k}^{(n)}\right] \\
&\geq \sum_{m=1}^{M} \pi_m \ln \mathbb{E}_{\mathbf{z}^{(n)} \sim q_{\phi_m}\left(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)}\right)} \left[\frac{1}{K} \sum_{k=1}^{K} w_{m,k}^{(n)}\right] \\
&\geq \sum_{m=1}^{M} \pi_m \mathbb{E}_{\mathbf{z}_{1:K}^{(n)} \sim q_{\phi_m}\left(\mathbf{z}^{(n)} \mid \mathbf{x}_m^{(n)}\right)} \left[\ln \frac{1}{K} \sum_{k=1}^{K} w_{m,k}^{(n)}\right],
\end{aligned}$$

$\mathcal{L}^{(\text{MoE})}$ minimises the average KL-divergence between the mixture of approximate posteriors and the true posterior from all sources:

$$\begin{aligned}
&\ln p(\mathbf{x}) - \mathcal{L}^{(\text{MoE})} \\
&= \sum_{m=1}^{M} \pi_m D_{\mathrm{KL}}\Big(\sum_{m'=1}^{M} \pi_{m'} q_{\phi_{m'}}(\mathbf{z} \mid \mathbf{x}_{m'}) \,||\, p(\mathbf{z} \mid \mathbf{x})\Big).
\end{aligned}$$

C Product of Gaussian experts

Here we consider the popular case of individual Gaussian approximate posteriors and a zero-centred Gaussian prior. Let the normal distributions be represented in the canonical form with canonical parameters $\{\Lambda; \eta\}$:

$$p(\mathbf{z}) = \frac{1}{Z(\mu, \Lambda)} \exp\Big(\eta^T \mathbf{z} - \frac{1}{2} \mathbf{z}^T \Lambda \mathbf{z}\Big).$$

Λ denotes the precision matrix and $\eta = \Lambda\mu$, where μ is the mean. Furthermore, $Z(\mu, \Lambda) = (2\pi)^{D_z/2} |\Lambda|^{-1/2} \exp(\frac{1}{2} \eta^T \Lambda^{-1} \eta)$ is the partition function.

Let the subscripts m, 0 and $*$ indicate the m-th approximate distribution, the prior and the integrated distribution. The natural parameters of the integrated variational posterior $q_\phi(\mathbf{zx})$ from Eq. (3) can then be calculated as follows:

$$\begin{aligned}
\Lambda_* &= \sum_{m=1}^{M} \Lambda_m - (M-1)\Lambda_0, \\
\eta_* &= \sum_{m=1}^{M} \eta_m - (M-1)\eta_0.
\end{aligned}$$

To obtain a valid integrated variational posterior, we require the precision matrix Λ_* to be positive semi-definite. This enforces requirements for the precision matrices Λ_m. In the case of diagonal precision matrices, the necessary and sufficient condition is that Λ_* has all positive entries. A sufficient condition for each entry $\Lambda_m[i]$ is $\Lambda_0[i] \leq \Lambda_m[i]$.

The partition function of the integrated belief can be calculated from the natural parameters, taking $\eta_* = \Lambda_* \mu_*$:

$$Z(\mu_*, \Lambda_*) = (2\pi)^{D_z/2} |\Lambda_*|^{-1/2} \exp\Big(\frac{1}{2} (\eta_*)^T (\Lambda_*)^{-1} \eta_*\Big). \quad (7)$$

D Point-wise mutual information

Inspecting Eq. (3), we can see that the negative logarithm of the constant term corresponds to the pointwise mutual information (PMI) between the observations. We do not need to calculate this constant since we impose assumptions about the parametric forms of the distributions and can calculate the partition function $Z(\mu_*, \Lambda_*)$ of the integrated belief using Eq. (7).

However, we can also calculate this partition function from the product of individual partition functions and the above mentioned constant in Eq. (3):

$$\frac{1}{Z(\mu_*, \Lambda_*)} = \left[\prod_{m=1}^{M} \frac{1}{Z(\mu_m, \Lambda_m)}\right] \cdot Z(\mu_0, \Lambda_0)^{(M-1)} \cdot \frac{\prod_{m=1}^{M} p(\mathbf{x}_m)}{p(\mathbf{x})}.$$

The PMI can then be calculated as:

$$\mathrm{PMI} = \ln\left(\frac{Z(\mu_0, \Lambda_0)^{M-1}}{\prod_{m=1}^{M} Z(\mu_m, \Lambda_m)} \cdot Z(\mu_*, \Lambda_*)\right).$$

The pointwise mutual information can be calculated between any subset of information sources. We note however, that it is based on the assumption that all involved probability density functions—the prior and all approximate posterior distributions—are normal distributions.

E Visualisation of samples from individual and integrated beliefs on mixture of bi-variate Gaussians dataset

For the mixture of bi-variate Gaussians dataset, we show latent samples from both information sources in Fig. 6a (left) and samples obtained by sampling importance re-sampling (SIR) using the full likelihood model in 6a (right). We also show random samples from the integrated beliefs as well as samples obtain by SIR in Fig. 6b (left) and 6b (right) respectively. We conclude that the integrated beliefs are much better proposal distributions, resolving the ambiguity of the individual sources.

F Visualisation of missing data imputation

Fig. 7 shows the mean of generated images for 50 steps of the Markov chain procedure for missing data imputation. As can be seen in Fig. 7a, the chain does not converge for many digits within 50 steps if too large portions of the data are missing. Indeed, we observed that the procedure randomly fails or succeeds to converge for the same input even after 150 steps.

G Conditional generations on Caltech-UCSD Birds 200

We show conditional generations of images, inferred from images or segmentation masks in Fig. 8. When inferring from segmentation masks, the conditional distribution $p_{\theta_{\mathrm{img}}}(\mathbf{x}_{\mathrm{img}} \mid \mathbf{x}_{\mathrm{label}})$ should be highly multimodal due to the missing colour information. This uncertainty should ideally be covered in the uncertainty of the belief. As can be seen in Fig. 8c, learning with a single importance sample leads to predictions of average images. For completeness, generated segmentation masks are shown in Fig. 9.

H Experiment setups

All inference (generative) models use the same neural network architectures for the different sources, except the first (last) layer, which depends on the dimensions of the data. We refer to main parts of the architectures, identical for each source, as "stem". In case of inference models, the stem is the input to a dense layer with linear (no activation) and sigmoid activations, parameterising the mean and std-dev of the approximate posterior distribution. In case of generative models, refer to the respective subsections.

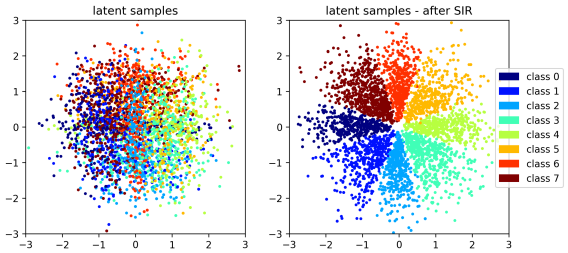
We use the Adam optimiser (Kingma and Ba 2014) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ in all experiments. In the tables, "dense" denotes fully connected layers, "conv" refers to convolutional layers, "pool" refers to pooling (down-sampling), and "interpol" refers to a bilinear interpolation (up-sampling). K is the number of importance-weighted samples and D_z refers to the number of latent dimensions, each modelled with a diagonal normal distribution with zero mean and unit standard deviation.

Partially observable mixture of bi-variate Gaussians In the pendulum experiment, we use 2 sources, corresponding to the x- and y-coordinates of the sample from a mixture of bi-variate Gaussians distribution. The neural network stems and training hyperparameters are summarised in Tab. 3. The generative models are both 1D Normal distributions, parameterised by linear dense layers, taking inputs from their respective stems.

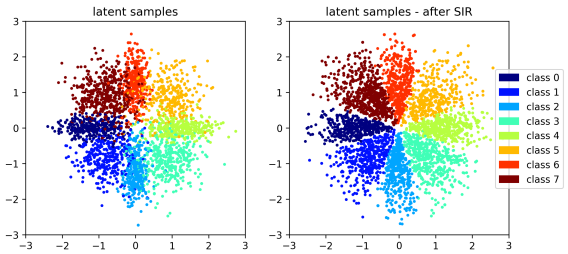
MNIST variants The neural network stems are summarised in Tab. 4. The data is modelled as Bernoulli distributions of dimensions 784 for MNIST-NO, 392 for MNIST-TB and 196 for MNIST-QU. The Bernoulli parameters are parameterised by linear dense layers, taking inputs from their respective stems.

Pendulum In the pendulum experiment, we use 3 sources with 32×32 images for the inference model, but a single observation of x- and y-coordinates of the pendulum centre. The generative

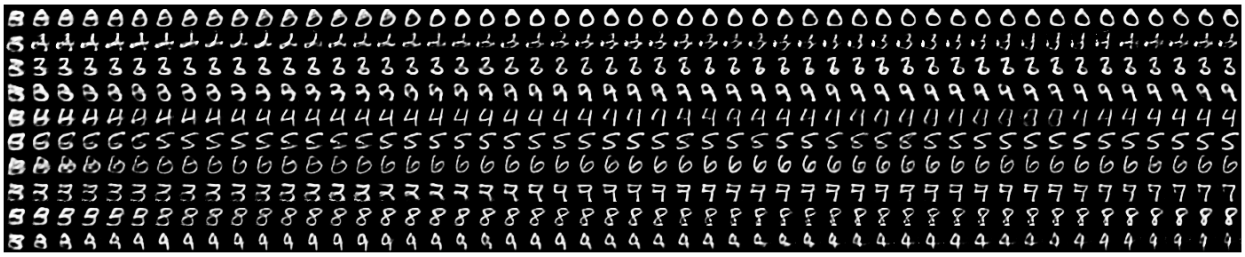
Figure 6: Samples from individual and integrated beliefs and samples obtained after SIR



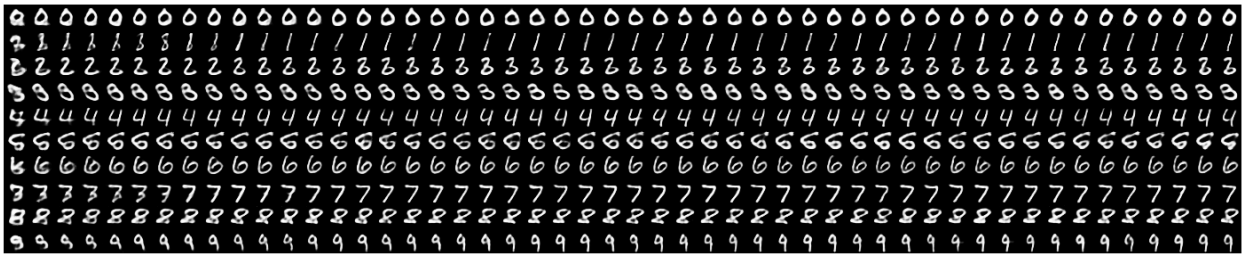
(a) Individual beliefs and their predictions. **Left:** Random samples from variational posterior without integration. Colours correspond to 8 test points, located at the means of the mixture of Gaussians data distribution. **Right:** Samples after sampling importance re-sampling using all likelihood functions.



(b) Integrated belief and its predictions. **Left:** Random samples from integrated variational posterior. Colours correspond to the test points. **Right:** Samples after sampling importance re-sampling using all likelihood functions.

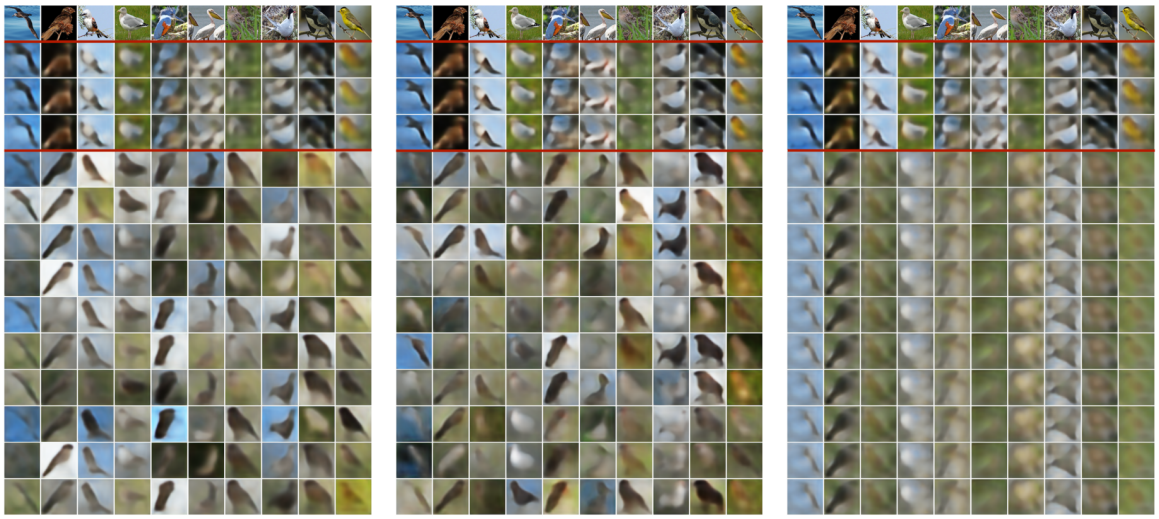


(a) Bottom half of the image is missing



(b) Bottom right quarter of the image is missing

Figure 7: Missing data imputation results: Mean of generated images. Observed data (fixed binarised) is kept unchanged, missing data is replaced with randomly generated (binary) image of previous iteration. The initial missing data is drawn randomly from $\mathrm{Ber}\,(0.5)$. Each of the 10 rows is an exemplar image of digits 0–9.

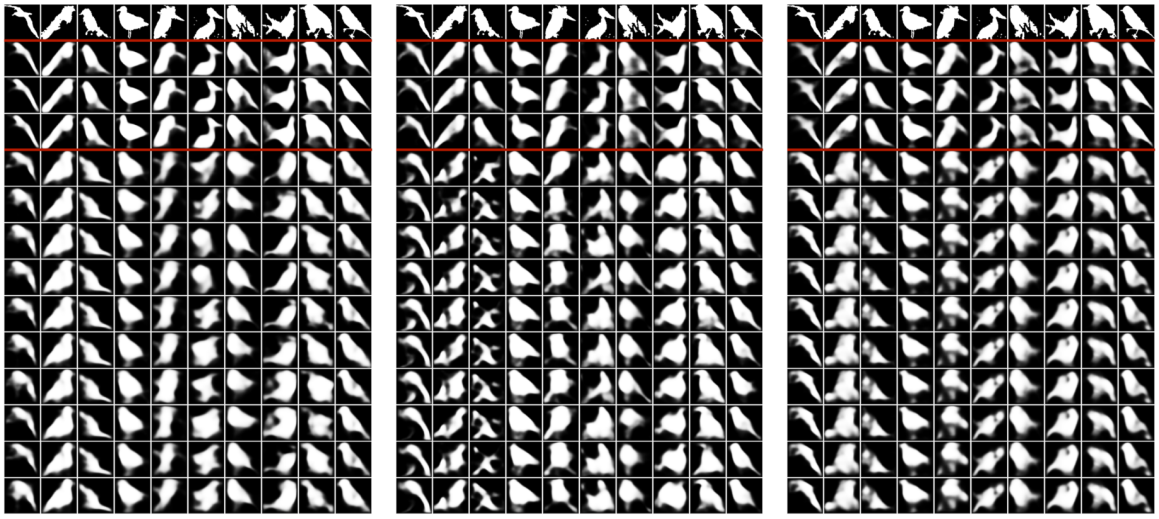


(a) Trained with $\mathcal{L}^{(\mathrm{ind})}$.

(b) Trained with $\mathcal{L}^{(\mathrm{hybrid})}$.

(c) Trained with $\mathcal{L}^{(\mathrm{hybrid})}$, K=1.

Figure 8: Conditional image generations, where latent variables are inferred from different sources. **Row 1:** Target observations. **Row 2–4:** Latent variables inferred from images. **Row 5–15:** Latent variables inferred from segmentation masks.



(a) Trained with $\mathcal{L}^{(\text{ind})}$. (b) Trained with $\mathcal{L}^{(\text{hybrid})}$. (c) Trained with $\mathcal{L}^{(\text{hybrid})}$, K=1.

Figure 9: Conditional segmentation mask generations, where latent variables are inferred from different sources. **Row 1:** Target observations. **Row 2–4:** Latent variables inferred from segmentation masks. **Row 5–15:** Latent variables inferred from images.

Table 3: Neural network architectures (stem) and hyperparameters used for experiments with partially observable mixture of bi-variate Gaussians

Inference models			
layer		activation	output shape
dense		tanh	32
dense		tanh	32
Generative models			
layer		activation	output shape
dense		tanh	32
dense		tanh	32
Hyperparameters			
K	D_z	batch size	learning rate	#iterations
8	2	32	0.0001	25k

Table 4: Neural network architectures and hyperparameters used for experiments with MNIST variants.

