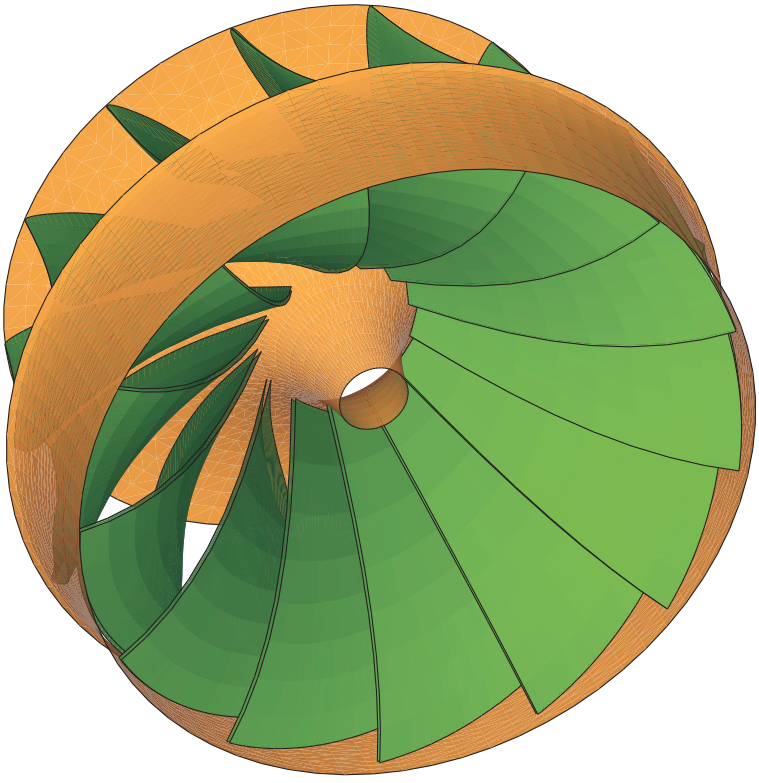
Technische Universität München
Institut für Energietechnik MW7

Lehrstuhl für Fluidmechanik

Entwurf von Beschaufelungen hydraulischer Maschinen mit Hilfe neuronaler Netze



Stefan Krämer

München, Februar 2006

Technische Universität München
Institut für Energietechnik MW7

Lehrstuhl für Fluidmechanik

Entwurf von Beschaufelungen hydraulischer Maschinen mit Hilfe neuronaler Netze

Stefan Krämer

Vollständiger Abdruck der von der Fakultät für Maschinenwesen der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr.-Ing., Dr.-Ing. habil. G. H. Schnerr
Prüfer der Dissertation:	
1.	Univ.-Prof. Dr.-Ing., Dr.-Ing. habil. R. Schilling
2.	Univ.-Prof. Dr.-Ing. H. Baier

Die Dissertation wurde am 21.11.2005 bei der Technischen Universität München eingereicht und durch die Fakultät für Maschinenwesen am 08.02.2006 angenommen.

Vorwort

Die vorliegende Dissertation entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Fluidmechanik der Technischen Universität München.

Mein besonderer Dank gilt Herrn Univ.-Prof. Dr.-Ing., Dr.-Ing. habil. Rudolf Schilling, der mir die Anfertigung dieser Arbeit ermöglichte und durch seine wissenschaftliche Anleitung und konstruktive Unterstützung sowie das entgegen gebrachte Vertrauen wesentlich zu ihrem Gelingen beigetragen hat. Seine wertvollen Hinweise sowie die lehrreichen Diskussionen waren mir stets eine große Hilfe während meiner Tätigkeit am Lehrstuhl.

Herrn Univ.-Prof. Dr.-Ing. Horst Baier danke ich für die Übernahme des Koreferates ebenso wie Herrn Univ.-Prof. Dr.-Ing., Dr.-Ing. habil. Günter H. Schnerr, der freundlicherweise als Vorsitzender der Prüfungskommission zur Verfügung stand.

Für die außerordentlich gute Zusammenarbeit gebührt meinen Kollegen, die mich in einer ausgezeichneten Arbeitsatmosphäre während der Arbeit begleitet und unterstützt haben, mein aufrichtiger Dank. Insbesondere seien hier meine Kollegen Thomas Lepach und Norbert Müller erwähnt.

Ganz herzlichen Dank möchte ich auch meinen Eltern aussprechen, die mich beim Erreichen meiner beruflichen Ziele immer bestärkt und nach Kräften unterstützt haben.

Meiner Frau Eva gilt mein besonderer Dank für die umfassende Unterstützung, die Geduld und Rücksichtnahme während dieser Zeit. Ihr und meinen zwei Kindern Elena und Linus ist diese Arbeit gewidmet.

München, Februar 2006

Stefan Krämer

Inhaltsverzeichnis

Verwendete Formelzeichen und Abkürzungen

Abkürzungen

3D	dreidimensional
AiF	Arbeitsgemeinschaft industrieller Forschungsvereinigungen „Otto von Guericke“ e.V.
AK	Austrittskante
CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
CI	Computational Intelligence
DNS	Direkte numerische Simulation
DS	Druckseite
EES	Echtzeit-Entwurfssystem
EK	Eintrittskante
EQ3D	Quasi-dreidimensionale Euler Strömungsrechnung
FLM	Lehrstuhl für Fluidmechanik
FLT	Forschungsvereinigung für Luft- und Trocknungstechnik
FT	Francis Turbine
GUI	Graphical User Interface
KI	Künstliche Intelligenz
KNN	Künstliches neuronales Netz
LDA	Laser Doppler Anemometrie
LES	Large Eddy Simulation
MINMOD	Verfahren nach Harten [18]
MPI	Message Passing Interface
MultiBPG	Multi-Backpropagation
NACA	National Advisory Committee for Aeronautics
NN	Neuronales Netz
NURBS	Non Uniform Rational Basis Splines
OSHER	Verfahren nach Chakravarthy und Osher [7]
PC	Personal Computer
PEES	Paralleles Echtzeit-Entwurfssystem
PVM	Parallel Virtual Machine
RTD	Real-Time-Designsystem

SS	Saugseite
UDS	Upwind Differencing Scheme
VES	Ventilator-Entwicklungssystem

Lateinische Zeichen

A	[m^2]	Fläche
BP		Betriebspunktparameter
C_p		Dimensionsloser Druck
C_μ		Modellkonstante
D		B-Spline Kontrollpunkt
D	[m]	Laufraddurchmesser
E		Ausgabefehler des neuronalen Netzes
GP		Geometrieparameter
H	[m]	Fallhöhe; Förderhöhe
L		normierte konforme Länge
M		B-Spline Basisfunktion
M		normierte Meridiankoordinate
N		B-Spline Basisfunktion; Anzahl Rechennetzpunkte
P		Punkt; B-Spline Kurvenpunkt
Q	[m^3/s]	Volumenstrom
R		normierte Ortskoordinate; normierter Radius
Re		Reynolds-Zahl
S		Splinekurve
S_{rot}	[m/s^2]	Rotationsbedingter Quellterm in den Impulsgleichungen
$S1$		Gitterebene
$S2$		Meridianebene
$S2_m$		repräsentative Meridianebene
$S3$		Normalebene
T	[s]	Zeitintervall
U		normierte Umfangskoordinate
V	[m^3]	Volumen
X		normierte Ortskoordinate
Y		normierte Ortskoordinate
Z		normierte Ortskoordinate
Z_{SL}		Anzahl Stromlinien
a		Splinekoeffizient; Neuronenaktivierung
a	[m]	Abstand
b		Splinekoeffizient
c		Splinekoeffizient
c	[m/s]	Absolutgeschwindigkeit
d		Splinekoeffizient; Weight Decay
d	[m]	Profildicke
ext_inp		Externer Input

f		Allgemeine Funktion
f_{act}		Aktivierungsfunktion
f_{out}		Ausgabefunktion
g	[m/s^2]	Erdbeschleunigung
hi		Laufindex Nabe (hub)
i		Erste Index-Richtung (Eintritt – Austritt); Laufindex
j		Zweite Index-Richtung (Druckseite – Saugseite); Laufindex
k		Dritte Index-Richtung (Deckscheibe – Nabe); Laufindex; B-Spline Ordnung
k	[m^2/s^2]	Turbulente kinetische Energie
l		B-Spline Ordnung
l	[m]	Konforme Länge; Schaufellänge
m		Zählvariable
m	[m]	Meridiankoordinate
n		Flächennormale; Zählvariable
n	[m]	Koordinate in Normalenrichtung
n	[1/min]	Drehzahl
n_q	[1/min]	Spezifische Drehzahl
net		Nettoinput
o		Neuronenausgabe
p		Laufindex Lernmuster
p	[N/m^2]	Druck
q		Quellterm; Zählvariable
r		Zählvariable
r	[m]	Ortskoordinate; Radius
s		Zählvariable; B-Spline Knoten
s	[m]	Koordinate entlang der Schaufelkante
si		Laufindex Deckscheibe (shroud)
t		B-Spline Knoten; B-Spline Kurvenparameter; Zielwert
t	[s]	Zeit
u		B-Spline Kurvenparameter
u	[m/s]	Umfangsgeschwindigkeit
u	[m]	Umfangskoordinate
u_τ	[m/s]	Schubspannungsgeschwindigkeit
v		B-Spline Kurvenparameter
w		Verbindungsgewicht
w	[m/s]	Relativgeschwindigkeit
x		Eingabewert des neuronalen Netzes
x	[m]	Ortskoordinate; Schaufellängenkoordinate
y		Ausgabewert des neuronalen Netzes
y	[m]	Ortskoordinate; Wandabstand
z		Schaufelzahl
z	[m]	Ortskoordinate

$\vec{D}$		Ortsvektor des B-Spline Kontrollpunktes
$\vec{P}$		Ortsvektor des B-Spline Kurvenpunktes
$\vec{S}$		B-Spline Knotenvektor
$\vec{S}_{rot}$	[m/s^2]	Vektor des rotationsbedingten Quellterms in den Impulsgleichungen
$\vec{T}$		B-Spline Knotenvektor
$\vec{c}$	[m/s]	Vektor der Absolutgeschwindigkeit
$\vec{r}$	[m]	Ortsvektor
$\vec{w}$	[m/s]	Vektor der Relativgeschwindigkeit
$\vec{x}$		Eingabevektor des neuronalen Netzes
$\vec{y}$		Ausgabevektor des neuronalen Netzes
$\mathbf{C}$		B-Spline Koeffizientenmatrix
$\mathbf{D}$		B-Spline Kontrollpunktmatrix
$\mathbf{P}$		B-Spline Kurvenpunktmatrix
$\mathbf{W}$		Gewichtsmatrix

Griechische Zeichen

Γ		Allgemeiner Diffusionskoeffizient
Δ		Differenz
Ψ		Druckzahl
α		Momentum
β		Blendingfaktor
β_S	[°]	Schaufelwinkel
γ	[°]	Winkelparameter
δ		Fehlersignal
δ	[m]	Kreisdurchmesser; Schichtdicke
ϵ	[m^2/s^3]	Dissipationsrate
η		Lernrate
η	[%]	Wirkungsgrad
θ		Schwellwert
μ		Streckenverhältnis
μ	[kg/(m·s)]	Dynamische Viskosität
ν	[m^2/s]	Kinematische Viskosität
ν_T	[m^2/s]	Kinematische Wirbelviskosität
ρ	[kg/m^3]	Dichte
τ_w	[N/m^2]	Wandschubspannung
ϕ		Allgemeine Variable
φ		Volumenzahl
φ	[°]	Umfangskoordinate
ω	[1/s]	Winkelgeschwindigkeit
$\vec{\omega}$	[1/s]	Winkelgeschwindigkeitsvektor

Tiefgestellte Zeichen

0	Index; Bilanzierungsebene vor dem Stufenelement; Referenz
1	Index; Schaufeleintrittskante
2	Index; Schaufelaustrittskante
3	Index; Bilanzierungsebene nach dem Stufenelement
4	Index
5	Index
A	Parameterbezeichnung
AK	Austrittskante
B	Parameterbezeichnung
C	Parameterbezeichnung
D	Laufraddurchmesser; Parameterbezeichnung
EK	Eintrittskante
HK	Hinterkante
La	Laufrad
LE	Eintrittskante
Le	Leitrad
M	Parameterbezeichnung
Pu	Pumpe
T	Turbulent
TE	Austrittskante
Tu	Turbine
a	Außen (Deckscheibe)
$conf$	Konform
g	Gesamt
ges	Gesamt
h	hydraulisch
hi	Laufindex Nabe (hub)
i	Erste Index-Richtung (Eintritt – Austritt); Laufindex; Innen (Nabe)
j	Zweite Index-Richtung (Druckseite – Saugseite); Laufindex
k	Dritte Index-Richtung (Deckscheibe – Nabe); Laufindex; B-Spline Ordnung
l	B-Spline Ordnung
m	Meridiankomponente; Zählvariable; Mitte
max	Maximalwert
min	Minimalwert
n	Normalkomponente; Zählvariable
opt	Optimum
p	Laufindex Lernmuster
ref	Referenz
si	Laufindex Deckscheibe (shroud)

t	total
th	theoretisch
u	Umfangskomponente
x	x-Komponente
y	y-Komponente
z	z-Komponente

Hochgestellte Zeichen

$+$	Dimensionslose turbulente Größe
$*$	Normierte Werte
T	Transponierte

Sonstige

$\frac{d\phi}{dt}$	Ableitung nach der Zeit
$\frac{\partial\phi}{\partial t}$	Partielle Ableitung nach der Zeit
$\overline{\phi}$	zeitlicher Mittelwert; Flächengewichteter Mittelwert; Massenstromgewichteter Mittelwert
ϕ'	Erste Ableitung; Schwankungsgröße; Hilfsvariable
ϕ''	Zweite Ableitung

Zusammenfassung

Der stetig steigende Wettbewerbsdruck zwingt die Hersteller von hydraulischen Maschinen zu immer kürzeren Entwicklungszeiten besserer und neuer Produkte. Im Rahmen dieser Arbeit wird ein neu entwickeltes Entwurfssystem auf der Basis künstlicher neuronaler Netze vorgestellt, mit dem in kürzester Zeit nahezu optimale Beschaufelungen hydraulischer Maschinen generiert werden können. Das System ermöglicht ferner die Aufbereitung und Parametrisierung vorhandener Geometrien von Francis Turbinen und Kreiselpumpen, die in Abhängigkeit von einer beliebigen Anzahl an Betriebspunktparametern mit dem integrierten Backpropagation Lernverfahren trainiert werden können. Zur Systematisierung der parametrisierten Geometrien stehen dem Anwender zwei interaktive Modifikationsbausteine zur Verfügung. Mit Hilfe der trainierten neuronalen Netze können anschließend neue Strömungsmaschinengeometrien unter Angabe eines beliebigen Betriebspunkts generiert werden, die vergleichbare strömungsmechanische Eigenschaften wie die Originalgeometrien aufweisen.

Der modulare Aufbau des Systems ermöglicht es zum einen, unterschiedliche Strömungsmaschinenbauarten und weitere Parametrisierungsstrategien problemlos zu integrieren. Zum anderen kann das System als grafische Benutzeroberfläche und Analysewerkzeug für den entwickelten Neuronale-Netze-Simulator zum Trainieren beliebiger tabellarisch aufbereiteter Daten verwendet werden.

Am Beispiel einer Baureihe von sechs optimierten Stufen von Francis Turbinen im spezifischen Drehzahlbereich $n_q = 20 - 120$ min^{-1} wird gezeigt, dass der verwendete Lernalgorithmus in der Lage ist, einen funktionalen Zusammenhang zwischen Betriebspunkt und Geometrie zu erkennen und zu lernen. Sowohl bei den approximierten als auch bei den interpolierten Geometrien aus dem neuronalen Netz konnten im Vergleich zu den Originalgeometrien hervorragende Strömungseigenschaften festgestellt werden.

Auf die gleiche Weise wird auch eine Kreiselpumpenbaureihe von sechs Erstentwurfsgeometrien im spezifischen Drehzahlbereich $n_q = 10 - 90$ min^{-1} parametrisiert und trainiert. Bei der anschließenden numerischen Simulation konnte sogar eine Verbesserung des Strömungsverhaltens bei fast allen aus dem neuronalen Netz generierten Geometrien im Vergleich zu den Originalgeometrien festgestellt werden.

Als Ergebnis dieser Arbeit zeigt sich, dass der Einsatz künstlicher neuronaler Netze eine erhebliche Zeitersparnis im Entwurfs- und Optimierungsprozess neuer Strömungsmaschinen bedeutet.

Kapitel 1

Einleitung

1.1 Problemstellung

Die Forderung nach immer kürzer werdenden Entwicklungszeiten bei steigendem Anspruch an die Qualität sowie die konkurrenzbedingte Notwendigkeit, die Kosten zu reduzieren, zwingt die Industrie, die Anzahl aufwendiger experimenteller Untersuchungen zu reduzieren und den Entwicklungsprozess in zunehmendem Maße mit Hilfe von rechnergestützten Berechnungsverfahren zu realisieren. Dieser Trend wird durch die in den letzten Jahren enorm gestiegene und immer kostengünstigere Rechnerleistung begünstigt. Gerade im Turbomaschinenbau hat der Einsatz numerischer Rechenverfahren zur Simulation von Strömungen, sog. CFD[1]-Codes, eine immer größere Bedeutung erlangt. Die Fortschritte auf diesem Gebiet ermöglichen es heute, auch komplexe Strömungsvorgänge auf einem handelsüblichen Personalcomputer zu berechnen.