Inference models			
layer		activation	output shape
dense		elu	200
dense		elu	200
Generative models			
layer		activation	output shape
dense		elu	200
dense		elu	200
Hyperparameters			
K	D_z	batch size	learning rate	#iterations
16	16	128	0.00005	250k

model is assumed Normal for both coordinates, where the mean is predicted by a linear dense layer taking inputs from the stem, and the std deviation is a global variable. The neural network stems and training hyperparameters are summarised in Tab. 5.

Caltech-UCSD Birds 200 The neural network stems are summarised in Tab. 6. Images are modelled as diagonal normal distributions and segmentation masks as Bernoulli distributions. The generative model stem is the input to a 5×5-transposed convolutional layers with stride 2, yielding the mean of the likelihood function. The standard deviations are global and shared for all pixels. Leaky rectified linear units (lrelu) use $\alpha = 0.10$.

Table 5: Neural network architectures and hyperparameters used for perspective pendulum experiments

Inference models

layer	activation	output shape
dense	tanh	32
dense	tanh	32

Generative models

layer	activation	output shape
dense	tanh	256
dense	tanh	64
dense	tanh	16

Hyperparameters

K	D_z	batch size	learning rate	#iterations
16	2	16	0.00005	50k

Table 6: Neural network architectures and hyperparameters used for Caltech-UCSD Birds 200 experiments

Inference models

layer	kernel	stride	activation	output shape
conv	3×3	1	lrelu	128x128x16
conv	3×3	1	lrelu	128x128x16
pool	3×3	2	-	64x64x16
conv	3×3	1	lrelu	64x64x32
conv	3×3	1	lrelu	64x64x32
pool	3×3	2	-	32x32x32
conv	3×3	1	lrelu	32x32x48
conv	3×3	1	lrelu	32x32x48
pool	3×3	2	-	16x16x48
conv	3×3	1	lrelu	16x16x64
conv	3×3	1	lrelu	16x16x64
pool	3×3	2	-	8x8x64
conv	3×3	1	lrelu	8x8x96
conv	3×3	1	lrelu	8x8x96
pool	3×3	2	-	4x4x96
dense	-	-	linear	256

Generative models

layer	kernel	stride	activation	output shape
dense	-	-	linear	4x4x96
conv	3×3	1	lrelu	4x4x64
conv	3×3	1	lrelu	4x4x64
interpol	3×3	2	-	8x8x64
conv	3×3	1	lrelu	8x8x48
conv	3×3	1	lrelu	8x8x48
interpol	3×3	2	-	16x16x48
conv	3×3	1	lrelu	16x16x32
conv	3×3	1	lrelu	16x16x32
interpol	3×3	2	-	32x32x32
conv	3×3	1	lrelu	32x32x16
conv	3×3	1	lrelu	32x32x16
interpol	3×3	2	-	64x64x16

Hyperparameters

K	D_z	batch size	learning rate	#iterations
80	96	16	0.0002	25k

9 Deep Rao-Blackwellised Particle Filters for Time Series Forecasting

In the domain of time-series forecasting, learning probabilistic sequence models with accurate and efficient posterior approximations is complicated. The reason is that approximation errors typically accumulate with the length of the time series. While tractable models are usually too simple, particle filter approximations scale poorly with the state dimension. The approach taken in this chapter builds on the well-known SGLS that allows for efficient inference with a Rao-Blackwellised particle filter. In order to address two significant limitations of SGLSs, a state-to-switch recurrence and a decoder-type emission model are introduced. The state-to-switch recurrence improves long-term forecasts by propagating relevant information forward in time, and the emission model allows for non-linear emissions with non-Gaussian noise. An efficient Rao-Blackwellised particle filter is developed for this new model that computes expectations w.r.t. variables in the conditionally Gaussian linear part of the model in closed-form and approximates expectations w.r.t. the remaining variables using SMC.

The relevant background is provided in Sec. 2.3 – 2.5, and Ch. 5.

Authors	Richard Kurle Syama Sundar Rangapuram Emmanuel de Bézenac Stephan Günnemann Jan Gasthaus	
Conference	Advances in Neural Information Processing Systems, NeurIPS 2020	
Contribution	Problem definition	significantly contributed
	Literature survey	contributed
	Algorithm development	significantly contributed
	Method implementation	significantly contributed
	Experimental evaluation	significantly contributed
	Preparation of the manuscript	significantly contributed

Deep Rao-Blackwellised Particle Filters for Time Series Forecasting

Richard Kurle[*,†,2] **Syama Sundar Rangapuram**[1] **Emmanuel de Bezenac**[†,3]
Stephan Günnemann[2] **Jan Gasthaus**[1]
[1]AWS AI Labs [2]Technical University of Munich [3]Sorbonne Université

Abstract

This work addresses efficient inference and learning in switching Gaussian linear dynamical systems using a Rao-Blackwellised particle filter and a corresponding Monte Carlo objective. To improve the forecasting capabilities, we extend this classical model by conditionally linear state-to-switch dynamics, while leaving the partial tractability of the conditional Gaussian linear part intact. Furthermore, we use an auxiliary variable approach with a decoder-type neural network that allows for more complex non-linear emission models and multivariate observations. We propose a Monte Carlo objective that leverages the conditional linearity by computing the corresponding conditional expectations in closed-form and a suitable proposal distribution that is factorised similarly to the optimal proposal distribution. We evaluate our approach on several popular time series forecasting datasets as well as image streams of simulated physical systems. Our results show improved forecasting performance compared to other deep state-space model approaches.

1 Introduction

The Gaussian linear dynamical system (GLS) [4, 38, 31] is one of the most well-studied dynamical models with wide-ranging applications in many domains, including control, navigation, and time-series forecasting. This state-space model (SSM) is described by a (typically Markovian) latent linear process that generates a sequence of observations. The assumption of Gaussian noise and linear state transitions and measurements allows for exact inference of the latent variables—such as filtering and smoothing—and computation of the marginal likelihood for system identification (learning). However, most systems of practical interest are non-linear, requiring more complex models.

Many approximate inference methods have been developed for non-linear dynamical systems: Deterministic methods approximate the filtering and smoothing distributions e.g. by using a Taylor series expansion of the non-linear functions (known as extended Kalman filter (EKF) and second-order EKF) or by directly approximating these marginal distributions by a Gaussian using moment matching techniques [26, 27, 2, 3, 36]. Stochastic methods such as particle filters or smoothers approximate the filtering and smoothing distributions by a set of weighted samples (particles) using sequential Monte Carlo (SMC) [13, 16]. Furthermore, system identification with deep neural networks has been proposed using stochastic variational inference [17, 15] and variational SMC [23, 29, 20].

A common challenge with both types of approximations is that predictions/forecasts over long forecast horizons suffer from accumulated errors resulting from insufficient approximations at every time step. Switching Gaussian linear systems (SGLS) [1, 12]—which use additional latent variables to switch between different linear dynamics—provide a way to alleviate this problem: the conditional linearity can be exploited by approximate inference algorithms to reduce approximation errors. Unfortunately, this comes at the cost of reduced modelling flexibility compared to more general non-linear dynamical systems. Specifically, we identify the following two weaknesses of the SGLS: i) the switch transition

[*]Correspondence to richard.kurle@tum.de. [†]Work done while at AWS AI Labs.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

model is assumed independent of the GLS state and observation; ii) conditionally-linear emissions are insufficient for modelling complex multivariate data streams such as video data, while more suitable emission models (e.g. using convolutional neural networks) exist. The first problem has been addressed in [21] through augmentation with a Polya-gamma-distributed variable and a stick-breaking process. However, this approach uses Gibbs sampling to infer model parameters and thus does not scale well to large datasets. The second problem is addressed in [11] using an auxiliary variable between GLS states and emissions and stochastic variational inference to obtain a tractable objective function for learning. Yet, this model predicts the GLS parameters deterministically using an LSTM with an auto-regressive component, resulting in poor long-term forecasts.

We propose auxiliary variable recurrent switching Gaussian linear systems (ARSGLS), an extension of the SGLS to address both weaknesses by building on ideas from [21] and [11]. ARSGLS improves upon the SGLS by incorporating a conditionally linear state-to-switch dependence, which is crucial for accurate long-term out-of-sample predictions (forecasting), and a decoder-type neural network that allows modelling multivariate time-series data with a non-linear emission/measurement process. As in the SGLS, a crucial feature of ARSGLS is that approximate inference and likelihood estimation can be Rao-Blackwellised, that is, expectations involving the conditionally linear part of the model can be computed in closed-form, and only the expectations wrt. the switch variables need to be approximated. We leverage this feature and propose a Rao-Blackwellized filtering objective function and a suitable proposal distribution for this model class.

We evaluate our model with two different instantiations of the GLS: in the first scenario, the GLS is implemented with a constrained structure that models time-series patterns such as level, trend and seasonality [14, 34]; the second scenario considers a general, unconstrained conditional GLS. We compare our approach to closely related methods such as Deep State Space Models (DeepState) [30] and Kalman Variational Auto-Encoders (KVAE) [11] on 5 popular forecasting datasets and multivariate (image) data streams generated from a physics engine as used in [11]. Our results show that the proposed model achieves improved performance for univariate and multivariate forecasting.

2 Background

2.1 Gaussian Linear Dynamical Systems

The GLS models sequence data using a first-order Markov linear latent and emission process:

$$\mathbf{x}_t = A\mathbf{x}_{t-1} + B\mathbf{u}_t + \mathbf{w}_t, \qquad \mathbf{w}_t \sim \mathcal{N}(0, R), \tag{1a}$$

$$\mathbf{y}_t = C\mathbf{x}_t + D\mathbf{u}_t + \mathbf{v}_t, \qquad \mathbf{v}_t \sim \mathcal{N}(0, Q), \tag{1b}$$

where the vectors $\mathbf{u}$, $\mathbf{x}$ and $\mathbf{y}$ denote the inputs (also referred to as covariates or controls), latent states, and targets (observations). A, and C are the transition and emission matrix, B and D are optional control matrices, and R, Q are the state and emission noise covariances. In the following, we will omit the optional inputs $\mathbf{u}$ to ease the presentation, however, we use inputs e.g. for our experiments in Sec. 5.2. The appealing property of the GLS is that inference and likelihood computation is analytically tractable using the well-known Kalman filter algorithm, alternating between a prediction step $p(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}) = \int p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) p(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}) d\mathbf{x}_{t-1}$ and update step $p(\mathbf{x}_t \mid \mathbf{y}_{1:t}) \propto p(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}) p(\mathbf{y}_t \mid \mathbf{x}_t)$, where $p(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ is the state transition and $p(\mathbf{y}_t \mid \mathbf{x}_t)$ is the emission/measurement process corresponding to Eqs. (1a) and (1b), respectively.

2.2 Particle Filters

In non-linear dynamical systems, the filter distribution $p(\mathbf{x}_t \mid \mathbf{y}_{1:t})$ is intractable and needs to be approximated. Particle filters are SMC algorithms that approximate the filter distribution at every time-step t by a set of P weighted particles $\{\mathbf{x}^{(p)}\}^P$, combining importance sampling and re-sampling. Denoting the Dirac measure centred at $\mathbf{x}_t$ by $\delta(\mathbf{x}_t)$, the filter distribution is approximated as

$$p(\mathbf{x}_t \mid \mathbf{y}_{1:t}) \approx \sum_{p=1}^{P} w_t^{(p)} \delta(\mathbf{x}_t^{(p)}). \tag{2}$$

In first-order Markovian dynamical systems, the importance-weights are computed recursively as

$$\tilde{w}_t^{(p)} = w_{t-1}^{(p)} \gamma(\mathbf{x}_t^{(p)}, \mathbf{x}_{t-1}^{(p)}), \quad w_t^{(p)} = \frac{\tilde{w}_t^{(p)}}{\sum_{p=1}^{P} \tilde{w}_t^{(p)}}, \quad \gamma(\mathbf{x}_t^{(p)}; \mathbf{x}_{t-1}^{(p)}) = \frac{p(\mathbf{y}_t \mid \mathbf{x}_t^{(p)}) p(\mathbf{x}_t^{(p)} \mid \mathbf{x}_{t-1}^{(p)})}{\pi(\mathbf{x}_t^{(p)} \mid \mathbf{x}_{1:t-1}^{(p)})}, \tag{3}$$

where $\tilde{w}_t^{(p)}$ and $w_t^{(p)}$ denote the unnormalised and normalised importance-weights, respectively, $\gamma(\mathbf{x}_t^{(p)}; \mathbf{x}_{t-1}^{(p)})$ is the incremental importance-weight, and $\pi(\mathbf{x}_t \mid \mathbf{x}_{1:t-1}^{(p)})$ is the proposal distribution. To alleviate weight degeneracy issues, a re-sampling step is performed if the importance-weights satisfy a chosen degeneracy criterion. A common criterion is to re-sample when the effective sample size (ESS) drops below half the number of particles, i.e. $P_{\text{ESS}} = \left(\sum_{p=1}^{P} (w^{(p)})^2\right)^{-1} \leq P/2$.

2.3 Switching Gaussian Linear Systems with Rao-Blackwellised Particle Filters

Switching Gaussian linear systems (SGLS), also referred to as conditional GLS or mixture GLS, are a class of non-linear SSMs that use additional (non-linear) latent variables $\mathbf{s}_{1:t}$ that index the parameters (transition, emission, control, and noise covariance matrices) of a GLS, allowing them to "switch" between different (linear) regimes:

$$\begin{aligned} \mathbf{x}_t &= A(\mathbf{s}_t)\mathbf{x}_{t-1} + B(\mathbf{s}_t)\mathbf{u}_t + \mathbf{w}_t(\mathbf{s}_t), & \mathbf{w}_t(\mathbf{s}_t) &\sim \mathcal{N}(0, R(\mathbf{s}_t)), \\ \mathbf{y}_t &= C(\mathbf{s}_t)\mathbf{x}_t + D(\mathbf{s}_t)\mathbf{u}_t + \mathbf{v}_t(\mathbf{s}_t), & \mathbf{v}_t(\mathbf{s}_t) &\sim \mathcal{N}(0, Q(\mathbf{s}_t)). \end{aligned} \tag{4}$$

The switch variables $\mathbf{s}_t$ are typically categorical variables, indexing one of K base matrices; however, other choices are possible (see Sec. 3.1). Omitting the inputs $\mathbf{u}$ again, the graphical model factorises as $p(\mathbf{y}_{1:T}, \mathbf{x}_{0:T}, \mathbf{s}_{1:T}) = p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{y}_t \mid \mathbf{x}_t, \mathbf{s}_t) p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{s}_t) p(\mathbf{s}_t \mid \mathbf{s}_{t-1})$, where $p(\mathbf{s}_1 \mid \mathbf{s}_0) = p(\mathbf{s}_1)$ is the switch prior (conditioned on $\mathbf{u}_1$ if inputs are given). Note that we use an initial state $\mathbf{x}_0$ without corresponding observation for convenience.[2]

A crucial property of this model is that, while the complete model exhibits non-linear dynamics, given a sample of the trajectory $\mathbf{s}_{1:T}^{(p)}$, the rest of the model is (conditionally) linear. This allows for efficient approximate inference and likelihood estimation. The so-called Rao-Blackwellised particle filter (RBPF) leverages the conditional linearity by approximating expectations—that occur in filtering and likelihood computation—wrt. the switch variables $\mathbf{s}_{1:T}$ using SMC, while computing expectations wrt. the state variables $\mathbf{x}_{1:T}$ analytically. To achieve this, the posterior distribution is factorised as

$$p(\mathbf{x}_{1:t}, \mathbf{s}_{1:t} \mid \mathbf{y}_{1:t}) = p(\mathbf{x}_{1:t} \mid \mathbf{s}_{1:t}, \mathbf{y}_{1:t}) p(\mathbf{s}_{1:t} \mid \mathbf{y}_{1:t}). \tag{5}$$

The first term of Eq. (5) is tractable given a sample trajectory $\mathbf{s}_{1:t}^{(p)}$ using a Kalman smoother. However, for forecasting and loss computation (Sec. 3.3.2) we only require the state from the last step $p(\mathbf{x}_t \mid \mathbf{s}_{1:t}, \mathbf{y}_{1:t})$ for which even the Kalman filter suffices (cf. supplementary material for details). The second term of Eq. (5) can also be computed recursively using Bayes rule:

$$p(\mathbf{s}_{1:t} \mid \mathbf{y}_{1:t}) \propto p(\mathbf{y}_t \mid \mathbf{s}_{1:t}, \mathbf{y}_{1:t-1})\, p(\mathbf{s}_t \mid \mathbf{s}_{t-1}, \cancel{\mathbf{s}_{1:t-2}}, \cancel{\mathbf{y}_{1:t-1}})\, p(\mathbf{s}_{1:t-1} \mid \mathbf{y}_{1:t-1}). \tag{6}$$

Here and in the following we explicitly show the terms that cancel due to the Markov-property. The predictive distribution

$$p(\mathbf{y}_t \mid \mathbf{s}_{1:t}, \mathbf{y}_{1:t-1}) = \int p(\mathbf{y}_t \mid \mathbf{x}_t, \mathbf{s}_t, \cancel{\mathbf{s}_{1:t-1}}, \cancel{\mathbf{y}_{1:t-1}}) p(\mathbf{x}_t \mid \mathbf{s}_{1:t}, \mathbf{y}_{1:t-1}) d\mathbf{x}_t$$

is a by-product of the first term in Eq. (5); it can be computed in closed-form using the Kalman filter. Since all terms in $p(\mathbf{s}_{1:t} \mid \mathbf{y}_{1:t})$ can be computed in closed-form, this distribution can be approximated by a set of particles using SMC (cf. Eq. (2)). The corresponding incremental importance-weights are

$$\gamma(\mathbf{s}_t^{(p)}; \mathbf{s}_{1:t-1}^{(p)}) = \frac{p(\mathbf{y}_t \mid \mathbf{s}_{1:t}^{(p)}, \mathbf{y}_{1:t-1}) p(\mathbf{s}_t^{(p)} \mid \mathbf{s}_{t-1}^{(p)})}{\pi(\mathbf{s}_t^{(p)} \mid \mathbf{s}_{1:t-1}^{(p)})}. \tag{7}$$

The resulting joint filter distribution for both the state and switch variables is a mixture of Gaussians:

$$p(\mathbf{x}_t, \mathbf{s}_t \mid \mathbf{y}_{1:t}) \approx \sum_{p=1}^{P} w_t^{(p)} \delta(\mathbf{s}_t^{(p)})\, \mathcal{N}\big(\mathbf{x}_t \mid m_t(\mathbf{s}_{1:t}^{(p)}), V_t(\mathbf{s}_{1:t}^{(p)})\big), \tag{8}$$

where the history $\mathbf{s}_{1:t-1}^{(p)}$ is dropped. To be precise, the mean $m_t(\mathbf{s}_{1:t}^{(p)})$ and variance $V_t(\mathbf{s}_{1:t}^{(p)})$ of the filtered state $\mathbf{x}_t$ depend on the whole trajectory $\mathbf{s}_{1:t}^{(p)}$, although from an algorithmic perspective, only the current switch $\mathbf{s}_t^{(p)}$ is required to compute the incremental importance-weights in Eq.(7).