Die Integration von CFD-Codes in problemorientierte Entwicklungssysteme erlauben es dem Ingenieur, in kürzester Zeit neue Entwürfe zu generieren, diese interaktiv zu modifizieren, nachzurechnen und zu bewerten, s. Schilling [44, 45], Watzelt et al. [62], Sporer et al. [56] und Vu et al. [60]. Mit Hilfe dieser Systeme ist es heute möglich, Spitzenwirkungsgrade im Wasserturbinenbau von mehr als 96% zu erreichen.

Um ein optimales Betriebsverhalten von Strömungsmaschinen zu erreichen, sind aufgrund der großen Anzahl an Freiheitsgraden zahlreiche Geometriemodifikationen mit nachfolgender Strömungsberechnung notwendig. Es ist daher nicht verwunderlich, dass sich die Hersteller immer stärker mit der Automatisierung dieser extrem zeitintensiven Optimierungsprozedur befassen. Erste Ansätze in der Entwicklung von numerischen Optimierungsverfahren wurden bereits in den Arbeiten von Fernández [12] und Aschenbrenner [2] beschrieben. Der hohe zeitliche Aufwand, vor allem bei der Zielfunktionsauswertung, stellte bei diesen Systemen jedoch immer noch ein großes Problem dar. Durch den Einsatz einer sog. Multi-Level-CFD Strategie, d.h. der Verwendung unterschiedlicher CFD-Verfahren in hierarchischer Folge

[1] CFD = Computational Fluid Dynamics

bezüglich ihrer Genauigkeit und damit ihrer Schnelligkeit, lässt sich der zeitliche Rahmen der Optimierung beträchtlich reduzieren, s. Wöhler [66], Thum und Schilling [58] und Schilling et al. [50].

Der Erfolg und die Dauer einer numerischen Optimierung hängt neben der Wahl geeigneter Optimierungsparameter entscheidend von der zu optimierenden Initialgeometrie ab. Je mehr die Initialgeometrie von der Optimalgeometrie abweicht, desto mehr Geometriemodifikationen sind notwendig, um das Optimum zu erreichen. Bis heute stützt sich jedoch der Erstentwurf von Strömungsmaschinen im Wesentlichen auf Erfahrungswerte, die aufgrund der großen Anzahl an Freiheitsgraden nur unzureichende systematische Eigenschaften aufweisen und somit in vielen Fällen nicht zu einer quasi optimalen Initialgeometrie führen. Es liegt deshalb nahe, diese Erfahrungswerte auf Regelmäßigkeiten zu analysieren, um einen systematischen bzw. funktionellen Zusammenhang zwischen Betriebspunkt und optimaler Geometrie zu erhalten. Mit Hilfe der ermittelten Abbildungsvorschrift ließe sich dann zu jedem beliebigen Betriebspunkt in kürzester Zeit eine der Systematik entsprechende optimale Geometrie berechnen. Eine solche Abbildungsfunktion wäre wiederum bedingt durch die zahlreichen freien Betriebspunkt- und Geometrieparameter viel zu komplex, um sie mit konventionellen Programmierverfahren in akzeptabler Zeit berechnen zu können. Zur Bewältigung dieser Aufgabe bieten sich aus dem Bereich der künstlichen Intelligenz die sog. künstlichen neuronalen Netze an.

1.2 Stand des Wissens

1.2.1 Anwendung von CFD-Verfahren im Strömungsmaschinenbau

Die industrielle Nutzung von CFD-Codes zur Simulation der Strömung beim Entwicklungsprozess von Turbomaschinen geht in die siebziger Jahre zurück. Die damals noch stark eingeschränkte Rechnerleistung ließ sinnvollerweise nur die Anwendung von vereinfachten Rechenmodellen zu.

Das Mittelschnittverfahren berechnete die Strömung nur in einer repräsentativen Meridianebene der Strömungsmaschine auf der Basis einer Stromfunktionsformulierung der strömungsmechanischen Grundgleichungen. Die Wirkung der Schaufelkräfte auf den Verlauf der Meridianströmung wurde zunächst unter der Annahme schaufelkongruenter Strömung ermittelt, s. Schilling [43]. Eine Verbesserung ließ sich erzielen, indem die Abweichung des Strömungswinkels von dem Schaufelwinkel bei der Bestimmung der Wirbelfunktion, die in die Berechnung der reibungsfreien, jedoch rotationsbehafteten Meridianströmung eingeht, berücksichtigt wurde, s. Müller [28].

Mit steigender Rechnerleistung konnten dann realitätsnähere Berechnungsmodelle verwendet werden, um so die komplexe, dreidimensionale Strömung durch Turbomaschinen näherungsweise berechnen zu können. Ein erster Ansatz zur Berechnung der reibungsfreien dreidimensionalen Strömung in Turbomaschinen basiert auf den

Arbeiten von Wu [67]. In der Modellvorstellung von Wu wird angenommen, dass die dreidimensionale Strömung auf beliebigen Stromflächen verläuft. Durch die Superposition von drei zweidimensionalen Strömungen in den Gitterebenen $S1_i$ und in den Meridianebenen $S2_j$ sowie in den Normalebenen $S3_k$ wird die reale Strömung approximiert. Zur Minimierung des Rechenaufwands wird in der vereinfachten Modellvorstellung von Wu von rotationssymmetrischen Stromflächen ausgegangen und unter Vernachlässigung der Strömung in den Normalebenen $S3_k$ eine repräsentative Meridianebene $S2_m$ in der Kanalmitte eingeführt, die mit den $S1_i$ Gitterebenen wechselseitig gekoppelt wird, s. Schilling [43], [44]. Diese sog. EQ3D-Verfahren sind wegen der geringen Rechenzeiten auch heute noch im interaktiven Entwurf von Beschaufelungen sehr verbreitet, s. Watzelt [61] und Schilling [46]. Allerdings werden Sekundärströmungseffekte in dieser Modellvorstellung a priori vernachlässigt.

Für eine voll dreidimensionale Simulation der inkompressiblen Strömung in Turbomaschinen ist die Lösung der strömungsmechanischen Erhaltungsgleichungen für die Masse und den Impuls im gesamten Lösungsgebiet notwendig. Dadurch erhält man Aufschluss über die 3D Struktur der Strömung einschließlich der Sekundärströmung sowie Aussagen über Strömungsverluste. Dank der heutigen Computertechnik ist die dreidimensionale Strömungssimulation von Standardproblemen auf handelsüblichen PC's in einem vertretbaren Zeitrahmen möglich.

Vor allem bei der Optimierung von Strömungsmaschinenbeschaufelungen werden zur Strömungsberechnung aus Zeitgründen noch 3D Eulerverfahren eingesetzt. Hierbei bleibt der Einfluss der Fluidviskosität auf die Strömung unberücksichtigt, wodurch sich der numerische Berechnungsaufwand verringert. Des Weiteren müssen Grenzschichtbereiche nicht mit hoher Punktedichte aufgelöst werden, wodurch die Dimension des Rechennetzes kleiner und somit der Speicherbedarf geringer wird. Die diskretisierten Gleichungen werden in der Regel mit Finite-Volumen-Verfahren oder Finite-Elemente-Verfahren gelöst. Für Strömungen mit hohen Reynolds-Zahlen, wie sie in hydraulischen Strömungsmaschinen üblicherweise vorkommen, liefern die 3D Euler-Verfahren im Betriebsbereich $Q/Q_{opt} \geq 0.7$ gute Ergebnisse, s. Schilling [48] und Riedel [36].

Die Nachrechnung bestehender Entwürfe von Strömungsmaschinenbeschaufelungen erfolgt hinsichtlich der Berechnung der Verluste mit Navier-Stokes-Verfahren. Die numerische Lösung der strömungsmechanischen Erhaltungsgleichungen enthält nun auch den Einfluss der Reibungskräfte. Für laminare Strömungen werden die originären Navier-Stokes Gleichungen gelöst, die sich nach dem Ansatz von Stokes aus den Impulsgleichungen ergeben, s. z.B. Schlichting [52]. Für turbulente Strömungen, wie sie beispielsweise in Turbomaschinen auftreten, können die augenblicklichen Strömungsgrößen nach dem Ansatz von Reynolds [33] in einen Mittelwert und einen Schwankungsanteil aufgeteilt werden. Nach Einführung dieser Formulierung in die Impulsgleichungen und einer zeitlichen Mittelung erhält man zusätzliche Reibungsterme, die scheinbaren Schubspannungen, die im Reynolds'schen Spannungstensor zusammengefasst werden. Durch den Informationsverlust infolge der zeitlichen Mittelwertbildung entsteht im Gleichungssystem ein Schließungsproblem, das durch die Einführung eines geeigneten Turbulenzmodells gelöst werden muss. Eine Übersicht verschiedener Turbulenzmodelle findet man z.B. bei Rodi und Berge-

LES [38], RODI [37] bzw. WILCOX [65].