[2] We do not include an initial, unconditional switch variable $\mathbf{s}_0$, as the optimal proposal distribution would be proportional to $p(\mathbf{s}_0, \mathbf{x}_0)$; thus not using observations in the initial proposal distribution would lead to inefficient proposals for the following time steps.

3 Auxiliary Variable Recurrent Switching Gaussian Linear Systems

The SGLS provides an attractive tradeoff between model complexity and efficient inference. However, this model is limited in the following respects: i) switch transitions are assumed to be independent of the state variable; ii) multivariate observations with a complex non-linear dependence—such as streams of image data—are not well-described by a conditionally linear emission model. We propose the auxiliary variable recurrent SGLS (ARSGLS) to address both of these issues in Secs. 3.1 and 3.2. Furthermore, we leverage the conditional linearity using a RBPF (Sec. 3.3) with a suitable proposal distribution that is structurally simliary to the optimal (minimum variance) proposal distribution. The resulting graphical model and corresponding proposal distribution is visualised in Fig. 1 and the algorithm for filtering and loss computation is presented in Alg. 1 in the supplementary material.

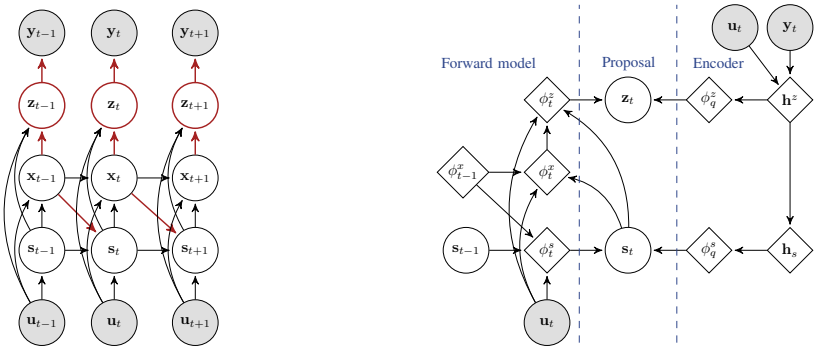


Figure 1: Left: Forward graphical model of ARSGLS, incl. inputs (controls) $\mathbf{u}_t$. Shaded/unshaded nodes indicate observed/hidden variables. New components (auxiliary variable, state-to-switch recurrence) are marked in red. Right: Inference proposal distribution for time step t. The left part is identical (shared) to the forward model, the right part is the approximated likelihood function given by a ladder-type encoder, and the proposal distribution results from taking the product of the two respective Gaussian functions (cf. Sec. 3.3.3, Eq. (14)). To visualise the marginalisation of $\mathbf{x}_{t-1}$ and $\mathbf{x}_t$ as well as the product of Gaussian functions of the variables $\mathbf{s}_t$ and $\mathbf{z}_t$, we explicitly denote the distribution parameters (location, covariance) of the respective Gaussians by ϕ.

3.1 Gaussian Recurrent Switch Transition

Although the classical SGLS is capable of generating sequences with non-linear dynamics (due to non-linear switches), it makes a very limiting assumption: the switch variables $\mathbf{s}_{1:T}$ are independent of the state variables $\mathbf{x}_{1:T}$, resulting in open loop switch dynamics. However, most non-linear systems can be approximated by a conditionally linear system only through linearisation *at the current state*. Thus, a feedback (closed loop) coupling with the states $\mathbf{x}_{t-1}$ would be necessary (cf. Fig. 2 for an example with a simple non-linear system). Unfortunately, using a more powerful switch transition model $p(\mathbf{s}_t \mid \mathbf{s}_{t-1}, \mathbf{x}_{t-1})$ complicates inference [5, 21]: In order to compute $p(\mathbf{s}_{1:t} \mid \mathbf{y}_{1:t}) = \int p(\mathbf{s}_{1:t}, \mathbf{x}_t \mid \mathbf{y}_{1:t}) d\mathbf{x}_t$, the previous state variable must be integrated out in closed-form (since we want to avoid sampling the states). While this problem has been addressed in [21], we take a different route that admits a re-parametrised proposal distribution, suitable for optimisation with stochastic gradient descent. It has been shown that Gaussian-distributed switch variables [6] can perform as well or better than continuous relaxations of categorical variables [24] in this setting. We propose to use Gaussian switch variables with a learnable Gaussian prior $p(\mathbf{s}_1)$ and conditionally linear state-to-switch transformations that allow for closed-form marginalisation of the state variable:

$$\mathbf{s}_t^{(p)} = F(\mathbf{s}_{t-1}^{(p)})\mathbf{x}_{t-1} + f(\mathbf{s}_{t-1}^{(p)}) + \epsilon_t(\mathbf{s}_{t-1}^{(p)}), \qquad \epsilon_t(\mathbf{s}_{t-1}^{(p)}) \sim \mathcal{N}\big(0, S(\mathbf{s}_{t-1}^{(p)})\big), \tag{9}$$

where f is a non-linear function (neural network), and $F(\mathbf{s}_{t-1})$ and $S(\mathbf{s}_{t-1})$ are the transition and covariance matrix predicted by the previous switch variable. Marginalising the states results in

$$p(\mathbf{s}_t|\mathbf{s}_{1:t-1}^{(p)}) = \mathcal{N}\big(\mathbf{s}_t; F(\mathbf{s}_{t-1}^{(p)})m_{t-1}(\mathbf{s}_{1:t-1}^{(p)}) + f(\mathbf{s}_{t-1}^{(p)}),\ F(\mathbf{s}_{t-1}^{(p)})V_{t-1}(\mathbf{s}_{1:t-1}^{(p)})F^T(\mathbf{s}_{t-1}^{(p)}) + S(\mathbf{s}_{t-1}^{(p)})\big). \tag{10}$$

The SGLS parameters $\{A, B, C, D, R, Q, F, S\}$ can be predicted from the Gaussian $\mathbf{s}_t^{(p)}$ and $\mathbf{s}_{t-1}^{(p)}$ in different ways, e.g. i) as the direct outputs a neural network; or ii) as a weighted average of a set of base matrices, where the weights are predicted by a neural network. We use the latter approach, which has been shown to be effective by previous work [6]. Deterministic weighted averages are also used in [37, 15, 11], while the "direct" approach of predicting the parameters is used e.g. by [30].

3.2 Non-linear Multivariate Emission Model

Time-series data is often non-Gaussian or multivariate with a complex dependence on hidden variables (e.g. images, point clouds, discrete data). To model such data, we augment the recurrent SGLS with a Gaussian latent variable $\mathbf{z}_{1:T}$ that decouples the observations from the SGLS through the conditional independence $p(\mathbf{x}_{1:t} \mid \mathbf{s}_{1:t}, \mathbf{z}_{1:t}, \mathbf{y}_{1:t}) = p(\mathbf{x}_{1:t} \mid \mathbf{s}_{1:t}, \mathbf{z}_{1:t})$. The resulting joint distribution is given as

$$p(\mathbf{y}_{1:T}, \mathbf{z}_{1:T}, \mathbf{x}_{0:T}, \mathbf{s}_{1:T}) = p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{y}_t \mid \mathbf{z}_t) p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s}_t) p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{s}_t) p(\mathbf{s}_t \mid \mathbf{s}_{t-1}, \mathbf{x}_{t-1}).$$

This *auxiliary variable* $\mathbf{z}_t$ enables the use of arbitrary conditional distributions $p(\mathbf{y}_t \mid \mathbf{z}_t)$ for which the distribution parameters are predicted by a non-linear emission model such as a neural network. Samples of the auxiliary variables can be interpreted as pseudo-observations in the SGLS. This emission model and auxiliary is similar to the additional latent variable in the KVAE [11]; cf. Sec. 4 for further details on the similarities and differences.

3.3 Inference and Parameter Estimation

We extend the RBPF from Sec. 2.3 to infer the latent variables $\{\mathbf{s}, \mathbf{x}, \mathbf{z}\}$; and we use maximum likelihood estimation to learn the model parameters θ (shared between multiple time-series). These include i) the base matrices of the (recurrent) SGLS, ii) the parameters of the state prior, iii) the switch prior and transition (neural network), and iv) the auxiliary variable decoder (neural network).

3.3.1 Rao-Blackwellised Particle Filtering

Filtering can be performed analogous to the standard Rao-Blackwellised particle filter (cf. Sec. 2.3). To this end, we factorise the posterior distribution similarly as was shown for the SGLS:

$$p(\mathbf{z}_{1:t}, \mathbf{x}_{1:t}, \mathbf{s}_{1:t} \mid \mathbf{y}_{1:t}) = p(\mathbf{x}_{1:t} \mid \mathbf{z}_{1:t}, \mathbf{s}_{1:t}, \cancel{\mathbf{y}_{1:t}})\, p(\mathbf{z}_{1:t}, \mathbf{s}_{1:t} \mid \mathbf{y}_{1:t}). \tag{11}$$

The first term in Eq. (11) can again be computed in closed-form while the second density can be approximated using SMC. The incremental importance-weights for this model are given as

$$\gamma(\mathbf{s}_t^{(p)}, \mathbf{z}_t^{(p)};\, \mathbf{s}_{1:t-1}^{(p)}, \mathbf{z}_{1:t-1}^{(p)}) = \frac{p(\mathbf{y}_t \mid \mathbf{z}_t^{(p)})\, p(\mathbf{z}_t^{(p)} \mid \mathbf{z}_{1:t-1}^{(p)},\, \mathbf{s}_{1:t}^{(p)})\, p(\mathbf{s}_t^{(p)} \mid \mathbf{z}_{1:t-1}^{(p)},\, \mathbf{s}_{1:t-1}^{(p)})}{\pi(\mathbf{z}_t^{(p)}, \mathbf{s}_t^{(p)} \mid \mathbf{s}_{1:t-1}^{(p)}, \mathbf{z}_{1:t-1}^{(p)})}. \tag{12}$$

The numerator $p(\mathbf{y}_t, \mathbf{z}_t, \mathbf{s}_t \mid \mathbf{y}_{1:t-1}, \mathbf{z}_{1:t-1}, \mathbf{s}_{1:t-1})$ reveals that the switch transition (last term) with state-to-switch dynamics is no longer Markovian. This is because we marginalise the filtered state $p(\mathbf{s}_t \mid \mathbf{s}_{1:t-1}, \mathbf{z}_{1:t-1}) = \int p(\mathbf{s}_t \mid \mathbf{s}_{t-1}, \mathbf{x}_{t-1}) p(\mathbf{x}_{t-1} \mid \mathbf{z}_{1:t-1}, \mathbf{s}_{1:t-1}) d\mathbf{x}_{t-1}$ that depends on all previous switches and pseudo-observations similarly to the generative case in Eq. (10). As for the SGLS in Sec. 2.3, the conditional distribution of the auxiliary variable (second term) is a by-product of the Kalman filter prediction and update step that is required for computing the first term in Eq. (11):

$$p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}^{(p)}, \mathbf{s}_{1:t}^{(p)}) = \int p(\mathbf{z}_t \mid \mathbf{x}_t) p(\mathbf{x}_t \mid \mathbf{z}_{1:t-1}^{(p)}, \mathbf{s}_{1:t}^{(p)}) d\mathbf{x}_t,$$

$$p(\mathbf{x}_t \mid \mathbf{z}_{1:t-1}^{(p)}, \mathbf{s}_{1:t}^{(p)}) = \int p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{s}_t^{(p)}) p(\mathbf{x}_{t-1} \mid \mathbf{z}_{1:t-1}^{(p)}, \mathbf{s}_{1:t-1}^{(p)}) d\mathbf{x}_{t-1}.$$

3.3.2 Parameter Estimation

For learning the model parameters shared between different time-series we use maximum likelihood estimation and stochastic gradient descent (SGD). In the SMC setting, an unbiased estimator of the marginal likelihood $\hat{p}(\mathbf{y}_{1:T}; \theta) = \prod_{t=1}^{T} \sum_{p=1}^{P} \tilde{w}_t^{(p)}(\theta)$ can be obtained from the unnormalised importance-weights [9] (cf. supplementary material for more details). Consequently, $\mathbb{E}\big[\log \hat{p}(\mathbf{y}_{1:T}; \theta)\big] \leq \log p(\mathbf{y}_{1:T}; \theta)$ due to Jensen's inequality. Based on this, [29, 23, 20] proposed a tractable lower bound to the log-marginal likelihood that can be optimised with SGD:

$$\mathcal{L}(\mathbf{y}_{1:T}; \theta) = \sum_{t=1}^{T} \log \hat{p}(\mathbf{y}_t \mid \mathbf{y}_{1:t-1}; \theta) = \sum_{t=1}^{T} \log \sum_{p=1}^{P} \tilde{w}_t^{(p)}(\theta) \leq \log p(\mathbf{y}_{1:T}; \theta). \tag{13}$$

We propose to use the same objective function, however, leveraging the conditional linearity of our model, using the incremental importance-weights of Eq. (12) for computing the importance-weights (cf. Eq. (3)).

3.3.3 Ladder Proposal Distribution

Choosing a good proposal distribution is essential to obtain reliable estimates with low variance. The optimal (minimum variance) proposal distribution is proportional to the joint distribution $p(\mathbf{y}_t, \mathbf{z}_t, \mathbf{s}_t \mid \mathbf{s}_{1:t-1}, \mathbf{z}_{1:t-1}, \cancel{\mathbf{y}_{1:t-1}})$, i.e. the numerator of the incremental importance-weights. We therefore propose a proposal distribution with similar structure, while reusing the known densities:

$$\begin{aligned}\pi(\mathbf{z}_t, \mathbf{s}_t \mid \mathbf{z}_{1:t-1}, \mathbf{s}_{1:t-1}, \mathbf{y}_t) &\propto q(\mathbf{z}_t, \mathbf{s}_t \mid \mathbf{y}_t) p(\mathbf{z}_t, \mathbf{s}_t \mid \mathbf{z}_{1:t-1}, \mathbf{s}_{1:t-1}) \\ &= q(\mathbf{z}_t, \mathbf{s}_t \mid \mathbf{y}_t) p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{s}_{1:t}) p(\mathbf{s}_t \mid \mathbf{z}_{1:t-1}, \mathbf{s}_{1:t-1})\end{aligned} \tag{14}$$

The last two terms are the transition of the switches and the resulting auxiliary variable in the generative model; the first term is a Gaussian approximation of the likelihood, predicted by an encoder neural network.[3] Both pairs of Gaussians are combined as a product of experts, resulting in a Gaussian proposal distribution. The switch transition is readily available from the previous step $t-1$, while the predictive distribution for the auxiliary variable requires sampling $\mathbf{s}_t$ first. We therefore propose to structure the encoder with dependencies in the same direction as the generative model, i.e. as presented in Eq. (14). The resulting encoder resembles the encoder in the Ladder VAE [35]. Thus, combining it with the densities from the forward model, we obtain a proposal distribution $\pi(\mathbf{z}_t, \mathbf{s}_t \mid \mathbf{z}_{1:t-1}, \mathbf{s}_{1:t-1}, \mathbf{y}_t) = \pi(\mathbf{s}_t \mid \mathbf{z}_{1:t-1}, \mathbf{s}_{1:t-1}, \mathbf{y}_t)\pi(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{s}_{1:t}, \mathbf{y}_t)$ that is factorised such that samples can be drawn in the same direction as the generative process, while reusing the predictive quantities of the forward (generative) model. The proposal distribution is optimal if the Gaussian encoder distribution is proportional to the likelihood.

We use the effective sample-size criterion mentioned in Sec. 2.2 with *systematic re-sampling* [8]. Following previous work, we omit the score function estimator term for re-sampled variables, resulting in biased but lower variance gradient estimates. We refer to [23, 29, 20] for a discussion.

4 Related work

Extensions of the classical SGLS with a state-to-switch dependence have been proposed in previous work [5, 21, 6]: In [5], the required marginalisation of previous Gaussian states is approximated numerically through sampling. [21] uses logistic stick-breaking for the discrete switches and augments the model with a Polya-Gamma distributed auxiliary variable, rendering the states conjugate to the switch variables such that marginalising the Polya-Gamma variables leaves the original model intact. Inference in this model is accomplished through Gibbs sampling, whereas our work uses a RBPF and SGD for state and parameter inference. Different from these two approaches, [6] uses a Gaussian switch variable. In contrast to our work, inference is performed using stochastic variational inference without exploiting the conditional tractability of the SGLS, instead sampling both states and switches.

Many efficient approximate inference methods have been proposed for the classical SGLS, including approaches using variational inference [12], expectation propagation [39], and a RBPF [10]. Our ARSGLS uses an extension of RBPF that includes the additional auxiliary variable (cf. Sec. 3.2) in the SMC approximation. Learning through Monte Carlo objectives [28] for particle filters has been proposed in [23, 29, 20]. Our objective function is a special case that exploits the conditionally linear structure of the ARSGLS through Rao-Blackwellisation to reduce variance in the estimates.

Parameter learning with a conditional GLS and closed-form inference has been proposed previously in DeepState [30] and the KVAE [11]. Our model differs substantially in the i) graphical model, ii) latent variable inference and iii) parameter learning:
i) DeepState and KVAE can be interpreted as a *deterministic* SGLS, where the GLS parameters are predicted by an RNN, whereas our proposed model uses *probabilistic* switch transitions that receive feedback from the previous state. Furthermore, the GLS parameters in DeepState have a fixed structure that model time-series patterns such as level, trend and seasonality; transition and emission matrices are fixed and the two (diagonal) noise covariance matrices are predicted by the RNN directly. The KVAE uses unrestricted GLS parameters, which are predicted as a weighted average of a set of base matrices, where the weights are given by the RNN. Our proposed model uses a similar weighted

[3] Note that we omitted inputs $\mathbf{u}_t$; however we use both $\mathbf{u}_t$ and $\mathbf{y}_t$ in the encoder. Furthermore, we made the simplifying assumption that q only depends on the latest observation $\mathbf{y}_t$. One way to go beyond that would be to include $\mathbf{y}_{1:t-1}$ through sufficient statistics, such as the natural parameters, of the state variable $\mathbf{x}_{t-1}$. We decided to use only the latest observation $\mathbf{y}_t$ in our model, since information from $\mathbf{x}_{t-1}$ and thus $\mathbf{y}_{1:t-1}$ is incorporated into the proposal distribution through the state-to-switch recurrence.

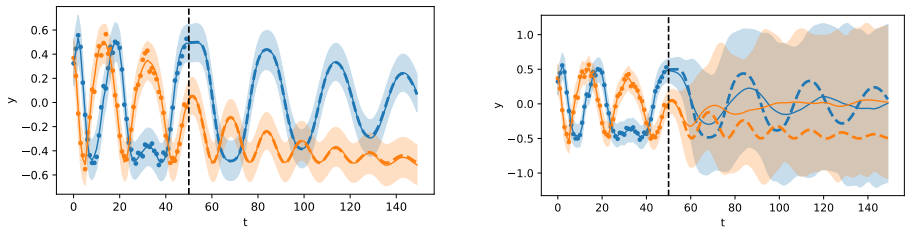


Figure 2: Filtered and forecasted emissions of a recurrent (left) and non-recurrent (right) SGLS for synthetic data of a swinging pendulum. Observations are noisy x-/y-position (blue/orange). The plots show the mean (line) and 3 std. deviations (shaded); filter and forecast range are separated by a vertical line, 50 (noisy) observations (dots) are used for filtering, the forecast ground-truth is shown as a dashed line. The SGLS relies on the filter update, whereas the RSGLS successfully uses state information to select the correct base matrices for prediction.

average of base matrices, but the weights are functions of a *probabilistic* Gaussian switch variable. A further difference to both models is the dependence of the RNN/switch transition on observations and controls: The RNN in DeepState is conditioned on inputs (controls), whereas the RNN in the KVAE depends (auto-regressively) on *samples* of the previous (pseudo-) observation of the GLS. In contrast, the switch transition of the ARSGLS is conditioned on the previous GLS state variable, thus using information from all previous inputs and observations. Finally, the KVAE uses the same type of auxiliary variable and non-linear emission model as our ARSGLS. However, the encoder is conditioned only on the current observations; in contrast, the structure of our proposal distribution (corresponding to the encoder in the KVAE) is chosen to resemble the optimal (minimum variance) proposal distribution, thus including the current observation and inputs as well as the predictive distribution of the previous state.
ii) Given the *deterministically* predicted GLS parameters, DeepState/KVAE uses the Kalman filter/smoother for inference. In contrast, we use a Rao-Blackwellised particle filter to infer non-linear switch variables through SMC and the conditionally linear states through the Kalman filter.
iii) DeepState uses maximum likelihood for parameter learning where the log-likelihood is estimated using the filter formulation (normalisation constants), whereas the KVAE is learnt using a variational EM objective. Our proposed model is learnt through a (Rao-Blackwellised) particle filter-based Monte Carlo objective (cf. prev. paragraph).

5 Experiments

5.1 Pendulum

We start with a qualitative assessment of the state-to-switch dependence from Sec. 3.1. We refer to this model (without the auxiliary variable of Sec. 3.2) as `RSGLS`. We consider noisy observations of the xy-positions of a dampened pendulum. Initial states (angle, angular velocity) are drawn randomly from a Gaussian centred at $180\,\mathrm{deg}$ (top position) and zero velocity. Both models are trained on 5000 sequences with 50 time-steps. For evaluation, we filter for 50 time-steps and forecast the next 100 time-steps. We visualise the resulting filter and forecast distribution together with noisy observations ($t \leq 50$) and ground-truth ($t > 50$) in Fig. 2. As can be seen, the `SGLS` can not learn linearised dynamics without information from the state. On the other hand, the `RSGLS` generates very accurate forecasts with reasonable predictive uncertainty, showing the necessity of state-to-switch dynamics.

5.2 Time series forecasting

We evaluate our approach in the context of time-series forecasting on 5 popular public datasets (`electricity`, `traffic`, `solar`, `exchange`, `wiki`) used in [32]. The data is recorded with hourly or daily frequency and exhibits seasonal patterns with different frequency (e.g. daily and weekly).

We experimented with two instantiations of our proposed model: Similar to previous work [30], we implement the GLS as an innovation state space model (ISSM) with a constrained structure that models temporal patterns such as level, trend and seasonality (cf. supplementary material for details). In this model, labelled as `RSGLS-ISSM`, the dimension of the state $\mathbf{x}_t$ and the transition and emission

CRPS rolling					
	exchange	solar	electricity	traffic	wiki
DeepAR	0.009±0.001	**0.357±0.002**	**0.057±0.003**	**0.120±0.003**	0.281±0.008
DeepState	0.010±0.001	0.379±0.002	0.071±0.000	0.131±0.002	0.296±0.007
KVAE-MC	0.010±0.000	0.377±0.005	0.319±0.010	0.233±0.014	0.276±0.028
KVAE-RB	0.009±0.000	0.384±0.005	0.296±0.024	0.179±0.001	0.245±0.004
RSGLS-ISSM (ours)	**0.007±0.000**	**0.355±0.004**	0.070±0.001	0.148±0.003	0.248±0.006
ARSGLS (ours)	0.009±0.000	0.369±0.008	0.138±0.003	0.136±0.005	**0.217±0.010**
CRPS long-term					
DeepAR	0.019±0.002	0.440±0.004	**0.062±0.004**	0.138±0.001	0.855±0.552
DeepState	0.017±0.002	0.379±0.002	0.088±0.007	**0.131±0.005**	0.338±0.017
KVAE-MC	0.020±0.001	0.389±0.005	0.318±0.011	0.261±0.016	0.341±0.032
KVAE-RB	0.018±0.001	0.393±0.006	0.305±0.022	0.221±0.002	0.317±0.013
RSGLS-ISSM (ours)	**0.014±0.001**	**0.358±0.001**	0.091±0.004	0.206±0.002	0.345±0.010
ARSGLS (ours)	0.022±0.001	0.371±0.007	0.154±0.005	0.175±0.008	**0.283±0.006**

Table 1: CRPS metrics (lower is better). Mean and std. deviation are computed over 4 independent runs for our method and 3 runs for the competing methods. The method achieving the best result is highlighted in **bold**.

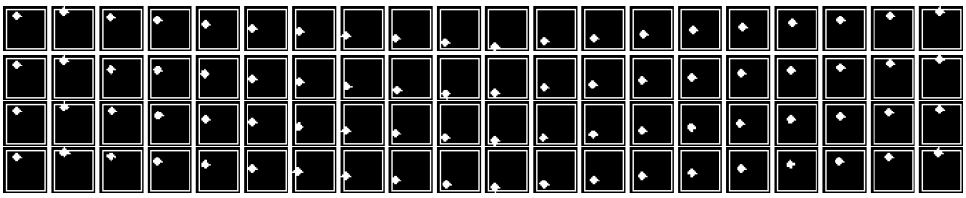


Figure 3: 20 time-steps of groundtruth (1st row) and 3 random trajectories from forecast distribution (remaining rows) for ARSGLS on box dataset. Forecasts were generated after 20 filtering time-steps (not shown).

matrices A and C are pre-defined and not learned; for this model, we do not use the non-linear emission model from Sec. 3.2. The second model, labelled ARSGLS, uses a general, unrestricted GLS; here we use the complete model, including a non-linear emission model. We compare to two closely related methods, i.e. DeepState [30] and KVAE [11], and a strong baseline that uses an autoregressive RNN (DeepAR) [33]. DeepState is implemented with the same ISSM structure as RSGLS-ISSM, whereas KVAE is implemented with the same unconstrained GLS and measurement model as ARSGLS. Furthermore, we note that the objective function proposed in [11] uses *samples* from the smoothing distribution. However, in line with this work, the corresponding expectation can be computed in closed-form; the resulting objective function has lower variance. We implemented both objective functions, denoting them as KVAE-MC (Monte Carlo) and KVAE-RB (Rao-Blackwellised), respectively.

We use data prior to a fixed forecast date for training and test the forecasts on the remaining data. The probabilistic forecasts are conditioned on the training range and computed with 100 samples for each method. We evaluate in a rolling fashion and with a single long-term forecast. In case of daily/hourly data, the rolling evaluation uses 5/7 windows each with a forecast covering 30 days/24 hours; long-term evaluation uses all 5/7 windows (i.e. 150 days/168 hours) without updating the model with the new data. We score forecasts using the *continuous ranked probability score* (CRPS) [25].

$$\mathrm{CRPS}(F^{-1}, y) = \int_0^1 2\Lambda_\alpha(F^{-1}(\alpha), y)d\alpha, \tag{15}$$

where $\Lambda_\alpha(q, y) = (\alpha - I_{[y<q]})(y - q)$ is the quantile loss at the quantile level $\alpha \in [0, 1]$, q is the predicted α-th quantile level and y is the observation. The results are summarised in Tab. 1. Both of our model variants compare favorably or similarly to their respective closely related competitive model (i.e. RSGLS-ISSM and DeepState; ARSGLS and KVAE). Only the auto-regressive baseline DeepAR remains challenging on 2/5 datasets.

5.3 Simulated physical environments

We evaluate our model for unsupervised forecasting of high-dimensional multivariate data streams (video), emitted from simulated physical environments. To this end, we consider the 4 synthetic

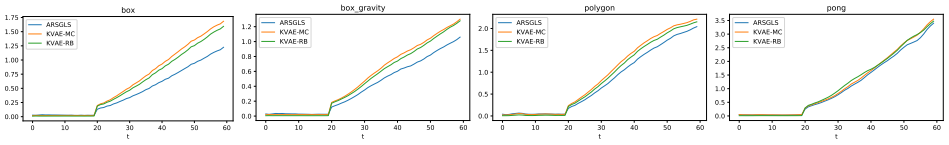


Figure 4: Wasserstein distance in filter/smoothing (ARSGLS/ KVAE) range ($t < 20$) and forecast range ($t \geq 20$). Results are averaged over 1000 test samples and 3 independent runs.

video datasets (Box, Box-Gravity, Polygon, Pong) used in [11], consisting of streams of 32×32 binary images of an object moving in an environment and colliding with walls. We compare our model to the KVAE-MC (proposed in [11]) and our Rao-Blackwellised version KVAE-RB cf. 5.2). We use the same model architecture as in [11], except that each model has 10-dimensional states and 10 base matrices (cf. App. 7.7.3 for more details). All models are trained on 5000 time-series of 20 time-steps. We visualise forecasted trajectories in Fig. 3.

Next, we test the learned hidden dynamics quantitatively, comparing the forecasts to the true data. This is challenging for image data: standard metrics such as predictive log-likelihood, (pixel-wise) accuracy or squared error do not differentiate between small and large dis-placements of object in the environment (e.g. if all pixels corresponding to the object are falsely predicted). Interpreting these binary images as a 2D distribution over the xy-positions of the objects in the scene, we propose to score the models using the Wasserstein metric with the Euclidean norm as distance function:

$$\mathrm{D}(\mathbf{y}_\tau, \hat{\mathbf{y}}_\tau) = \int \mathrm{W}(\mathbf{y}_\tau, \hat{\mathbf{y}}_\tau) p(\hat{\mathbf{y}}_\tau \mid \mathbf{y}_{1:t}) d\hat{\mathbf{y}}_\tau, \tag{16}$$

where $p(\hat{\mathbf{y}}_\tau \mid \mathbf{y}_{1:t})$ is the forecast distribution (time-steps $\tau > t$) or predictive distribution from the filter ($t = \tau$). The natural interpretation of this metric is the minimum average distance (in pixel coordinates) needed to move each pixel from the forecast to its true position in the xy-plane. We approximate $p(\hat{\mathbf{y}}_\tau \mid \mathbf{y}_{1:t}) = \int p(\hat{\mathbf{y}}_\tau \mid \mathbf{z}_t, \mathbf{x}_t, \mathbf{s}_t) p(\mathbf{z}_t, \mathbf{x}_t, \mathbf{s}_t \mid \mathbf{y}_{1:t}) d\mathbf{z}_t \mathbf{x}_t \mathbf{s}_t$ (no $\mathbf{s}_t$ in KVAE) using 32 samples and Monte Carlo integration. We evaluate the metric from Eq. (16)—averaged over 3 independent runs and 1000 test data points—for 20 time-steps of filtering (in case of ARSGLS) and smoothing (in case of KVAE) and 40 time-steps forecasting. Results (Fig. 4) show that ARSGLS produces more accurate forecasts.

6 Conclusion

In this work, we proposed an extension of the classical SGLS that addresses two weaknesses, while leaving the conditional tractability of this model intact. We improve the forecasting capabilities by conditionally linear state-to-switch recurrence with Gaussian switch variables, keeping the conditional tractability of the GLS intact. Furthermore, we augment the emission model by an auxiliary variable that allows for modelling multi-variate and non-Gaussian observations with complex non-linear transformations such as convolutional neural networks. We leverage the conditional linear structure of this model using Rao-Blackwellised particle filtering and we propose a corresponding Monte Carlo objective for parameter estimation. Experiments on popular time series forecasting datasets and simulated video data from physical environments demonstrate improvements compared to other deep state-space model approaches.

This work offers several interesting research directions: the proposed approach can be generalised to hierarchical models, interleaving linear and non-linear variables; observed data and the corresponding ladder-encoder can be extended to handle multimodal and missing data similar to [19]; offline inference can be done through Rao-Blackwellised particle smoothing [22] and used for learning with a corresponding variational EM objective function; finally, the model parameters may be inferred by means of a variational Bayesian approximation [7] with further extensions to online learning scenarios under data drift [18].