Eine direkte numerische Simulation der turbulenten Strömung (DNS), bei der auch die kleinsten Wirbel berechnet werden, scheitert bei technisch relevanten Reynolds-Zahlen derzeit noch an dem enormen Speicher- und Rechenaufwand. Auch die Large Eddy Simulation (LES), bei der die kleinsten Wirbel durch einen Modellansatz beschrieben werden, ist im industriellen Bereich derzeit noch nicht zur Nachrechnung von Turbomaschinen geeignet. RODI [37] gibt einige Anhaltspunkte über die Rechenzeiten einfacher Testbeispiele. So ergeben sich bei der Large Eddy Simulation der Umströmung eines Würfels Berechnungszeiten, die um den Faktor 30 über den Berechnungszeiten mit einem Zwei-Gleichungs-Turbulenzmodell liegen.

Die Navier-Stokes Gleichungen werden ähnlich zur Vorgehensweise bei 3D Euler Verfahren mit Hilfe von Finite-Elemente bzw. Finite-Volumen-Verfahren gelöst. Einen Vergleich dreier kommerzieller Navier-Stokes Programme mit unterschiedlichen Lösungsansätzen und Turbulenzmodellen findet man bei SCHACHENMANN ET AL. [42]. Vergleiche der Rechenergebnisse mit LDA[2]-Messungen zeigen, dass das globale Verhalten der Strömung mit allen Verfahren gut wiedergegeben wird. Strömungen in komplexen Geometrien mit Drall und Ablösungen oder Rückströmungen können jedoch auch mit modernen Verfahren nur sehr unzuverlässig berechnet werden. Dies ist hauptsächlich auf die unzureichende Turbulenzmodellierung zurückzuführen. Ein Beispiel dafür ist die numerischen Simulation der Strömung durch das Ellbogensaugrohr einer Kaplan Vollspiralturbine, s. SCHILLING ET AL. [47] und BADER ET AL. [4].

Die korrekte Berechnung von Verlusten in Strömungsmaschinen basiert deshalb auf der sorgfältigen Kalibrierung von CFD-Verfahren mittels relativer Vergleiche der Berechnungsergebnisse mit Messdaten, wie sie z.B. bei SCHILLING ET AL. [49] dargestellt sind.

1.2.2 Entwurfs- und Optimierungssysteme

Der Entwurf von Strömungsmaschinen ist durch eine große Anzahl geometrischer und fluiddynamischer Freiheitsgrade gekennzeichnet. Ziel einer jeden Auslegung ist es, ein optimales Design zu finden, das zuvor definierten Randbedingungen genügt. Moderne Strömungsmaschinen zeichnen sich heute durch einen hohen Wirkungsgrad aus. Für das Erreichen einer weiteren Leistungssteigerung ist daher ein zunehmender technischer Aufwand nötig. Aus diesem Grunde wurden am Lehrstuhl für Fluidmechanik auch verstärkt Entwurfs- und Optimierungssysteme entwickelt.

EES (Echtzeit-Entwurfssystem)

Im Rahmen eines AiF-Forschungsprojekts wurde am Lehrstuhl für Fluidmechanik ein Entwurfssystem für Pumpenlaufräder entwickelt, s. SCHILLING ET AL. [51]. Bei dem im System implementierten CFD-Code handelt es sich um ein EQ3D-Verfahren, das nur die Schaufelskelettflächen einbezieht. Zusätzlich besteht die Möglichkeit, auf

[2]LDA = Laser Doppler Anemometrie

den berechneten Stromflächen nachträglich eine reine Gitterrechnung durchzuführen, um somit auch den Einfluss der Profilierung zu berücksichtigen. Das Entwicklungssystem ermöglicht die Modifikation sowohl axialer als auch radialer Laufräder, s. Sporer [55] und Watzelt [61].

PEES (Paralleles Echtzeit-Entwurfssystem)

Durch das von Watzelt [61] entwickelte Parallele Echtzeit-Entwurfssystem konnte der Entwurfsprozess weiter beschleunigt werden. Beim Einsatz des schnellen EQ3D-Verfahrens wurde die Berechnung der Strömung auf den $S1_i$ Stromflächen auf mehrere parallel arbeitende Prozessoren aufgeteilt. Zur Parallelisierung wurde die Parallelisierungssoftware PVM (Parallel Virtual Machine) eingesetzt, s. Geist et al. [15].

VES (Ventilator-Entwicklungssystem)

Im Auftrag der Forschungsvereinigung für Luft- und Trocknungstechnik (FLT) wurde ein Entwicklungssystem für Ventilatoren erstellt, das als Stufenentwurfssystem konzipiert wurde. Damit ist der Anwender in der Lage, beliebige Kombinationen radialer, axialer oder halbaxialer Lauf- und Leiträder zu entwerfen, zu modifizieren und als Stufe gekoppelt nachzurechnen. Im System sind drei unterschiedliche CFD-Codes implementiert. Für die schnelle Nachrechnung ist ein EQ3D-Verfahren unter Berücksichtigung profilierter Schaufeln integriert. Für die detaillierte 3D-Nachrechnung wurde das System zunächst um den von Riedel [35] entwickelten 3D Euler Stufen-Code erweitert, der mit der Software PVM parallelisiert wurde. Um eine realitätsnahe, reibungsbehaftete Strömungsberechnung zu ermöglichen, wurde in das System zusätzlich ein 3D-Navier-Stokes-Code implementiert, s. Bader [3]. Der mit der Software MPI (Message Passing Interface, s. [29]) parallelisierte Stufen-Code ist sowohl auf Strömungen inkompressibler als auch auf subsonische Strömungen kompressibler Fluide anwendbar.

Zur einfacheren Handhabung des Systems wurde eine Grafische Benutzerschnittstelle (GUI) implementiert, um Strömungsmaschinengeometrien interaktiv zu erstellen und zu modifizieren. Zur Auswertung der Strömungsergebnisse verfügt das System über ein umfangreiches grafisches Postprocessing.

RTD (Real-Time-Designsystem)

Um den Optimierungsprozess von Strömungsmaschinen zu beschleunigen, wurde am FLM ein System entwickelt, das neben einem Erstentwurfs- und Nachrechnungsbaustein ein umfangreiches Modifikationsmodul enthält, s. Richter [34] und Lepach [26]. Während in früheren Entwurfssystemen Modifikation und Nachrechnung unabhängig voneinander durchgeführt werden mussten, was einen erheblichen zeitlichen Aufwand bei der Optimierung zur Folge hatte, wurde die Strömungsberechnung beim RTD in den Modifikationsbaustein integriert. Ausgehend von kleinen geometrischen Veränderungen und der letzten bekannten Strömungslösung lassen sich mit heutiger Rechnerhardware auf Knopfdruck sehr kurze Antwortzeiten realisieren. Der gesamte Modifikations-, Nachrechnungs- und Auswertungsprozess wurde in eine einzige interaktive grafische Benutzeroberfläche integriert. Im Gegensatz zum VES ist

die gleichzeitige Modifikation aller Stufenelemente im Meridianschnitt möglich.

Das System ist als Stufenentwurfssystem sowohl zur Entwicklung von Pumpen und Ventilatoren als auch von Turbinen einsetzbar. Für die Nachrechnung sind drei komplett neu entwickelte CFD-Stufen-Codes implementiert worden, ein EQ3D-Code, ein 3D Euler- sowie ein 3D-Navier-Stokes-Verfahren. Da die Speicherallokierung in den neuen Codes dynamisch erfolgt, ist die Strömungsrechnung nicht mehr auf ein vorher festgelegtes maximales Rechengebiet beschränkt. Um bei komplexeren Stufenberechnungen, vor allem im Real-Time-Modifikationsmodul, kurze Antwortzeiten zu erreichen, ist das RTD für den Einsatz auf einem PC-Cluster konzipiert worden. Das Rechengebiet wird hierfür in Teilgebiete zerlegt (Domain Decomposition) und auf mehrere parallel arbeitende Rechner verteilt. Die Parallelisierung der 3D-Codes erfolgt mit Hilfe der Software MPI.

Für den Erstentwurf von Strömungsmaschinen wurden im RTD unterschiedliche Verfahren implementiert. Zum Erstellen der Meridiankonturen, bestehend aus Deckscheibe, Nabe, Schaufelein- und -austrittskante, steht einerseits ein einfach zu bedienendes CAD-Modul zur Verfügung, andererseits lassen sich für den Kreiselpumpenerstentwurf auch Meridiankonturen aus einer im System integrierten Statistik berechnen. Für den Laufschaufelentwurf stehen drei Methoden zur Verfügung. Zum Generieren von Kreiselpumpenlaufrädern wurde in das System ein nach der Theorie von Pfleiderer [31] arbeitendes Erstentwurfsmodul implementiert. Beim Entwurf von Turbinenbeschaufelungen wird im Gegensatz zur Beschaufelungen bei Pumpen meist eine Austrittsdrallverteilung vorgegeben, so dass in das System ein Modul integriert wurde, das in einem iterativen Verfahren eine passende Laufschaufel generiert. Im Falle axialer Maschinen lässt sich die Schaufelform im System nach der NACA-Philosophie berechnen.

Optimierungssysteme

Um den steigenden Anforderungen bezüglich des hydraulischen Wirkungsgrades gerecht werden zu können, ist der Einsatz numerischer Optimierungsverfahren erforderlich. Deshalb wurde am Lehrstuhl ein Optimierungssystem entwickelt, das auf dem Drei-Säulen-Konzept von Eschenauer et al. [10] aufbaut und sich somit innerhalb des Systems eine klare Trennung zwischen Entwurfsbaustein, CFD-Codes und Optimierungsalgorithmen ergibt. Der Anwender kann somit verschiedene Aufgabenstellungen definieren und die entsprechenden CFD-Codes bestimmen, mit denen die Optimierungsaufgabe gelöst werden soll, s. Fernández [12] und Aschenbrenner [2].

Da die verschiedenen Optimierungsverfahren eine Vielzahl, im Allgemeinen Hunderte von Iterationszyklen durchlaufen, ist die Verwendung von 3D-Navier-Stokes-Verfahren während des gesamten Optimierungsprozesses zu aufwändig. Aus diesem Grund wurde ein effizienteres halbautomatisches Optimierungssystem basierend auf einem Multi Level CFD-Ansatz entwickelt, s. Wöhler [66] und Thum und Schilling [58]. Je nach Optimierungsgrad werden unterschiedliche CFD-Codes zur Strömungssimulation eingesetzt. Zum Einstellen des Betriebspunktes der Maschine werden die schnellen EQ3D-Verfahren verwendet. Dieser Optimierung folgt in einem zweiten Schritt die Optimierung der Schaufeldruckverläufe mittels 3D Euler-

Verfahren. Die Feinoptimierung der Strömung und Minimierung der Verluste erfolgt im letzten Optimierungsschritt durch einen 3D-Navier-Stokes-Code. Als am besten geeigneter Optimierungsalgorithmus hat sich die Mustersuche nach HOOKE UND JEEVES [19] erwiesen.

1.2.3 Künstliche und berechenbare Intelligenz

Zunehmend stellt das menschliche Ingenium einen limitierenden Faktor bei der Handhabung der Komplexität von Strömungsmaschinen dar. So sind trotz der Vereinfachung durch die parametrisierte Darstellung der Geometrie, z.B. durch B-Splines, zahlreiche freie Parameter bei der Auslegung einer mehrstufigen Strömungsmaschine in Beziehung zueinander zu setzen und dabei Abhängigkeiten zu beachten. Daher liegt es nahe, Verfahren und Methoden zu entwickeln, die menschliche Intelligenz in den Rechner abbilden und somit zur Unterstützung im Auslegungsprozess von Strömungsmaschinen herangezogen werden können.

Seit der Entwicklung des ersten programmgesteuerten Rechners ist die Frage einer maschinellen Intelligenz von zentralem Interesse. Es haben sich hierbei zwei konkurrierende Paradigmen zur Erfassung von künstlicher Intelligenz (KI) gebildet, s. ZELL [68]:

Symbolische Informationsverarbeitung versucht, Wissen in Form von Regeln und Produktionssystemen in einen rechnerkonformen Formalismus abzubilden. Die Symbolmanipulation basiert dabei auf der philosophischen bzw. psychologischen Vorstellung menschlichen Denkens, wobei ausgehend von einzelnen Fakten und einem Regelwerk neues Wissen erfolgt.

Konnektionismus wird als der Ansatz verstanden, das menschliche Gehirn als Interaktion von vernetzten und parallel arbeitenden Neuronen nachzuvollziehen. Diese Simulation von Neuronen entspricht eher den Erkenntnissen der Neurowissenschaften aus Medizin und Biologie. Durch neue Erkenntnisse seit Mitte der achtziger Jahre gewinnen die sog. künstlichen neuronalen Netze zunehmende Bedeutung und stellen ein breites Forschungsgebiet unterschiedlicher Disziplinen dar.

Zur Diskussion, was künstliche Intelligenz ist und was diese zu leisten vermag sowie zu auftretenden Problemen, sei auf PENROSE [30] und GRAUBARD [17] hingewiesen. Ungeachtet der theoretischen und psychologischen Fragestellungen zur künstlichen Intelligenz hat sich seit Mitte der neunziger Jahre ein Forschungsfeld etabliert, das unter dem Begriff der berechenbaren Intelligenz (Computational Intelligence, CI) die folgenden drei Wissenszweige zusammenfasst und miteinander verbindet:

Künstliche neuronale Netze sind Abbildungen von Verbänden biologischer Nervenzellen und bilden informationsverarbeitende Einheiten, die vielfach untereinander vernetzt sind. Ein wesentliches Merkmal ist ihre Lernfähigkeit, d.h. die Fähigkeit, eine Aufgabe anhand zuvor trainierter Beispiele zu lösen.

Genetische Algorithmen und Evolutionsstrategien sind stochastische, d.h. dem Zufall unterliegende Optimierungsverfahren, die sich an den Prinzipien der na-

türlichen Evolution orientieren. Im Gegensatz zu klassischen Gradientenverfahren ist eine Bestimmung der Ableitung der Zielfunktion nicht nötig.

Fuzzy-Control Ansätze stellen den Versuch dar, Informationen auf einer symbolischen Ebene zu verarbeiten. Erfahrungen werden in Form von Wenn-Dann-Regeln festgelegt. Die mathematischen Grundlagen werden durch die Theorie der unscharfen Mengen bereitgestellt.

Methoden der berechenbaren Intelligenz orientieren sich insbesondere an praktischen Problemen. Diese stellen somit ein Betätigungsfeld für Informatiker und Ingenieure gleichermaßen dar.

In der vorliegenden Arbeit werden Verfahren künstlicher neuronaler Netze eingesetzt. Genetische Algorithmen und Fuzzy-Control Methoden werden hier nicht betrachtet und bleiben somit Forschungsgegenstand weiterer, zukünftiger Untersuchungen. Für eine Einführung in die Thematik von evolutionären Verfahren aus dem Bereich der Ingenieurwissenschaften sowie von Fuzzy-Logik bzw. Fuzzy-Control sei auf Goldberg [16] und Koch et al. [21] hingewiesen.

1.3 Zielsetzung

Die vorliegende Arbeit soll einen weiteren Beitrag zum Entwurf von Strömungsmaschinen leisten. Das Ziel ist es, ein interaktives grafisches Entwurfswerkzeug mit Verfahren zur Analyse und zur Systematisierung von radialen Strömungsmaschinengeometrien zu entwickeln. Ferner sollen die aufbereiteten Geometrien in Abhängigkeit des Betriebspunktverhaltens von einem künstlichen neuronalen Netz gelernt werden, um schließlich beliebige neue, der Systematik entsprechende Geometrien, generieren zu können. Zur Erstellung eines solchen Entwurfsbausteins sind vor allem bei der Aufbereitung der Geometrien für ein künstliches neuronales Netz mehrere Arbeitsschritte nötig, die an dieser Stelle kurz beschrieben werden sollen.

Vorbereitung

Ausgangsbasis für das erfolgreiche Trainieren eines künstlichen neuronalen Netzes ist das Vorhandensein einer umfangreichen, systematisch aufbereiteten Datenbank. Aufgrund der hohen Anzahl freier Parameter, die eine Strömungsmaschine charakterisieren, ist eine Beschränkung auf eine bestimmte Baugruppe unerlässlich. In dieser Arbeit sollen insgesamt zwei Strömungsmaschinentypen getrennt behandelt werden: Radial durchströmte Kreiselpumpen und Francis Turbinen. Hierbei ist zu überlegen, wie die vorhandenen Geometrien, abhängig von ihren Betriebspunkt- bzw. Strömungseigenschaften, aufzubereiten sind, um einerseits eine Reduzierung der freien Parameter zu erzielen und andererseits eine möglichst systematische Repräsentation der Geometrieparameter zu gewährleisten.

Geometrischer Erstentwurf

In einem ersten Schritt müssen für eine bestimmte Strömungsmaschinenbaureihe Geometrien in elektronischer Form generiert werden. Bei der Auslegung der Meridiankonturen fließen hierbei vor allem Erfahrungswerte ein. Die zugehörige Beschaufelung enthält man durch die oben beschriebenen Verfahren der lehrstuhleigenen Entwurfssysteme.