Broader impact

This paper stems from the author's work on time series forecasting and anomaly detection in industrial settings. The proposed methods are applicable to forecasting from univariate and multivariate data streams more generally. Business applications include supply chain monitoring and sales prediction. Furthermore, accurate forecasts allows better resource management, such as waste reduction and optimisation of energy consumption.

Funding disclosure

This work was funded by Amazon Research.

References

[1] G Ackerson and K Fu. On state estimation in switching environments. *IEEE Transactions on Automatic Control*, 15:10–17, 1970.

[2] I. Arasaratnam and S. Haykin. Cubature kalman filters. *IEEE Transactions on Automatic Control*, 54:1254–1269, 2009.

[3] I. Arasaratnam, S. Haykin, and R. J. Elliott. Discrete-time nonlinear filtering algorithms using gauss–hermite quadrature. *Proceedings of the IEEE*, 95:953–977, 2007.

[4] Yaakov Bar-Shalom and Xiao-Rong Li. *Estimation and Tracking: Principles, Techniques, and Software*. Artech House, 1993.

[5] David Barber. Expectation correction for smoothed inference in switching linear dynamical systems. *J. Mach. Learn. Res.*, 7:2515–2540, 2006.

[6] Philip Becker-Ehmck, Jan Peters, and Patrick Van Der Smagt. Switching linear dynamics for variational Bayes filtering. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 553–562. PMLR, 2019.

[7] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. volume 37 of *Proceedings of Machine Learning Research*, pages 1613–1622. PMLR, 2015.

[8] J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for nonlinear problems. *IEE Proceedings - Radar, Sonar and Navigation*, 146:2–7, 1999.

[9] Pierre Del Moral. *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems With Applications*. 2004.

[10] Arnaud Doucet, Nando de Freitas, Kevin P. Murphy, and Stuart J. Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, UAI '00, page 176–183. Morgan Kaufmann Publishers Inc., 2000.

[11] Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3601–3610. Curran Associates, Inc., 2017.

[12] Zoubin Ghahramani and Geoffrey E. Hinton. Variational learning for switching state-space models. *Neural Computation*, 12:831–864, 2000.

[13] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F (Radar and Signal Processing)*, 140:107–113(6), 1993.

[14] Rob Hyndman, Anne B Koehler, J Keith Ord, and Ralph D Snyder. *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media, 2008.

[15] Maximilian Karl, Maximilian Sölch, Justin Bayer, and Patrick van der Smagt. Deep variational bayes filters: Unsupervised learning of state space models from raw data. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

[16] Genshiro Kitagawa. Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5:1–25, 1996.

[17] Rahul G. Krishnan, Uri Shalit, and David Sontag. Structured inference networks for nonlinear state space models. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, page 2101–2109. AAAI Press, 2017.

[18] Richard Kurle, Botond Cseke, Alexej Klushyn, P. V. D. Smagt, and Stephan Günnemann. Continual learning with bayesian neural networks for non-stationary data. In *ICLR*, 2020.

[19] Richard Kurle, Stephan Günnemann, and P. V. D. Smagt. Multi-source neural variational inference. In *AAAI*, 2019.

[20] Tuan Anh Le, Maximilian Igl, Tom Jin, Tom Rainforth, and Frank Wood. Auto-encoding sequential monte carlo. 2017.

[21] Scott Linderman, Matthew Johnson, Andrew Miller, Ryan Adams, David Blei, and Liam Paninski. Bayesian Learning and Inference in Recurrent Switching Linear Dynamical Systems. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54, pages 914–922. PMLR, 20–22 Apr 2017.

[22] F. Lindsten, P. Bunch, S. Särkkä, T. B. Schön, and S. J. Godsill. Rao-blackwellized particle smoothers for conditionally linear gaussian models. *IEEE Journal of Selected Topics in Signal Processing*, 10(2):353–365, 2016.

[23] Chris J Maddison, John Lawson, George Tucker, Nicolas Heess, Mohammad Norouzi, Andriy Mnih, Arnaud Doucet, and Yee Whye Teh. Filtering variational objectives. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6573–6583. Curran Associates, Inc., 2017.

[24] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *ArXiv*, 2016.

[25] James E. Matheson and Robert L. Winkler. Scoring rules for continuous probability distributions. *Management Science*, 22:1087–1096, 1976.

[26] P. S. Maybeck. *Stochastic Models, Estimation and Control*. Mathematics in science and engineering. Academic Press, 1982.

[27] Thomas Peter Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

[28] Andriy Mnih and Danilo J. Rezende. Variational inference for monte carlo objectives. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2188–2196. JMLR.org, 2016.

[29] Christian A. Naesseth, Scott W. Linderman, Rajesh Ranganath, and David M. Blei. Variational sequential Monte Carlo. In *21st International Conference on Artificial Intelligence and Statistics (AISTATS 2018)*, pages 968–977, January 2018.

[30] Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7785–7794. Curran Associates, Inc., 2018.

[31] Sam Roweis and Zoubin Ghahramani. A unifying review of linear Gaussian models. *Neural Computation*, 11:305–345, 1999.

[32] David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processes. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 6827–6837. Curran Associates, Inc., 2019.

[33] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 2019.

[34] Matthias Seeger, Syama Rangapuram, Yuyang Wang, David Salinas, Jan Gasthaus, Tim Januschowski, and Valentin Flunkert. Approximate Bayesian inference in linear state space models for intermittent demand forecasting at scale. *arXiv preprint arXiv:1709.07638*, 2017.

[35] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3745–3753. Curran Associates Inc., 2016.

[36] E. A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158, 2000.

[37] Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2746–2754. Curran Associates, Inc., 2015.

[38] Mike West and Jeff Harrison. *Bayesian Forecasting and Dynamic Models (2nd Ed.)*. Springer, 1997.

[39] Onno Zoeter and Tom Heskes. Change point problems in linear dynamical systems. *J. Mach. Learn. Res.*, 6:1999–2026, 2005.

7 Supplementary Material

7.1 Conditional Kalman Filter equations

As mentioned in Sec. 2.3, the first term in Eq. (5) can be computed using the standard Kalman smoother. In this work, we are interested mainly in forecasting and parameter estimation (Sec. 3.3.2). Thus, the filter distribution $p(\mathbf{x}_t \mid \mathbf{s}_{1:t}^{(p)}, \mathbf{y}_{1:t-1})$ suffices, however the (conditionally) Gaussian joint distribution $p(\mathbf{x}_{1:t} \mid \mathbf{s}_{1:t}^{(p)}, \mathbf{y}_{1:t-1})$ could be computed straightforwardly using a Kalman smoother (e.g. running the RTS smoother backwards). Here we denote the location and covariance parameters of the Gaussian distributions corresponding to the prediction step as $m_{t|t-1}, V_{t|t-1}$, the predictive distribution (wrt. targets $\mathbf{y}_t$) as $m_{t|t}, V_{t|t}$, and the update (measurement) step as m_t, V_t.

The *prediction step* is given as

$$\begin{aligned} p(\mathbf{x}_t \mid \mathbf{s}_{1:t}^{(p)}, \mathbf{y}_{1:t-1}) &= \int p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{s}_t^{(p)})\; p(\mathbf{x}_{t-1} \mid \mathbf{s}_{1:t-1}^{(p)}, \mathbf{y}_{1:t-1})\; d\mathbf{x}_{t-1} \\ &= \mathcal{N}\big(\mathbf{x}_t; m_{t|t-1}(\mathbf{s}_{1:t}^{(p)}), V_{t|t-1}(\mathbf{s}_{1:t}^{(p)})\big), \end{aligned} \tag{17}$$

where

$$\begin{aligned} m_{t|t-1}(\mathbf{s}_{1:t}^{(p)}) &= A(\mathbf{s}_t^{(p)})\, m_{t-1}(\mathbf{s}_{1:t-1}^{(p)}) + B(\mathbf{s}_t^{(p)})\, \mathbf{u}_t, \\ V_{t|t-1}(\mathbf{s}_{1:t}^{(p)}) &= A(\mathbf{s}_t^{(p)}) V_{t-1}(\mathbf{s}_{1:t-1}^{(p)}) A(\mathbf{s}_t^{(p)})^T + R(\mathbf{s}_t^{(p)}). \end{aligned}$$

Similarly, the *predictive distribution* (used in the *update step* below) is given as

$$\begin{aligned} p(\mathbf{y}_t \mid \mathbf{s}_{1:t}^{(p)}, \mathbf{y}_{1:t-1}) &= \int p(\mathbf{y}_t \mid \mathbf{x}_t, \mathbf{s}_t^{(p)}) p(\mathbf{x}_t \mid \mathbf{s}_{1:t}^{(p)}, \mathbf{y}_{1:t-1}) d\mathbf{x}_t \\ &= \mathcal{N}\big(\mathbf{y}_t; m_{t|t}(\mathbf{s}_{1:t}^{(p)}), V_{t|t}(\mathbf{s}_{1:t}^{(p)})\big), \end{aligned} \tag{18}$$

where

$$\begin{aligned} m_{t|t}(\mathbf{s}_{1:t}^{(p)}) &= C(\mathbf{s}_t^{(p)})\, m_{t|t-1}(\mathbf{s}_{1:t}^{(p)}) + D(\mathbf{s}_t^{(p)})\, \mathbf{u}_t, \\ V_{t|t}(\mathbf{s}_{1:t}^{(p)}) &= C(\mathbf{s}_t^{(p)}) V_{t|t-1}(\mathbf{s}_{1:t}^{(p)}) C(\mathbf{s}_t^{(p)})^T + Q(\mathbf{s}_t^{(p)}). \end{aligned}$$

And the *update step* yields

$$\begin{aligned} p(\mathbf{x}_t \mid \mathbf{s}_{1:t}^{(p)}, \mathbf{y}_{1:t}) &= \frac{1}{Z}\, p(\mathbf{y}_t \mid \mathbf{x}_t, \mathbf{s}_t)\; p(\mathbf{x}_t \mid \mathbf{s}_{1:t}^{(p)}, \mathbf{y}_{1:t-1}) \\ &= \mathcal{N}\big(\mathbf{x}_t; m_t(\mathbf{s}_{1:t}^{(p)}), V_t(\mathbf{s}_{1:t}^{(p)})\big), \end{aligned} \tag{19}$$

where Z is the normalisation constant and where

$$\begin{aligned} m_t(\mathbf{s}_{1:t}^{(p)}) &= m_{t|t-1}(\mathbf{s}_{1:t}^{(p)}) + V_{t|t-1}^T(\mathbf{s}_{1:t}^{(p)}) C^T(\mathbf{s}_t^{(p)}) V_{t|t}^{-1}(\mathbf{s}_{1:t}^{(p)}) \big(\mathbf{y}_t - m_{t|t}(\mathbf{s}_{1:t}^{(p)})\big), \\ V_t(\mathbf{s}_{1:t}^{(p)}) &= V_{t|t-1}(\mathbf{s}_{1:t}^{(p)}) - V_{t|t-1}^T(\mathbf{s}_{1:t}^{(p)}) C^T(\mathbf{s}_t^{(p)}) V_{t|t}^{-1}(\mathbf{s}_{1:t}^{(p)}) C(\mathbf{s}_t^{(p)}) V_{t|t-1}(\mathbf{s}_{1:t}^{(p)}). \end{aligned}$$

7.2 SMC marginal likelihood estimate

SMC provides unbiased estimates of the marginal likelihood $p(\mathbf{y}_{1:T}) = \prod_{t=1}^T p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1})$ as a by-product [9] that can be used for learning (cf. Sec. 3.3.2). The conditionals $p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1})$ can be estimated by

$$\begin{aligned} p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1}) &= \int\int p(\mathbf{s}_{1:t-1} \mid \mathbf{y}_{1:t-1}) p(\mathbf{s}_t, \mathbf{y}_t \mid \mathbf{y}_{1:t-1}, \mathbf{s}_{1:t-1})\, d\mathbf{s}_t d\mathbf{s}_{1:t-1} \\ &= \int p(\mathbf{s}_{1:t-1} \mid \mathbf{y}_{1:t-1}) \int p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1}, \mathbf{s}_{1:t}) p(\mathbf{s}_t \mid \mathbf{s}_{t-1})\, d\mathbf{s}_t d\mathbf{s}_{1:t-1} \\ &= \int p(\mathbf{s}_{1:t-1} \mid \mathbf{y}_{1:t-1})\; \mathbb{E}_{\pi(\mathbf{s}_t \mid \mathbf{s}_{1:t-1})} \left[\frac{p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1}, \mathbf{s}_{1:t}) p(\mathbf{s}_t \mid \mathbf{s}_{1:t-1})}{\pi(\mathbf{s}_t \mid \mathbf{s}_{1:t-1})} \right] d\mathbf{s}_{1:t-1} \\ &\approx \sum_{p=1}^P w_{t-1}^{(p)} \frac{1}{N} \sum_{n=1}^N \gamma_t(\mathbf{s}_t^{(n,p)}, \mathbf{s}_{1:t-1}^{(p)}) \\ &\approx \sum_{p=1}^P w_{t-1}^{(p)} \gamma_t(\mathbf{s}_t^{(p)}, \mathbf{s}_{1:t-1}^{(p)}) = \sum_{p=1}^P \tilde{w}_t^{(p)}. \end{aligned}$$

The approximation of the inner importance-sampling expectation implies using a single sample (conditioned on $\mathbf{s}_{1:t-1}^{(p)}$) as is standard in SMC. The approximation wrt. the outer integral follows from $p(\mathbf{s}_{1:t-1} \mid \mathbf{y}_{1:t-1}) \approx \sum_{p=1}^{P} w_{t-1}^{(p)} \delta(\mathbf{s}_{1:t-1}^{(p)})$, using the normalised importance-weights of the previous step $w_{t-1}^{(p)} = {}^{w_{t-1}^{(p)}}/_{\sum_{p'=1}^{P} w_{t-1}^{(p')}}$.

Note that the above estimate of the conditional $p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1}) = {}^{p(\mathbf{y}_{1:t})}/_{p(\mathbf{y}_{1:t-1})}$ is a ratio estimate, since the previous importance-weights are normalized by the sum of importance-weights. Although these ratio estimates are in general not unbiased, the product $p(\mathbf{y}_{1:T}) \approx \prod_{t=1}^{T} \sum_{p=1}^{P} \tilde{w}_t^{(p)}$ yields an unbiased estimator of the marginal likelihood [9, 23].

7.3 Innovation State Space Model

Here we describe the structure of ISSM models used for `RSGLS-ISSM`. We start by simpler exemplar instantiations of the ISSM that use a *level*, *level-trend*, or *seasonality* component. In the ISSMs considered here, the emission and control matrices A, C are determined entirely by the assumed time series patterns. Only the diagonal state and observation noise covariances Q, R and optionally the control matrices B, D are learnable; these are computed as a weighted average of base matrices in our model. We will omit the optional matrices B, D in the following for simplification.

An ISSM with only a *level* component has just a single latent variable; $A = [1]$, $C = [1]$, and noise covariances $Q = [q_1], R = [r_1]$ are positive scalars. The latent state (level) evolves over time only through innovation with additive noise, and the innovation strength is given by the (square root of the) scalar covariance R.

An ISSM with *level-trend* components has a 2-dimensional latent state, one representing the level and the other representing the slope of a linear trend; the model is defined by

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 1 \end{bmatrix}, \quad R = \begin{bmatrix} r_1 & 0 \\ 0 & r_2 \end{bmatrix}, \quad Q = \begin{bmatrix} q_1 \end{bmatrix}.$$

Both level and trend component evolve over time through additive noise (with covariance R), and the level is additionally updated with the (previous) slope of the trend. The sum of the current level and trend components are emitted (with additive noise given by scalar covariance Q).

ISSMs with *seasonal component* can be instantiated in several ways. Here we use the same seasonality models as in [30]. These models are described by a set of seasonal factors that assume a certain periodicity. For example, day-of-week patterns use 7 factors, one for each day of the week; similarly, hour-of-day patterns use 24 factors. Each factor can be represented by one component of the latent state. For day-of-week seasonality, we thus have a 7-dimensional latent state. The transition matrix A is then the identity matrix, and the emission matrix $C = 1_{\{\mathrm{day}(t)=j\}_{j=1}^{7}}$ is an indicator (vector) that selects the component corresponding to the current day, zeroing out all other components. The noise covariance matrix Q is a (positive) scalar, and $R = \mathrm{diag}\big([r_1, \ldots, r_7] \odot 1_{\{\mathrm{day}(t)=j\}_{j=1}^{7}}\big)$ is a diagonal matrix, where all components except the one corresponding to the current day are zeroed out. This is done here through element-wise multiplication (of the diagonal) with the same indicator as used for the emission matrix.