Parametrisierung und Systematisierung

Da sich die Approximation von Strömungsmaschinengeometrien mit Hilfe der B-Spline-Theorie in den am Lehrstuhl vorhandenen Entwurfssystemen vielfach bewährt hat, wird auf diese Theorie auch in dieser Arbeit zur Reduzierung der freien Parameter zurückgegriffen. Im Gegensatz zu zusammengesetzten Bézier-Kurven lassen sich vor allem Stetigkeitsbedingungen einfacher garantieren. Durch die B-Spline-Approximation vorhandener Meridiankonturen und Schaufelgeometrien ist es möglich, diese in eine einheitliche mathematische Form zu überführen. Die Kontrollpunkte der berechneten B-Spline-Kurven bilden dann zusammen mit den strömungsmechanischen Größen die Basis für eine Datenbank, die mit Hilfe künstlicher neuronaler Netze gelernt werden kann. Die Fähigkeit des zu entwickelnden Systems, beliebige Strömungsmaschinengeometrien durch B-Splines zu approximieren und somit zu parametrisieren, reicht aber bei weitem nicht aus, akzeptable Resultate beim Trainieren von neuronalen Netzen zu erzielen. Damit die eingesetzten Lernverfahren in den vorhandenen Trainingsdaten eine gewisse Systematik bzw. stetige Abhängigkeit finden können, muss in das System ein Analyse- und Modifikationsbaustein integriert werden, mit dem die Parameter interaktiv systematisiert werden können, ohne dabei aber die daraus resultierende Geometrie wesentlich zu ändern.

Approximation mit neuronalen Netzen

Künstliche neuronale Netze stellen den Versuch dar, die Struktur und die Eigenschaften des biologischen Vorbilds in eine formale, berechenbare Form abzubilden. Abstrakt lässt sich ein neuronales Netz als Black Box betrachten, die eine n-stellige Eingabe in eine m-stellige Ausgabe überführt. In der Regel besteht zwischen Ein- und Ausgabe ein nichtlinearer Zusammenhang.

Neuronale Netze sollen hier für den Entwurf sowohl der Meridiangeometrie als auch der Schaufelform eingesetzt werden, um Zusammenhänge zwischen Geometrie und Betriebspunktverhalten abzubilden. Zum Aufbau des neuronalen Netzes soll die aus der Parametrisierung erstellte Datenbank als Trainingsmenge dienen. Beim Erlernen der Zusammenhänge aus den Trainingsdaten wird ein erweitertes Backpropagation-Lernverfahren eingesetzt, das in der Literatur als äußerst zuverlässig gilt. Durch die Definition bestimmter Zielvorgaben, z.B. der spezifischen Drehzahl n_q, ist ein trainiertes neuronales Netz in der Lage, ohne rechenzeitaufwändige Strömungsberechnung sowohl eine Strömungsmaschinengeometrie zu generieren als auch Aussagen über zu erwartende strömungsmechanische Eigenschaften zu treffen. Das trainierte neuronale Netz lässt sich somit als schnelles Entwurfswerkzeug bei der Strömungsmaschinenauslegung einsetzen.

Dreidimensionale Strömungsberechnung

Mit Hilfe des neuronalen Netz soll in einem weiteren Schritt eine komplette Pumpen- bzw. Turbinenbaureihe generiert werden, die unter Verwendung des am Lehrstuhl entwickelten Real-Time-Designsystems (RTD) nachgerechnet wird. Bei der Strömungsberechnung kommt ein 3D-Navier-Stokes Code zum Einsatz. Eine abschließende Analyse soll zeigen, dass beliebig generierte Designs im Vergleich zu den gelernten Originalgeometrien ähnliche strömungsmechanische Eigenschaften aufweisen und somit zur Reduzierung der benötigten Entwurfs- und Optimierungszeit beitragen.

Nicht unerwähnt bleiben soll an dieser Stelle die durchweg objektorientierte Modellierung und Implementierung der beschriebenen Verfahren und Algorithmen. Die Programmierung erfolgt für das Backpropagation-Verfahren des neuronalen Netzes in der Programmiersprache C++, vgl. Stroustrup [57]. Zur Gestaltung der interaktiven, grafischen Benutzerschnittstelle sowie der geometrieaufbereitenden Algorithmen wurde auf die plattformunabhängige Programmiersprache Java zurückgegriffen, s. Arnold et al. [1].

Kapitel 2

Theoretische Grundlagen

Wie bereits in Kap. 1.3 erwähnt, befasst sich diese interdisziplinäre Arbeit mit Verfahren dreier Wissensgebiete, der Geometrieaufbereitung und der Strömungsberechnung sowie der Simulation künstlicher neuronaler Netze, wobei die Schwerpunkte dieser Arbeit vor allem in der Geometriebehandlung sowie in der Entwicklung und Anwendung des Neuronale-Netze-Simulators liegen.

2.1 Strömungsberechnung

Zunächst sollen die wesentlichen theoretischen Grundlagen behandelt werden, die für die Strömungsberechnung der aufbereiteten Geometrien von Bedeutung sind. Nach der Vorstellung der Grundgleichungen wird auf die zeitlich gemittelten Gleichungen und deren Schließung mit Turbulenzmodellen eingegangen.

2.1.1 Erhaltungsgleichungen

Zur Berechnung einer inkompressiblen, dreidimensionalen, instationären und reibungsbehafteten Strömung in einer hydraulischen Maschine müssen unter der Annahme einer von der Temperatur unabhängigen Viskosität μ vier Erhaltungsgleichungen gelöst werden, die Kontinuitätsgleichung sowie drei Impulsgleichungen. Unter Vernachlässigung der Schwerkraft lauten die partiellen Differentialgleichungen unter Verwendung der Einsteinschen Summationskonvention für den Absolutgeschwindigkeitsvektor $\vec{c} = (c_x, c_y, c_z)^T$ und den statischen Druck p in kartesischen Koordinaten wie folgt, s. Schlichting und Gersten [53]:

$$\frac{\partial c_i}{\partial x_i} = 0 \,, \tag{2.1}$$

$$\frac{\partial c_i}{\partial t} + c_j \frac{\partial c_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\nu\left(\frac{\partial c_i}{\partial x_j} + \frac{\partial c_j}{\partial x_i}\right)\right] \,. \tag{2.2}$$

Hierin bezeichnen t die Zeit, ρ die Dichte und $\nu = \mu/\rho$ die kinematische Viskosität des Fluids.

Die Lösung der Erhaltungsgleichungen für Strömungen in rotierenden Maschinen erfolgt im Bereich des Laufrades zweckmäßigerweise in einem mit der Winkelgeschwindigkeit $\vec{\omega}$ um die Maschinenachse drehenden Koordinatensystem. Hierfür müssen die Gln. 2.1 und 2.2 mit $\vec{c} = \vec{w} + (\vec{\omega} \times \vec{r})$ ins Relativsystem transformiert werden, wobei in den Impulsgleichungen zusätzliche Quellterme $\vec{S}_{rot}$ für die Flieh- und die Corioliskraft auftreten:

$$\vec{S}_{rot} = -\vec{\omega} \times (\vec{\omega} \times \vec{r}) - 2\,\vec{\omega} \times \vec{w} \; . \tag{2.3}$$

Somit ergeben sich die Navier-Stokes-Gleichungen im Relativsystem zu:

$$\frac{\partial w_i}{\partial x_i} = 0 \; , \tag{2.4}$$

$$\frac{\partial w_i}{\partial t} + w_j \frac{\partial w_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\nu\left(\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i}\right)\right] + S_{rot,i} \; . \tag{2.5}$$

Eine ausführliche Darstellung der Transformation findet sich beispielsweise bei Truckenbrodt [59].

Da dieses System gekoppelter, nichtlinearer Differentialgleichungen selbst für laminare Strömungen nur in wenigen Sonderfällen eine analytische Lösung erlaubt, müssen zur Lösung numerische Verfahren herangezogen werden. Die Gleichungssysteme 2.1 und 2.2 bzw. 2.4 und 2.5 gelten sowohl für laminare als auch für turbulente Strömungen und lassen sich mittels der DNS[1] ohne zusätzliche Modellannahmen lösen. Bei der LES[2] werden nur die kleinen, dissipativen Strukturen approximiert. Bei beiden Verfahren müssen die dreidimensionalen, instationären Gleichungen gelöst werden, die das komplexe Verhalten der turbulenten Strömung beschreiben. Für den Auslegungsprozess und die Optimierung von Turbomaschinen sind die resultierenden Rechenzeiten um Größenordnungen zu hoch, weshalb Modellannahmen unerlässlich sind. Deshalb geht man zu einer statistischen Beschreibung der Turbulenz über, s. Skoda [54].