As in [30], ISSMs with multiple *seasonal components* (corresponding to different periodicities) as well as *level* or *level-trend* can be combined. The resulting transition matrix A and noise covariance matrix R are block-diagonal, where each block corresponds to one component. Similarly, C is a concatenation of the corresponding components; consequently, the sum of the level, trend and each currently "active" seasonal component is emitted with additive noise.

In the experiments of this paper, we used a combination of *level component* and 1 or 2 *seasonal components* for the model variant `RSGLS-ISSM`: For data with daily measurement frequency (`wiki`, `exchange`), we used only day-of-week seasonality, resulting in a latent state with $7 + 1$ dimensions. In case of hourly data (`electricity`, `traffic`, `solar`), we used both hour-of-day and day-of-week seasonality, resulting in a latent state with $24 + 7 + 1$ dimensions.

7.4 Algorithm

The algorithm for Rao-Blackwellised particle filtering and loss computation for the ARSGLS proposed in Sec. 3 is presented in Alg. 1.

Algorithm 1 Rao-Blackwellised particle filter with parameter estimation.

Require: $\mathbf{s}_{t-1}$, $\mathbf{z}_{t-1}$, m_{t-1}, V_{t-1} are tensors with a particle, data, feature dimension. $\log \tilde{w}_{t-1}$ has particle, data dimensions. Data $\mathbf{u}_t$, $\mathbf{y}_t$ have data, feature dimensions. Tensors with counting indices, i.e. $\mathbf{s}_{1:T}$, have an additional time dimension that is indexed as e.g. $\mathbf{s}_t$.

function FILTER_LOSS($\mathbf{s}_{1:T}$, $\mathbf{z}_{1:T}$, $m_{1:T}$, $V_{1:T}$, $\mathbf{u}_{1:T}$, $\mathbf{y}_{1:T}$)
  $\log \tilde{w}_1, \mathbf{s}_1, \mathbf{z}_1, m_1, V_1 \leftarrow$ FILTER_STEP$(\mathbf{u}_1, \mathbf{y}_1)$
  for $t \leftarrow 2 \dots T$ **do**
    $\log \tilde{w}_t, \mathbf{s}_t, \mathbf{z}_t, m_t, V_t \leftarrow$ FILTER_STEP$(\log \tilde{w}_{t-1}, \mathbf{s}_{t-1}, \mathbf{z}_{t-1}, m_{t-1}, V_{t-1}, \mathbf{u}_t, \mathbf{y}_t)$
  end for
  $\mathcal{L} \leftarrow \text{compute\_marginal\_estimate}(\tilde{w}_{1:T})$ ▷ Eq. (13)
  return $\mathcal{L}$
end function

function FILTER_STEP($\log \tilde{w}_{t-1}$, $\mathbf{s}_{t-1}$, $\mathbf{z}_{t-1}$, m_{t-1}, V_{t-1}, $\mathbf{u}_t$, $\mathbf{y}_t$)
  initial_step $\leftarrow$ is_initial$(\log \tilde{w}_{t-1}, \mathbf{s}_{t-1}, \mathbf{z}_{t-1}, m_{t-1}, V_{t-1})$ ▷ if not provided (None)
  if initial_step **then**
    $\log w_t \leftarrow \log(1/P)$ ▷ uniform weights, P is the number of particles
    $m_{t-1}, V_{t-1} \leftarrow m_0, V_0$ ▷ initial state prior $p(\mathbf{x}_0)$
    $p(\mathbf{s}_t) \leftarrow \text{switch\_prior}(\mathbf{u}_t)$ ▷ initial switch prior $p(\mathbf{s}_1)$
  else
    $\log w_t \leftarrow \text{normalise}(\log \tilde{w}_t)$
    $\log w_t, \mathbf{s}_{t-1}, \mathbf{z}_{t-1}, m_{t-1}, V_{t-1} \leftarrow \text{resample}(\log w_t, \mathbf{s}_{t-1}, \mathbf{z}_{t-1}, m_{t-1}, V_{t-1})$
    $p(\mathbf{s}_t) \leftarrow$ SWITCH_RECURRENT_TRANSITION$(\mathbf{s}_{t-1}, m_{t-1}, V_{t-1})$ ▷ cf. below
  end if
  $q(\mathbf{s}_t), q(\mathbf{z}_t) \leftarrow \text{encoder}(\mathbf{y}_t, \mathbf{u}_t)$ ▷ Sec. 3.3.3
  $\pi(\mathbf{s}_t) \leftarrow p(\mathbf{s}_t) \times q(\mathbf{s}_t)$ ▷ Eq. (14)
  $\mathbf{s}_t \sim \pi(\mathbf{s}_t)$ ▷ sample switch particles
  $\psi_t \leftarrow \text{make\_base\_params}(\mathbf{s}_t, \mathbf{u}_t)$ ▷ base matrices A, B, C, D, Q, R
  $m_{t|t-1}, V_{t|t-1} \leftarrow \text{prediction\_step}(\mathrm{m_{t-1}}, \mathrm{V_{t-1}}, \psi_t)$ ▷ Eq. (17) in App. 7.1
  $m_{t|t}, V_{t|t} \leftarrow \text{auxiliary\_predictive}(m_{t|t-1}, V_{t|t-1}, \psi_t)$ ▷ Eq. (18) in App. 7.1
  $p(\mathbf{z}_t) \leftarrow \mathcal{N}(\mathbf{z}_t;\ m_{t|t}, V_{t|t})$
  $\pi(\mathbf{z}_t) \leftarrow p(\mathbf{z}_t) \times q(\mathbf{z}_t)$ ▷ Eq. (14)
  $\mathbf{z}_t \sim \pi(\mathbf{z}_t)$ ▷ sample auxiliary particles
  $m_t, V_t \leftarrow \text{update\_step}(\mathbf{y}_t, m_{t|t}, V_{t|t}, \psi_t)$ ▷ Eq. (19) in App. 7.1
  $p(\mathbf{y}_t) \leftarrow \text{decoder}(\mathbf{z}_t)$ ▷ Sec. 3.2
  $\log \gamma_t \leftarrow \log p(\mathbf{y}_t) + \log p(\mathbf{z}_t) + \log p(\mathbf{s}_t) - \log q(\mathbf{z}_t) - \log q(\mathbf{s}_t)$ ▷ Eq. (12)
  $\log \tilde{w}_t \leftarrow \log w_t + \log \gamma_t$
  return $\log \tilde{w}_t$, $\mathbf{s}_t$, $\mathbf{z}_t$, m_t, V_t
end function

function SWITCH_RECURRENT_TRANSITION($\mathbf{s}_{t-1}, m_{t-1}, V_{t-1}$)
  $F_t, S_t \leftarrow \text{make\_recurrent\_base\_params}(\mathbf{s}_{t-1})$ ▷ cond. linear state-to-switch transition
  $m_s, V_s \leftarrow \text{marginalise\_state}(m_{t-1}, V_{t-1}, F_t, S_t)$ ▷ state-to-switch prediction step
  $p_{\mathbf{s}|\mathbf{x}} \leftarrow \mathcal{N}(\mathbf{s}_t;\ m_s, V_s)$ ▷ Gaussian state-to-switch transition
  $p_{\mathbf{s}|\mathbf{s}} \leftarrow \text{switch\_transition}(\mathbf{s}_{t-1}, \mathbf{u}_t)$ ▷ Gaussian switch-to-switch transition
  $p(\mathbf{s}_t) \leftarrow \text{gaussian\_linear\_combination}(p_{\mathbf{s}|\mathbf{x}},\ p_{\mathbf{s}|\mathbf{s}})$ ▷ sum of means and covariances
  return $p(\mathbf{s}_t)$
end function

7.5 Further results

Here we provide further results for `ARSGLS` on `wiki` and `Box` datasets, evaluating the impact of the encoder and the number of particles during training. Furthermore, we provide pixel accuracy results for the experiments described in Sec. 5.3. On `Box`, we run each experiment with 3 different seeds and for `wiki`, we run with 4 random seeds as in the main text. Note that for `wiki` we use a different initialisation scheme compared to the experiments in the main text (Xavier instead of the default in Pytorch 1.6), resulting in better scores.

7.5.1 Encoder and proposal distribution

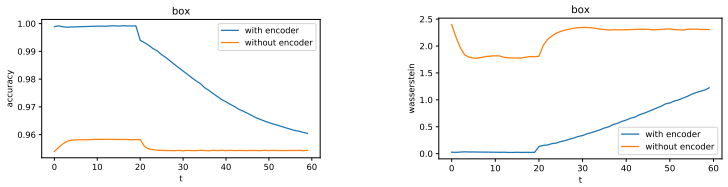


Figure 5: Pixel accuracy (left) and Wasserstein distance (right) on `Box` with/without encoder.

Here we show the importance of the encoder that approximates the likelihood function by a Gaussian (cf. Sec. 3.3.3). We conducted experiments where we omit the encoder and thus use a bootstrap proposal distribution. On `wiki`, using 32 particles during training without encoder yields CRPS scores 0.265 ± 0.010 for rolling evaluation and 0.360 ± 0.006 for non-rolling (long-term) evaluation, respectively. In contrast, the same number of particles with encoder yields significantly better CRPS scores of 0.207 ± 0.000 and 0.263 ± 0.003, respectively.

Similarly, the accuracy and Wasserstein distance for the `Box` dataset is significantly better when using our ladder-type encoder, as shown in Fig. 5. Without encoder for the proposal distribution, the model is not able to learn the dynamics and fails to converge to a reasonable fit.

7.6 Number of particles

We evaluate our model for a different numbers of particles $(1, 8, 16, 32, 64, 96)$ used during training. The performance on the `wiki` dataset is shown in Fig. 6. Surprisingly, the forecasting performance does not improve with more particles.

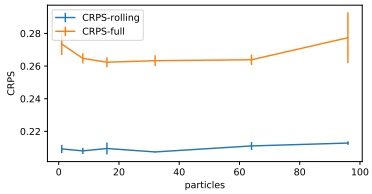


Figure 6: CRPS scores on `wiki` dataset for varying number of particles used for training.

In contrast, Fig. 7 shows that using more particles leads to a significantly better forecasting performance on the `Box` dataset as expected.

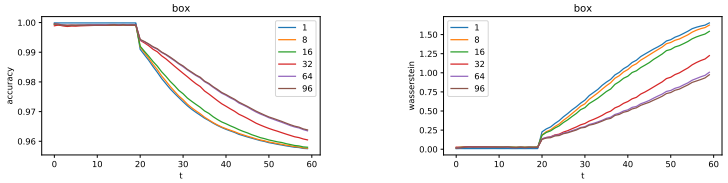


Figure 7: Accuracy (left), Wasserstein distance (right) on `Box` for varying number of particles used for training.

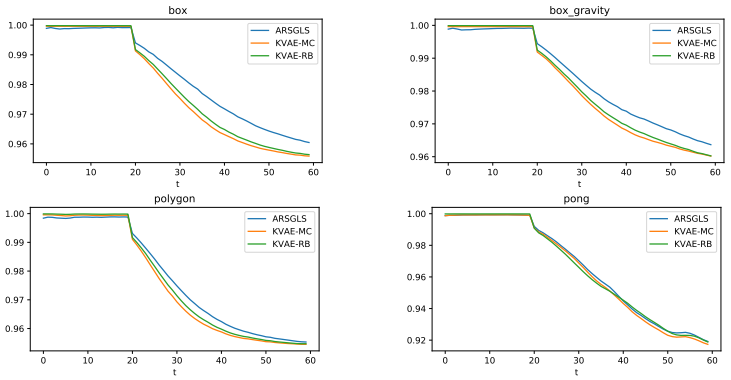


Figure 8: Pixel accuracies in filter/smoothing (`ARSGLS`/ `KVAE`) range ($t < 20$) and forecast range ($t \geq 20$). Results are averaged over 1000 test samples and 3 independent runs.

7.6.1 Simulated physical environments

Additional to the the Wasserstein distance reported in the main text, we provide pixel accuracies in Fig. 8

7.7 Experiment details

7.7.1 Pendulum

The model used in the pendulum experiment has 3 and 5 state and switch dimensions, respectively. We used 10 base matrices (for each of A, C, Q, R and additionally F, S in case of the recurrent model); the weights are predicted by an MLP with 1 hidden layer of 32 units with leaky relu activations (and a softmax output). All covariance matrices (R, Q, S) are diagonal and represented as log of the inverse scale to ensure positive variance. State and switch prior $p(\mathbf{x}_0)$, $p(\mathbf{s}_1)$ are both trainable diagonal Gaussians (scale parameters are again represented as log inverse scale), initialised as the standard Normal. The switch transition function $f(\mathbf{s}_{t-1})$ (cf. Eq.(9)) is an MLP with 1 hidden layer of 32 units and leaky relu activations; no encoder (cf. Sec.3.3.3) is used in this experiment.

Both models are trained using the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and (initial) learning rate 1×10^{-2} with an exponential decay to 1×10^{-4} over a total of 50 epochs. The batch size is 100, $P = 64$ particles are used for learning and $P = 100$ particles for computing the empirical mean and std. deviation of the GMM (cf. Eq. (8)) filter and forecast distribution in the evaluation plot.

7.7.2 Univariate time series forecasting

A short summary of the datasets, as used in [32], considered for these experiments are given:

- `electricity`: hourly electricity consumption of 370 customers;
- `traffic`: hourly occupancy rates of 963 car lanes of San Francisco bay area freeways;
- `solar`: hourly photo-voltaic production of 137 stations;
- `exchange`: daily exchange rate of 8 currencies;
- `wiki`: daily page view of 2000 Wikipedia pages.

For all datasets, the same model and training hyper-parameters (except learning rate) were used. Furthermore, `RSGLS-ISSM` and `ARSGLS` use the same model architecture where possible. For these datasets, each model are given time-features (hour of day, day of week, etc.) and time-series indicators as inputs $\mathbf{u}_{1:T}$ as in [30, 33]. Time-series indicators are embedded in 50 dimensions (except 8 dimensions for `exchange` dataset), and combined with the time-features by a single neural network layer with 64 units and leaky relu activations.

The state dimension of `RSGLS-ISSM` is determined by the ISSM structure (cf. App.7.3), whereas for `ARSGLS` the state has 16 dimensions. The switch $\mathbf{s}$ and auxiliary variable $\mathbf{z}$ (in case of `ARSGLS`) have 10 dimensions each. Furthermore, 20 base matrices are used. In case of `RSGLS` these include only

D, Q, R, F, S since A, C are determined by the ISSM. ARSGLS uses additionally learnable matrices A, C. For both models B was not used. The weights for averaging the matrices are predicted by an MLP with 1 hidden layer of 64 units with leaky relu activations and a softmax output (taking $\mathbf{s}_t$ as input). Covariance matrices (R, Q, S) are diagonal and parametrised as the log of the inverse scale.

The state prior $p(\mathbf{x}_0)$ is a trainable diagonal Gaussian, initialised as the standard Normal; the switch prior $p(\mathbf{s}_1 \mid \mathbf{u}_1)$ is a diagonal Gaussian for which the location and scale parameters are predicted by a linear transformation (and additional softplus in case of scale), taking the 64 input features from the embedding and subsequent neural network layer (see above) as inputs. Similarly, the switch transition function $f(\mathbf{s}_{t-1}, \mathbf{u}_t)$ (cf. Sec. 3.1) is an MLP with 1 hidden layer of 64 units and leaky relu activations, which takes both the 64 input features and the previous switch as inputs.

Encoders differ for RSGLS-ISSM and ARSGLS, respectively, since the latter has the additional auxiliary variable (cf. Sec. 3.2). However, in both cases the proposal distribution is formed as a product of Gaussians as described in Sec. 3.3.3. For RSGLS-ISSM, the encoder is a diagonal Gaussian for which the parameters are predicted by an MLP with 1 hidden layer of 64 units and leaky relu activations (outputs corresponding to the scale use softplus activations). The additional non-linear emission model of ARSGLS is a diagonal Gaussian for which the parameters are predicted by an MLP with 2 hidden layers with 64 units and leaky relu activations. The ladder encoder shares an MLP that predicts the parameters of the Gaussians corresponding to the auxiliary and switch variable, respectively. This shared MLP has 3 hidden layers with leaky relu activations; the parameters of the Gaussians are predicted from the 2nd and 3rd (last) hidden layer, respectively.

Each model is trained using the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$. For solar, the initial learning rate is 1×10^{-2}, for electricity 1×10^{-3} and for all other datasets 5×10^{-3}. In each experiment, the learning rate is decayed over a total of 2500 iterations by a factor 10^{-2}. The batch size is 50 and $P = 10$ particles are used.

7.7.3 Simulated physical environments

Model architectures (for components that are similar between both models) are chosen as in [11], except that the state dimension and number of base matrices is 10 for both, instead of 4 and 3, respectively. Furthermore, the (diagonal) covariance matrices R, Q (parameterised as log inverse scale, i.e. $\log R^{-1/2}$) are learnable, whereas in [11] these are fixed hyperparameters.

The switch in ARSGLS has 8 dimensions, and the switch prior is a trainable diagonal Gaussian, initialised as the standard Normal. The auxiliary variable has 6 dimensions for Pong (to encode the position of the ball and both pads) and 2 dimensions for all other datasets as in [11]. The switch transition function $f(\mathbf{s}_{t-1})$ from Eq. (9) is an MLP with 1 hidden layer of 64 units and relu activations. The ladder encoder of ARSGLS uses the same convolutional architecture for the auxiliary variable as the encoder in KVAE, and an additional hidden layer with 64 units for the switch. In contrast to the KVAE, the Gaussian from the encoder is combined with the respective Gaussian of the generative model as a product of these densities (cf. Sec. 3.3.3).

The same optimisation hyper-parameters are used for training as in [11], with the exception that we train each model for 400 epochs with an initial learning rate of 0.002, decaying every 20 epochs by 0.85 (instead of 80 epochs with initial learning rate 0.007 with the same decay). The number of particles in ARSGLS is $P = 32$.

Part III

Conclusion and Outlook

10 Summary

This cumulative dissertation has extended several Bayesian approximate-inference methods in the context of NNs. Part I provided an introduction and the fundamentals for this broad area of research that combines the principles of Bayesian inference with NN models. Two types of Bayesian deep learning methods have been distinguished: i) approximate inference of the NN weights (Ch. 3), and ii) methods that use deterministic NNs to parametrise the conditional distributions in DLVMs (Ch. 4) and DSSMs (Ch. 5) and the distributions of the corresponding inference models. The publications in Part II—which constitute the contributions of this cumulative dissertation—have shown that NN models can be successfully combined with variational inference algorithms to tackle different problem areas: for instance, continual learning for non-stationary data can be approached using variational BNNs (see Ch. 6), though problems due the detrimental effect of invariances in these models remain challenging (see Ch. 7). On the other hand, the latent variables in models of multiple modalities/views can be inferred in a principled manner (Ch. 8), and DSSMs can be trained successfully to provide competitive probabilistic long-term forecasts (Ch. 9). By building on the probabilistic framework, the main challenges arising in these domains are addressed in a principled manner through modelling assumptions and approximate inference, as summarised subsequently.

Ch. 6 addressed learning BNN parameters continually from a stream of datasets under the assumption that the marginal distribution of the data changes over time. The main challenge is how to retain previous inference results efficiently, while adapting to the changing data distribution, i.e. finding a trade-off between plasticity and stability, which is the ability to learn and retain knowledge, respectively. The proposed method provides a principled approach that explicitly defines the mechanism for forgetting and thus enables adaptation. The posterior approximation for the BNN model consists of two components: a Gaussian distribution and a complementary running memory, which are updated jointly as new data arrives by using a variational inference algorithm. Two alternative adaptation methods—which assume evolving NN weights or an exponential decay of the likelihood—enable adapting to distribution shifts. An advantage of the proposed framework is that novel time-varying priors corresponding to different drift assumptions (e.g. [141]) can be implemented.

The predominant and currently remaining downside of the above described continual-learning method is that variational Bayes for BNN often suffers from poor posterior approximations and even posterior collapse, where the posterior revert to the prior. The contribution in Ch. 7 explained an important factor contributing to this problem: invariances in the likelihood of over-parametrised models have a detrimental effect for variational inference. An important insight from this work is that certain invariances—such as translation and permutation invariance—leave the posterior predictive unchanged, but introduce an additional, non-constant gap in the ELBO objective. This invariance gap incentives posterior collapse, since the gap vanishes as

the posterior reverts to the prior. To show this, the detrimental effect of translation invariance for mean-field variational inference in an over-parametrised linear model has been studied. It turned out that ignoring the invariance indeed leads to posterior collapse, while correcting for the invariance gap in the ELBO optimisation achieves the true posterior.

In Ch. 8, amortised variational inference is extended to infer latent variables underlying data from different modalities or views (referred to as sources). The main assumption that each modality/view is generated from a shared latent state with independent noise is imposed straightforwardly by using distinct likelihood models (decoder) for each source and assuming conditional independence (given the state). Similarly, an important requirement is to also have individual inference models corresponding to each source. The proposed method addresses this requirement, noting that the factorisation of the generative model implicates that the optimal integration of the individual inference models is achieved by a product of experts. Having individual inference models has the advantage that missing sources are handled naturally by leaving out the corresponding inference model from the product of experts. Furthermore, it enables computing meaningful quantities between beliefs that relate these sources. To this end, a measure of conflict between the sources is proposed, and it is shown experimentally that it can be used to detect anomalies, such as failing sensors.

Ch. 9 addresses probabilistic forecasting, efficient inference, and parameter estimation in DSSMs. One appealing property that motivated using SSMs is that the latent variables (states) naturally encode the aleatoric uncertainty in the model's dynamics and forecast distribution. Building on amortised variational inference, both inference and likelihood estimation in DSSMs has high variance due to inefficient MC approximation. The proposed model builds on the classical SGLS, which allows for conditionally tractable inference. Two weaknesses of the classical SGLS are identified and addressed without compromising the conditional tractability: first, the model is extended by a state-to-switch recurrence that provides a closed-loop between states and switches; the importance of this dependency for accurate long-term forecasting of non-linear dynamical systems (e.g. pendulum dynamics) is shown in the paper. Second, a linear emission model is not suitable for highly-structured and high-dimensional data, such as images. This is addressed by introducing an auxiliary variable that decouples the conditionally linear SGLS from the observations and allows using a NN emission model that is suitable for the data. The main technical contribution of the paper is i) the extension of Rao-Blackwellised particle filtering from a classical SGLS to the proposed model, ii) the design of a suitable NN-based proposal distribution, and iii) the derivation of a variational SMC objective that can be successfully used for learning both model and variational parameters.

11 Future Research

This dissertation presented contributions in multiple, diverse research areas, building on variational-Bayesian methods for inference in probabilistic models. While concrete research directions for each of the individual methods are provided in the respective chapters in Part II, this section outlines promising directions for potential syntheses of these contributions and discusses important future directions on a wider scope.

The DSSM from Ch. 9 can be extended to include multi-modal/multi-view observations and missing data as considered in Ch. 8. The observations can be assumed conditionally independent given the auxiliary variable. Gaussian ladder-encoders for each source can be used to improve the proposal distribution. To this end, the different modalities can be combined with the sequential prior through a product-of-experts approach. This is feasible, since the prior distribution—resulting from the prediction step—and each of the likelihood approximations are Gaussian functions. Missing data can then be handled naturally by omitting the respective source, which corresponds to a non-informative (uniform) likelihood.

Both of the above discussed methods (from Ch. 9, and Ch. 8) can in principle be combined with the continual-learning method for non-stationary streaming data presented in Ch. 6. The continual-learning method uses a variational-Bayesian approach for the NN parameters, whereas the methods developed for DSSMs and DLVMs approximate the posterior of additional latent variables per observation. These can be combined by approximating the posterior of the parameters using variational Bayes (Sec. 3.4) and the latent variables using amortised/neural variational inference or variational SMC. Short-term temporal patterns are then modelled by the sequence model, while data drift over longer time periods (i.e. changing patterns of the time series) can be accounted for by the variational continual learning and adaptation methods proposed in Ch. 6. One possible challenge in combining the inference methods for parameters and latent variables is the increased computational and memory cost due to using independent sampling approaches for the two types of latent variables. Another challenge is the increased variance in the estimates of the objective function and its gradients. This is especially complicated for CNNs and sequence models, since the *local reparametrisation trick* is not applicable without further approximations.

Despite major efforts to scale variational BNNs to large model sizes and to modern NN architectures, these approaches still have not replaced deterministic NNs despite having only a minor increase in computational complexity. To further reap the benefits of variational BNNs for epistemic uncertainty estimation and continual learning under distribution shift, further progress must be made. In particular, it is essential to develop sound and provable/testable explanations for the performance difference between practical variational-Bayesian and point-estimation methods for NN models. Ch. 7 already made progress towards this ambitious goal with the

developed framework. However, while both permutation and translation invariance in BNNs have been fully characterised, concrete and tractable approximations of the corresponding invariance gap are yet to be developed. Similarly, this novel view on the underfitting problem of variational Bayes could inform alternative approximation methods that circumvent this problem, similarly to [142, 31, 143].

Diffusion probabilistic models [144] have recently gained substantial attention as a novel type of DLVM that are closely related to hierarchical VAEs [145, 146]. Both models optimise the variational ELBO objective, however, diffusion models use a much simpler inference model. These models have now established state-of-the-art performance in the computer-vision domain, achieving impressive results in density estimation, data compression, and image generation. Building on these advances in computer vision, it seems very promising to explore how to build on ideas and insights from these models in order to inform the development of novel time-series forecasting models, potentially leading to similar performance improvements. A particularly promising direction is to relate latent diffusion models (e.g. [147]) to DSSMs.

Bibliography

[1] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis. "Deep Learning for Computer Vision: A Brief Review". In: *Computational Intelligence and Neuroscience* 2018 (2018). Ed. by D. Andina, p. 7068349. ISSN: 1687-5265.

[2] D. W. Otter, J. R. Medina, and J. K. Kalita. "A Survey of the Usages of Deep Learning for Natural Language Processing". In: *IEEE Transactions on Neural Networks and Learning Systems* 32.2 (2021), pp. 604–624. DOI: 10.1109/TNNLS.2020.2979670.

[3] T. Lesort, V. Lomonaco, A. Stoian, D. Maltoni, D. Filliat, and N. Díaz-Rodríguez. "Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges". In: *Information Fusion* 58 (2020), pp. 52–68. ISSN: 1566-2535.

[4] S. Athey and G. W. Imbens. "Machine Learning Methods That Economists Should Know About". In: *Annual Review of Economics* 11.1 (2019), pp. 685–725.

[5] A. Rajkomar, J. Dean, and I. Kohane. "Machine Learning in Medicine". In: *New England Journal of Medicine* 380.14 (2019). PMID: 30943338, pp. 1347–1358.

[6] M. W. Libbrecht and W. S. Noble. "Machine learning applications in genetics and genomics". In: *Nature Reviews Genetics* 16.6 (2015), pp. 321–332. ISSN: 1471-0064.

[7] J. M. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Zídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. A. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis. "Highly accurate protein structure prediction with AlphaFold". In: *Nature* 596 (2021), pp. 583 –589.

[8] Z. Ghahramani. "Probabilistic machine learning and artificial intelligence". In: *Nat.* 521.7553 (2015), pp. 452–459.

[9] J. Schmidhuber. "Deep Learning in Neural Networks: An Overview". In: *Neural Networks* 61 (2015). Published online 2014; based on TR arXiv:1404.7828 [cs.NE], pp. 85–117. DOI: 10.1016/j.neunet.2014.09.003.

[10] Y. LeCun, Y. Bengio, and G. Hinton. "Deep learning". In: *Nature* 521.7553 (2015), pp. 436–444. ISSN: 1476-4687.

[11] Y. Gal. "Uncertainty in Deep Learning". PhD thesis. University of Cambridge, 2016.

[12] H. Wang and D.-Y. Yeung. "A Survey on Bayesian Deep Learning". In: *ACM Comput. Surv.* 53.5 (2020). ISSN: 0360-0300.

[13] A. G. Wilson. "The case for Bayesian deep learning". In: *arXiv preprint arXiv:2001.10995* (2020).

[14] Tishby, Levin, and Solla. "Consistent inference of probabilities in layered networks: predictions and generalizations". In: *International 1989 Joint Conference on Neural Networks*. 1989, 403–409 vol.2.

[15] W. L. Buntine and A. Weigend. "Bayesian Back-Propagation". In: *Complex Syst.* 5 (1991).

[16] J. S. Denker and Y. LeCun. "Transforming Neural-Net Output Levels to Probability Distributions". In: *Advances in Neural Information Processing Systems 3*. Ed. by R. P. Lippmann, J. E. Moody, and D. S. Touretzky. Morgan-Kaufmann, 1991, pp. 853–859.

[17] D. J. C. MacKay. "A Practical Bayesian Framework for Backpropagation Networks". In: *Neural Comput.* 4.3 (1992), 448–472. ISSN: 0899-7667.

[18] R. M. Neal. "Bayesian Learning via Stochastic Dynamics". In: *Advances in Neural Information Processing Systems 5*. Ed. by S. J. Hanson, J. D. Cowan, and C. L. Giles. Morgan-Kaufmann, 1993, pp. 475–482.

[19] G. E. Hinton and D. van Camp. "Keeping the Neural Networks Simple by Minimizing the Description Length of the Weights". In: *Proceedings of the Sixth Annual Conference on Computational Learning Theory*. COLT '93. Santa Cruz, California, USA: Association for Computing Machinery, 1993, 5–13. ISBN: 0897916115.

[20] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988. ISBN: 1558604790.

[21] R. Neal. "MCMC using Hamiltonian dynamics". In: *Handbook of Markov Chain Monte Carlo* (June 2012).

[22] M. Welling and Y. W. Teh. "Bayesian Learning via Stochastic Gradient Langevin Dynamics". In: ICML'11. Bellevue, Washington, USA: Omnipress, 2011, 681–688. ISBN: 9781450306195.

[23] A. Graves. "Practical Variational Inference for Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger. Vol. 24. Curran Associates, Inc., 2011, pp. 2348–2356.

[24] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. "Weight Uncertainty in Neural Networks". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, 2015, 1613–1622.

[25] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. "Stochastic Variational Inference". In: *Journal of Machine Learning Research* 14.4 (2013), pp. 1303–1347.

[26] D. P. Kingma and M. Welling. "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2014.

[27] D. J. Rezende, S. Mohamed, and D. Wierstra. "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by E. P. Xing and T. Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 2014, pp. 1278–1286.

[28] C. Louizos and M. Welling. "Multiplicative Normalizing Flows for Variational Bayesian Neural Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by D. Precup and Y. W. Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, 2017, pp. 2218–2227.

[29] G. Dikov and J. Bayer. "Bayesian Learning of Neural Network Architectures". In: *Proceedings of Machine Learning Research*. Ed. by K. Chaudhuri and M. Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, 2019, pp. 730–738.

[30] S. Ghosh, J. Yao, and F. Doshi-Velez. "Structured Variational Learning of Bayesian Neural Networks with Horseshoe Priors". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by J. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, 2018, pp. 1744–1753.

[31] S. Sun, G. Zhang, J. Shi, and R. Grosse. "Functional Variational Bayesian Neural Networks". In: *International Conference on Learning Representations*. 2019.

[32] A. Wu, S. Nowozin, E. Meeds, R. E. Turner, J. M. Hernández-Lobato, and A. L. Gaunt. "Deterministic Variational Inference for Robust Bayesian Neural Networks". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[33] Y. Burda, R. B. Grosse, and R. Salakhutdinov. "Importance Weighted Autoencoders". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2016.

[34] T. Salimans, D. P. Kingma, and M. Welling. "Markov Chain Monte Carlo and Variational Inference: Bridging the Gap". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, 2015, 1218–1226.

[35] D. Rezende and S. Mohamed. "Variational Inference with Normalizing Flows". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by F. Bach and D. Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, 2015, pp. 1530–1538.

[36] L. Mescheder, S. Nowozin, and A. Geiger. "Adversarial Variational Bayes: Unifying Variational Autoencoders and Generative Adversarial Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017.

[37] A. Klushyn, N. Chen, R. Kurle, B. Cseke, and P. van der Smagt. "Learning Hierarchical Priors in VAEs". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc., 2019, pp. 2870–2879.

[38] M. Karl, M. Sölch, J. Bayer, and P. van der Smagt. "Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

[39] M. Fraccaro, S. Kamronn, U. Paquet, and O. Winther. "A Disentangled Recognition and Nonlinear Dynamics Model for Unsupervised Learning". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017.

[40] C. J. Maddison, J. Lawson, G. Tucker, N. Heess, M. Norouzi, A. Mnih, A. Doucet, and Y. W. Teh. "Filtering Variational Objectives". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Curran Associates, Inc., 2017, pp. 6573–6583.

[41] A. Moretti, Z. zhao Wang, L. Wu, and I. Drori. "Variational Objectives for Markovian Dynamics with Backward Simulation". In: *ECAI*. 2020.

[42] Y. Wang, D. Blei, and J. P. Cunningham. "Posterior Collapse and Latent Variable Non-identifiability". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 5443–5455.

[43] R. Kurle, B. Cseke, A. Klushyn, P. van der Smagt, and S. Günnemann. "Continual Learning with Bayesian Neural Networks for Non-Stationary Data". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[44] R. Kurle, T. Januschowski, J. Gasthaus, and Y. Wang. "On Symmetries in Variational Bayesian Neural Nets". In: *Bayesian Deep Learning NeurIPS workshop*. 2021.

[45] R. Kurle, R. Herbrich, T. Januschowski, Y. Wang, and J. Gasthaus. "On the detrimental effect of invariances in the likelihood for variational inference". In: *Advances in Neural Information Processing Systems*. Vol. 35. Curran Associates, Inc., 2022.