2.1.2 Turbulenzmodellierung

Eine drastische Reduzierung des Berechnungsaufwandes bei der Lösung des Gleichungssystems 2.1 und 2.2 lässt sich durch eine zeitliche Mittelung erzielen. Nach dem Separationsansatz von Reynolds [33] lässt sich für stationäre Problemstellungen die lokal an einem Ort vorherrschende Strömungsgröße ϕ aufteilen in einen zeitlichen Mittelwert $\overline{\phi}$ und einen Schwankungsanteil ϕ':

[1] DNS = Direkte Numerische Simulation
[2] LES = Large Eddy Simulation

$$\phi(x_i, t) = \overline{\phi}(x_i) + \phi'(x_i, t) \ , \tag{2.6}$$

wobei sich die zeitlich gemittelte Größe $\overline{\phi}(x_i)$ durch Mittelung über einen ausreichend langen Zeitraum T ergibt.

Wird der Ansatz nach Gl. 2.6 in die Navier-Stokes-Gleichungen 2.1 und 2.2 eingesetzt und diese anschließend zeitlich gemittelt, so erhält man die sog. Reynolds-gemittelten Navier-Stokes-Gleichungen:

$$\frac{\partial \overline{c}_i}{\partial x_i} = 0 \ , \tag{2.7}$$

$$\frac{\partial \overline{c}_i}{\partial t} + \overline{c}_j \frac{\partial \overline{c}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\nu\left(\frac{\partial \overline{c}_i}{\partial x_j} + \frac{\partial \overline{c}_j}{\partial x_i}\right) - \overline{c_i' \, c_j'}\right] \ . \tag{2.8}$$

Die Impulsgleichungen enthalten neben der zeitlich gemittelten Geschwindigkeit und des Drucks zusätzlich den Reynolds-Spannungstensor $\overline{c_i' \, c_j'}$, der aus der Mittelung der konvektiven Terme stammt. Die Reynolds-Spannungen stellen die zeitlich gemittelte Wirkung der turbulenten Konvektion dar. Da die Fluktuationsgeschwindigkeiten im Reynolds-Spannungstensor aber unbekannt sind, ist das Gleichungssystem nicht mehr geschlossen. Dieses Schließungsproblem kann über sechs weitere Transportgleichungen durch ein sog. Reynolds-Spannungsmodell gelöst werden, s. Skoda [54]. Im Rahmen einer Optimierung der zu behandelnden Geometrien ist dieser Ansatz jedoch zu rechenzeitintensiv.

Ein stark vereinfachter Schließungsansatz ergibt sich über die direkte Modellierung des Reynolds-Spannungstensors über ein Wirbelviskositätsmodell, das von Boussinesq [6] erstmals eingeführt wurde. In der Modellvorstellung wird der physikalischen Viskosität ν eine zusätzliche von den Strömungsgrößen abhängige turbulente Wirbelviskosität ν_T überlagert. Das Schließungsproblem reduziert sich bei linearen Wirbelviskositätsmodellen auf die Bestimmung dieser turbulenten Scheinviskosität. Je nach Anzahl der notwendigen partiellen Differentialgleichungen zur Bestimmung von ν_T unterscheidet man Null-, Ein-, und Zweigleichungsmodelle.

Zweigleichungsmodelle haben sich im CFD-Bereich als Standard-Turbulenzmodelle etabliert und liefern für viele technische Strömungsprobleme gute Ergebnisse. Sie bereiten hinsichtlich der numerischen Berechnung in der Regel keine Probleme. Insbesondere vor dem Hintergrund, dass die Dauer einer Simulationsrechnung möglichst kurz gehalten werden soll, scheint die Verwendung von Zweigleichungsmodellen deshalb ein geeigneter Kompromiss zwischen der Forderung nach bestmöglicher Turbulenzmodellierung und kurzen Antwortzeiten zu sein.

Bei dem in dieser Arbeit eingesetzten Navier-Stokes-Code wird auf das Standard-k-ϵ-Modell von Launder und Spalding [25] zurückgegriffen. Bei diesem Zweigleichungs-Turbulenzmodell wird eine Differentialgleichung für die spezifische turbulente kinetische Energie k und eine für die Dissipationsrate ϵ gelöst. Aus der Lösung beider Transportgleichungen wird dann die gesuchte Wirbelviskosität ν_T berechnet:

$$\nu_T = C_\mu \frac{k^2}{\epsilon} \ . \tag{2.9}$$

C_μ ist eine Proportionalitätskonstante und wird empirisch oder durch Überlegungen aus der Grenzschichttheorie bestimmt.

Da das Standard-Modell nur für hohe Reynolds-Zahlen gültig ist, wird das Verhalten der Wirbelviskosität in Wandnähe bei reibungsbehafteter Strömung nicht korrekt wiedergegeben. Zusätzlich müssen die wandnahen Bereiche bei der numerischen Behandlung räumlich sehr fein aufgelöst werden, damit die großen Gradienten, die in den Grenzschichten auftretenden, richtig erfasst werden. Durch die Verwendung des sog. logarithmischen Wandgesetzes kann dieser Bereich überbrückt werden. Ausgehend von Messungen lässt sich eine vollentwickelte, turbulente Grenzschicht in drei Bereiche unterteilen, eine viskose Wandschicht, eine logarithmische Schicht und eine turbulente Kernströmung. Für die korrekte Anwendung des Wandgesetzes ist es allerdings erforderlich, dass der wandnächste Rechenpunkt im logarithmischen Bereich der Grenzschicht zu liegen kommt, der mit Hilfe des dimensionslosen Wandabstandes y^+ bestimmt werden kann:

$$y^+ = \frac{u_\tau \, \Delta y}{\nu} \ , \tag{2.10}$$

$$u_\tau = \sqrt{\frac{\tau_w}{\rho}} \ . \tag{2.11}$$

Dieser Bereich liegt unter der Annahme hydraulisch glatter Wände im Intervall $30 < y^+ < 500$. Da y^+ zunächst nicht bekannt ist, lässt sich die Gültigkeit des Wandgesetzes erst nach einer durchgeführten Strömungsberechnung überprüfen.

Eine genaue Beschreibung des hier vorgestellten sowie weiterer Turbulenzmodelle findet man z.B. in den Arbeiten von Bader [3] und Skoda [54].

2.1.3 Numerische Lösung

Zur Lösung der in Kap. 2.1 beschriebenen Gleichungssysteme müssen numerische Verfahren verwendet werden, da eine analytische Lösung nur für wenige Anwendungsfälle möglich ist.

Die grundsätzlichen numerischen Eigenschaften des eingesetzten Navier-Stokes-Code basieren auf einem CFD-Code von Ferziger und Perić [13]. Die Erhaltungsgleichungen werden nach der Finiten-Volumen Methode diskretisiert, wobei von einer konservativen Formulierung der zu lösenden Gleichungen ausgegangen wird. Hierfür wird das Lösungsgebiet zunächst in eine endliche Anzahl von Kontrollvolumina aufgeteilt. In jedem Teilvolumen werden die Navier-Stokes-Gleichungen in integraler Form angewendet, wodurch eine Bilanzgleichung entsteht, die eine Beziehung zwischen der lokalen Änderung einer Erhaltungsgröße ϕ und den über die Zellgrenzen

ein- und austretenden Flüssen sowie der in der Zelle wirkenden Quellen und Senken beschreibt. Die über jedes Teilvolumen integrierten Transportgleichungen lassen sich aufgrund ihrer ähnlichen Struktur und unter Verwendung des Integralsatzes von Gauß in einer allgemeinen Form wie folgt darstellen:

$$\frac{\partial}{\partial t}\int_V \phi\, dV + \int_A \phi\, c_j n_j\, dA = \int_A \Gamma \frac{\partial \phi}{\partial x_j} n_j\, dA + \int_V q_\phi\, dV\ . \tag{2.12}$$

Dabei beschreibt der Faktor Γ einen allgemeinen, dichtebezogenen Diffusionskoeffizient und q_ϕ einen Quellterm, in dem alle weiteren Einflüsse zusammengefasst sind, die nicht durch die anderen drei Terme abgebildet werden.

Durch Approximation der Oberflächen- und Volumenintegrale durch geeignete Funktionen erhält man für jedes Kontrollvolumen eine linearisierte, algebraische Beziehung zwischen der Erhaltungsgröße ϕ am betrachteten Punkt P und den Nachbarwerten sowie weiteren Einflussgrößen. Das daraus resultierende lineare Gleichungssystem kann anschließend mit Hilfe eines geeigneten Lösungsverfahren iterativ gelöst werden.

2.1.4 Rotor - Stator Koppelung

Die korrekte Angabe der Eintrittsrandbedingungen der Laufradströmung ist besonders bei schnellläufigen Wasserturbinen eine wichtige Aufgabe. Das verwendete Berechnungsprogramm kann deshalb die Lauf- und Leitradnetze über ein besonderes Interface koppeln, um so eine gemeinsame Berechnung des ruhenden und rotierenden Schaufelkanals zu ermöglichen.