[46] N. Chen, A. Klushyn, R. Kurle, X. Jiang, J. Bayer, and P. Smagt. "Metrics for Deep Generative Models". In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. Ed. by A. Storkey and F. Perez-Cruz. Vol. 84. Proceedings of Machine Learning Research. PMLR, 2018, pp. 1540–1550.

[47] R. Kurle and S. G. aßnd P. V. D. Smagt. "Multi-Source Neural Variational Inference". In: *AAAI*. 2019.

[48] E. de Bézenac, S. S. Rangapuram, K. Benidis, M. Bohlke-Schneider, R. Kurle, L. Stella, H. Hasson, P. Gallinari, and T. Januschowski. "Normalizing Kalman Filters for Multivariate Time Series Analysis". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 2995–3007.

[49] R. Kurle, S. S. Rangapuram, E. de Bézenac, S. Günnemann, and J. Gasthaus. "Deep Rao-Blackwellised Particle Filters for Time Series Forecasting". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 15371–15382.

[50] A. Klushyn, R. Kurle, M. Soelch, B. Cseke, and P. van der Smagt. "Latent Matters: Learning Deep State-Space Models". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 10234–10245.

[51] A. F. Ansari, K. Benidis, R. Kurle, A. C. Turkmen, H. Soh, A. Smola, B. Wang, and T. Januschowski. "Deep Explicit Duration Switching Models for Time Series". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 29949–29961.

[52] R. T. Cox. "Probability, Frequency and Reasonable Expectation". In: *American Journal of Physics* 14.1 (1946), pp. 1–13.

[53] D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Copyright Cambridge University Press, 2003.

[54] E. T. Jaynes. *Probability Theory: The Logic of Science*. Ed. by G. L. Bretthorst. Cambridge University Press, 2003.

[55] K. P. Murphy. *Probabilistic Machine Learning: An introduction*. MIT Press, 2022.

[56] Y. Gal. "Uncertainty in Deep Learning". PhD thesis. University of Cambridge, 2016.

[57] S. Depeweg. "Modeling Epistemic and Aleatoric Uncertainty with Bayesian Neural Networks and Latent Variables". Dissertation. München: Technische Universität München, 2019.

[58] P. Grünwald and T. van Ommen. "Inconsistency of Bayesian Inference for Misspecified Linear Models, and a Proposal for Repairing It". In: *Bayesian Anal.* 12.4 (2017), pp. 1069–1103.

[59] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. "Variational Inference: A Review for Statisticians". In: *Journal of the American Statistical Association* 112.518 (2017), pp. 859–877.

[60] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. "An Introduction to Variational Methods for Graphical Models". In: *Mach. Learn.* 37.2 (1999), 183–233. ISSN: 0885-6125.

[61] J. V. Jensen. "Sur les fonctions convexes et les inégalités entre les valeurs moyennes". In: *Acta Mathematica* 30 (), pp. 175–193.

[62] R. Ranganath, S. Gerrish, and D. Blei. "Black Box Variational Inference". In: *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*. Ed. by S. Kaski and J. Corander. Vol. 33. Proceedings of Machine Learning Research. Reykjavik, Iceland: PMLR, 2014, pp. 814–822.

[63] S. Mohamed, M. Rosca, M. Figurnov, and A. Mnih. "Monte Carlo Gradient Estimation in Machine Learning". In: *J. Mach. Learn. Res.* 21.1 (2020). ISSN: 1532-4435.

[64] R. J. Williams. "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning". In: *Mach. Learn.* 8.3–4 (1992), 229–256. ISSN: 0885-6125.

[65] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. "Policy Gradient Methods for Reinforcement Learning with Function Approximation". In: *Advances in Neural Information Processing Systems*. Ed. by S. Solla, T. Leen, and K. Müller. Vol. 12. MIT Press, 1999.

[66] J. Schulman. "Optimizing Expectations: From Deep Reinforcement Learning to Stochastic Computation Graphs". PhD thesis. EECS Department, University of California, Berkeley, 2016.

[67] F. J. R. Ruiz, M. K. Titsias, and D. M. Blei. "The Generalized Reparameterization Gradient". In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. Barcelona, Spain: Curran Associates Inc., 2016, 460–468. ISBN: 9781510838819.

[68] M. Figurnov, S. Mohamed, and A. Mnih. "Implicit Reparameterization Gradients". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. NIPS'18. Montréal, Canada: Curran Associates Inc., 2018, 439–450.

[69] C. J. Maddison, A. Mnih, and Y. W. Teh. "The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables". In: *International Conference on Learning Representations*. 2017.

[70] F. Nielsen and R. Nock. "Entropies and cross-entropies of exponential families". In: *2010 IEEE International Conference on Image Processing*. 2010, pp. 3621–3624.

[71] A. B. Owen. *Monte Carlo theory, methods and examples*. 2013.

[72] J. E. Matheson and R. L. Winkler. "Scoring Rules for Continuous Probability Distributions". In: *Management Science* 22.10 (1976), pp. 1087–1096. ISSN: 00251909, 15265501. (Visited on 07/14/2022).

[73] H. Hersbach. "Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems". In: *Weather and Forecasting - WEATHER FORECAST* 15 (Oct. 2000), pp. 559–570.

[74] T. Gneiting and A. E. Raftery. "Strictly Proper Scoring Rules, Prediction, and Estimation". In: *Journal of the American Statistical Association* 102.477 (2007), pp. 359–378.

[75] N. Gordon, D. Salmond, and A. Smith. "Novel approach to nonlinear/non-Gaussian Bayesian state estimation". English. In: *IEE Proceedings F (Radar and Signal Processing)* 140 (2 1993), 107–113(6).

[76] G. Kitagawa. "Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models". In: *Journal of Computational and Graphical Statistics* 5 (1996), pp. 1–25.

[77] A. Doucet and A. Johansen. "A Tutorial on Particle Filtering and Smoothing: Fifteen years later". In: 2008.

[78] C. A. Naesseth, F. Lindsten, and T. B. Schön. "Elements of Sequential Monte Carlo". In: *Found. Trends Mach. Learn.* 12 (2019), pp. 307–392.

[79] P. Del Moral. *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems With Applications*. 2004.

[80] N. Chopin. "Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference". In: *Ann. Statist.* 32.6 (2004), pp. 2385–2411.

[81] J. Carpenter, P. Clifford, and P. Fearnhead. "Improved particle filter for nonlinear problems". In: *IEE Proceedings - Radar, Sonar and Navigation* 146 (1999), pp. 2–7.

[82] K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[83] S. Hochreiter and J. Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[84] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014, pp. 1724–1734.

[85] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017, pp. 5998–6008.

[86] R. M. Neal. "Bayesian Learning for Neural Networks". PhD thesis. CAN, 1995. ISBN: 0612026760.

[87] J. F. G. De Freitas. "Bayesian methods for neural networks". PhD thesis. University of Cambridge, 2003.

[88] D. M. Titterington. "Bayesian Methods for Neural Networks and Related Models". In: *Statist. Sci.* 19.1 (2004), pp. 128–139.

[89] T. Chen, E. Fox, and C. Guestrin. "Stochastic Gradient Hamiltonian Monte Carlo". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by E. P. Xing and T. Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 2014, pp. 1683–1691.

[90] J. S. Denker and Y. LeCun. "Transforming Neural-Net Output Levels to Probability Distributions". In: *Advances in Neural Information Processing Systems 3*. Ed. by R. P. Lippmann, J. E. Moody, and D. S. Touretzky. Morgan-Kaufmann, 1991, pp. 853–859.

[91] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell. "Overcoming catastrophic forgetting in neural networks". In: *Proceedings of the National Academy of Sciences* 114 (2016).

[92] H Ritter, A Botev, and D Barber. "A Scalable Laplace Approximation for Neural Networks". In: 2018.

[93] N. Lawrence. "Variational inference in probabilistic models". In: (2000).

[94] A. Y. K. Foong, Y. Li, J. M. Hernández-Lobato, and R. Turner. "'In-Between' Uncertainty in Bayesian Neural Networks". In: *ArXiv* abs/1906.11537 (2019).

[95] D. P. Kingma, T. Salimans, and M. Welling. "Variational Dropout and the Local Reparameterization Trick". In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'15. Montreal, Canada: MIT Press, 2015, 2575–2583.

[96] M. B. Tomczak, S. Swaroop, and R. E. Turner. "Efficient Low Rank Gaussian Variational Inference for Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 33. 2020.

[97] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN: 0262018020.

[98] B. Fruchter. *Introduction to factor analysis.* Introduction to factor analysis. Oxford, England: Van Nostrand, 1954.

[99] H. H. Harman. *Modern factor analysis*. Modern factor analysis. Oxford, England: U Chicago Press, 1976.

[100] M. E. Tipping and C. M. Bishop. "Probabilistic Principal Component Analysis". In: *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B* 61.3 (1999), pp. 611–622.

[101] S. Roweis. "EM Algorithms for PCA and SPCA". In: *Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10*. NIPS '97. Denver, Colorado, USA: MIT Press, 1998, 626–632. ISBN: 0262100762.

[102] C. F. Beckmann and S. M. Smith. "Probabilistic independent component analysis for functional magnetic resonance imaging". In: *IEEE Transactions on Medical Imaging* 23.2 (2004), pp. 137–152.

[103] D. M. Blei, A. Y. Ng, and M. I. Jordan. "Latent Dirichlet Allocation". In: *J. Mach. Learn. Res.* 3.null (2003), 993–1022. ISSN: 1532-4435.

[104] R. M. Neal. "Connectionist Learning of Belief Networks". In: *Artif. Intell.* 56.1 (1992), 71–113. ISSN: 0004-3702.

[105] L. K. Saul, T. Jaakkola, and M. I. Jordan. "Mean Field Theory for Sigmoid Belief Networks". In: *J. Artif. Int. Res.* 4.1 (1996), 61–76. ISSN: 1076-9757.

[106] P. Dayan, G. E. Hinton, R. M. Neal, and R. S. Zemel. "The Helmholtz Machine". In: *Neural Comput.* 7.5 (1995), 889–904.

[107] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. "Ladder Variational Autoencoders". In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. Barcelona, Spain: Curran Associates Inc., 2016, 3745–3753. ISBN: 9781510838819.

[108] A. Vahdat and J. Kautz. "NVAE: A Deep Hierarchical Variational Autoencoder". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 19667–19679.

[109] R. Child. "Very Deep {VAE}s Generalize Autoregressive Models and Can Outperform Them on Images". In: *International Conference on Learning Representations*. 2021.

[110] K. Sohn, H. Lee, and X. Yan. "Learning Structured Output Representation using Deep Conditional Generative Models". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc., 2015, pp. 3483–3491.

[111] D. P. Kingma, S. Mohamed, D. Jimenez Rezende, and M. Welling. "Semi-supervised Learning with Deep Generative Models". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger. Vol. 27. Curran Associates, Inc., 2014, pp. 3581–3589.

[112] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum Likelihood from Incomplete Data via the EM Algorithm". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 39.1 (1977), pp. 1–38.

[113] M. J. Beal. "Variational Algorithms for Approximate Bayesian Inference". PhD thesis. Gatsby Computational Neuroscience Unit, University College London, 2003.

[114] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. "Improved Variational Inference with Inverse Autoregressive Flow". In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett. Vol. 29. Curran Associates, Inc., 2016, pp. 4743–4751.

[115] C.-W. Huang, D. Krueger, A. Lacoste, and A. Courville. "Neural Autoregressive Flows". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by J. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, 2018, pp. 2078–2087.

[116] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan. "Normalizing Flows for Probabilistic Modeling and Inference". In: *Journal of Machine Learning Research* 22.57 (2021), pp. 1–64.

[117] S. Särkkä. *Bayesian Filtering and Smoothing*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2013.

[118] G. Kitagawa. "The two-filter formula for smoothing and an implementation of the Gaussian-sum smoother". In: *Annals of the Institute of Statistical Mathematics* 46.4 (1994), pp. 605–623. ISSN: 1572-9052.

[119] M. Briers, A. Doucet, and S. Maskell. "Smoothing algorithms for state–space models". In: *Annals of the Institute of Statistical Mathematics* 62.1 (2009), p. 61. ISSN: 1572-9052.

[120] O. Zeitouni and A. Dembo. "Exact filters for the estimation of the number of transitions of finite-state continuous-time Markov processes". In: *IEEE Transactions on Information Theory* 34.4 (1988), pp. 890–893.

[121] P. Del Moral, A. Doucet, and S. Singh. "Forward Smoothing using Sequential Monte Carlo". In: (2010).

[122] O. Cappé. "Online EM Algorithm for Hidden Markov Models". In: *Journal of Computational and Graphical Statistics* 20.3 (2011), pp. 728–749. ISSN: 10618600.

[123] J. Olsson and J. Westerborn. "Efficient particle-based online smoothing in general hidden Markov models: The PaRIS algorithm". In: *Bernoulli* 23.3 (2017), pp. 1951–1996.

[124] O. Cappé, E. Moulines, and T. Ryden. *Inference in Hidden Markov Models (Springer Series in Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2005. ISBN: 0387402640.

[125] R. K. Olsson, K. B. Petersen, and T. Lehn-Schiøler. "State-Space Models: From the EM Algorithm to a Gradient Approach". In: *Neural Comput.* 19.4 (2007), 1097–1111. ISSN: 0899-7667.

[126] R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems". In: *Transactions of the ASME–Journal of Basic Engineering* 82.Series D (1960), pp. 35–45.

[127] H. E. RAUCH, F. TUNG, and C. T. STRIEBEL. "Maximum likelihood estimates of linear dynamic systems". In: *AIAA Journal* 3.8 (1965), pp. 1445–1450.

[128] A. Jazwinski. *Stochastic processes and filtering theory*. Mathematics in science and engineering 64. New York, NY [u.a.]: Acad. Press, 1970. XIV, 376. ISBN: 0123815509.

[129] P. S. Maybeck. *Stochastic Models, Estimation and Control*. Mathematics in science and engineering. Academic Press, 1982.

[130] M. Roth and F. Gustafsson. "An efficient implementation of the second order extended Kalman filter". In: *14th International Conference on Information Fusion*. 2011, pp. 1–6.

[131] T. P. Minka. "A family of algorithms for approximate Bayesian inference". PhD thesis. Massachusetts Institute of Technology, 2001.

[132] I. Arasaratnam and S. Haykin. "Cubature Kalman Filters". In: *IEEE Transactions on Automatic Control* 54 (2009), pp. 1254–1269.

[133] I. Arasaratnam, S. Haykin, and R. J. Elliott. "Discrete-Time Nonlinear Filtering Algorithms Using Gauss–Hermite Quadrature". In: *Proceedings of the IEEE* 95 (2007), pp. 953–977.

[134] E. A. Wan and R. Van Der Merwe. "The unscented Kalman filter for nonlinear estimation". In: *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*. 2000, pp. 153–158.

[135] M. Beal and Z. Ghahramani. "The variational Kalman smoother". In: 2001.

[136] V. Smidl and A. Quinn. "Variational Bayesian Filtering". In: *IEEE Transactions on Signal Processing* 56.10 (2008), pp. 5020–5030.

[137] S. J. Godsill, A. Doucet, and M. West. "Monte Carlo Smoothing for Nonlinear Time Series". In: *Journal of the American Statistical Association* 99.465 (2004), pp. 156–168.

[138] G. Poyiadjis, A. Doucet, and S. S. Singh. "Particle approximations of the score and observed information matrix in state space models with application to parameter estimation". In: *Biometrika* 98.1 (2011), pp. 65–80. ISSN: 0006-3444.

[139] C. Naesseth, S. Linderman, R. Ranganath, and D. Blei. "Variational Sequential Monte Carlo". English (US). In: *21st International Conference on Artificial Intelligence and Statistics (AISTATS 2018)*. 2018, pp. 968–977.

[140] T. A. Le, M. Igl, T. Jin, T. Rainforth, and F. Wood. "Auto-Encoding Sequential Monte Carlo". In: (2017).

[141] A. Li, A. Boyd, P. Smyth, and S. Mandt. "Detecting and Adapting to Irregular Distribution Shifts in Bayesian Online Learning". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 6816–6828.

[142] W. J. Maddox, T. Garipov, P. Izmailov, D. Vetrov, and A. G. Wilson. "A Simple Baseline for Bayesian Uncertainty in Deep Learning". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[143] E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, and P. Hennig. "Laplace Redux - Effortless Bayesian Deep Learning". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 20089–20103.

[144] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by F. Bach and D. Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, 2015, pp. 2256–2265.

[145] D. P. Kingma, T. Salimans, B. Poole, and J. Ho. "On Density Estimation with Diffusion Models". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. 2021.

[146] C.-W. Huang, J. H. Lim, and A. Courville. "A Variational Perspective on Diffusion-Based Generative Models and Score Matching". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. 2021.

[147] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10684–10695.