Die Strömung durch ruhende und rotierende Beschaufelungen ist a priori instationär und im strengen Sinn nicht periodisch mit der Schaufelteilung. Für eine exakte instationäre Berechnung müssten deshalb die Schaufelkanäle aller Elemente gleichzeitig berechnet werden, was für eine Optimierung zu inakzeptabel langen Berechnungszeiten führen würde. Aus diesem Grunde wird ein vereinfachtes Modell zur Kopplung von Rotor und Stator verwendet. Es wird pro Schaufelelement nur ein repräsentativer Kanal berechnet, wobei sowohl die Relativströmung im Laufrad als auch die Absolutströmung im Leitrad als stationär angenommen wird. Da die Stellung des Rotors gegenüber dem Stator nicht berücksichtigt wird, müssen umfangsgemittelte Werte in der Trennebene, im Allgemeinen einer Rotationsfläche, übergeben werden. Der Austausch der Strömungsgrößen erfolgt nach jedem Iterationsschritt zusammen mit der Aktualisierung der Randbedingungen. Diese Vorgehensweise ist sehr robust und führt zu einer quasi stationären Wechselwirkung von Rotor und Stator. Die Verwendbarkeit dieser Methode wird z.B. von Denton [9] und Riedel [35] nachgewiesen.

Die Koppelung von rotierenden und ruhenden Schaufelelementen erfolgt über das sog. *Stage Interface*, in der die Berechnungsnetze auf einer Rotationsfläche aneinandergrenzen. In Umfangsrichtung werden in jeder Zellschicht Mittelwerte berechnet und ausgetauscht. Damit sowohl die Massen- als auch die Impulserhaltung über das

Interface erfüllt ist, wird nach Riedel [35] in jeder Zellschicht eine Flussmittelung in Umfangsrichtung durchgeführt. Somit gehen zwar die Schichtungen in Umfangsrichtung verloren, bleiben aber entlang der Schaufelhöhe erhalten. Durch die Flussmittelung ergibt sich am Interface des jeweiligen Schaufelelementes ein Mittelwert. Die Differenz der Mittelwerte wird zur Verbesserung des Verfahrens auf die jeweiligen lokalen Größen addiert bzw. subtrahiert.

Von Richter [34] wurde nachgewiesen, dass die Wechselwirkungen zwischen Rotor und Stator in hydraulischen Maschinen bei nicht zu kleinen Abständen hinreichend genau berechnet werden können. Durch die Verwendung von charakteristischen Randbedingungen ist es auch möglich, die Strömung in relativ eng zusammenstehenden Gittern zuverlässig zu berechnen. Vergleiche mit instationären Rechnungen von Lauf- und Leitradbeschaufelung sind bei Fritz [14] zu finden.

2.2 Geometriebeschreibung von Strömungsmaschinen

Das für diese Arbeit entwickelte Entwurfssystem beeinhaltet zahlreiche Methoden sowohl zur Darstellung als auch zur Modifikation von Strömungsmaschinengeometrien. Hierbei erweisen sich die sog. Splines, insbesondere die kubischen Splines, zur Darstellung glatter Kurven bzw. zur einfachen Approximation komplexer Geometrieformen als vorteilhaft. In diesem Abschnitt soll deshalb ein Überblick der verwendeten Splinetechniken sowie eine Beschreibung des Aufbaus der strömungsführenden Geometrien einer Turbomaschine gegeben werden.

2.2.1 Kubische Splines

Ein kubischer Spline ist eine glatte Kurve, die durch gegebene Punkte im Koordinatensystem geht und eine minimale Gesamtkrümmung aufweist, s. Bild 2.1. Jedes Teilstück ist dabei durch eine kubische Parabel der Form $a_i x^3 + b_i x^2 + c_i x + d_i$ mit geeigneten Koeffizienten a_i, b_i, c_i und d_i definiert.

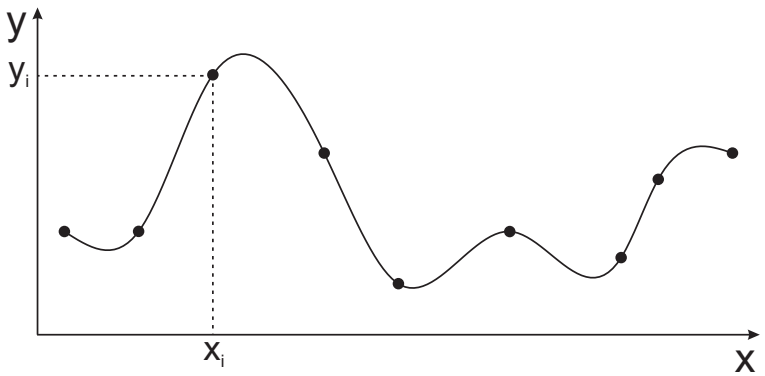


Bild 2.1: Kubischer Spline

„Glatte Kurve“ bedeutet dabei im mathematischen Sinne, dass die Kurve zweimal stetig differenzierbar sein soll. Alle gegebenen Punkte dienen dabei als Stützstellen der Kurve bzw. als Nahtstellen zwischen den Teilkurven, in denen jeweils der Funktionswert wie auch die erste und zweite Ableitung der zusammentreffenden Teilkurven übereinstimmen. Diese Naht- oder Stützstellen werden auch Knoten genannt.

Ausgehend von $n+1$ gegebenen Punkten (x_i, y_i) mit $0 \leq i \leq n$, wobei $x_{i-1} < x_i$ gelte, definiert man zur Gewinnung der Koeffizienten die n Teilstücke des Splines geeigneterweise mit

$$S_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i \ , \tag{2.13}$$

wobei $0 \leq i < n$ ist und $S_i(x)$ das Kurvenstück zwischen den Punkten (x_i, y_i) und (x_{i+1}, y_{i+1}) darstellt. Mit $S_i(x_i) = y_i$ folgt aus Gl. 2.13 sofort $d_i = y_i$.

Da die Teilstücke in den gegebenen Punkten nahtlos ineinander übergehen, gilt $S_{i-1}(x_i) = S_i(x_i)$ für $1 < i \leq n$ und somit:

$$a_{i-1}(x_i - x_{i-1})^3 + b_{i-1}(x_i - x_{i-1})^2 + c_{i-1}(x_i - x_{i-1}) + d_{i-1} = d_i \ . \tag{2.14}$$

In allen gegebenen Punkten haben die anschließenden Teilkurven gleiche Tangenten und es gilt $S'_{i-1}(x_i) = S'_i(x_i)$. Daraus folgt:

$$3a_{i-1}(x_i - x_{i-1})^2 + 2b_{i-1}(x_i - x_{i-1}) + c_{i-1} = c_i \ . \tag{2.15}$$

Ebenso folgt aus der Krümmungsstetigkeit mit $S''_{i-1}(x_i) = S''_i(x_i)$:

$$6a_{i-1}(x_i - x_{i-1}) + 2b_{i-1} = 2b_i \ . \tag{2.16}$$

Durch Umformen folgt aus Gl. 2.16:

$$a_{i-1} = \frac{b_i - b_{i-1}}{3(x_i - x_{i-1})} \ . \tag{2.17}$$

Das Einsetzen von Gl. 2.17 in Gl. 2.15 bzw. 2.14 ergibt:

$$(b_i + b_{i-1})(x_i - x_{i-1}) + c_{i-1} = c_i \tag{2.18}$$

bzw.

$$c_{i-1} = \frac{d_i - d_{i-1}}{x_i - x_{i-1}} - \frac{1}{3}(b_i - b_{i-1})(x_i - x_{i-1}) - b_{i-1}(x_i - x_{i-1}) \ . \tag{2.19}$$

Durch die Erhöhung des Index i ergibt sich eine äquivalente Darstellung von Gl. 2.19:

$$c_i = \frac{d_{i+1} - d_i}{x_{i+1} - x_i} - \frac{1}{3}(b_{i+1} - b_i)(x_{i+1} - x_i) - b_i(x_{i+1} - x_i) \ . \tag{2.20}$$

Damit folgt für Gl. 2.18 durch Einsetzen von Gl. 2.19 und 2.20 und anschließendem Umformen:

$$(x_i - x_{i-1})b_{i-1} + 2(x_{i+1} - x_{i-1})b_i + (x_{i+1} - x_i)b_{i+1} = 3\left(\frac{d_{i+1} - d_i}{x_{i+1} - x_i} - \frac{d_i - d_{i-1}}{x_i - x_{i-1}}\right) . \tag{2.21}$$

Wegen $d_i = y_i$, ist die rechte Seite von Gl. 2.21 für $i > 0$ und $i < n$ bekannt. Somit lassen sich die Koeffizienten b_i für $0 < i < n$ durch Lösen des aus allen Gln. 2.21 bestehenden linearen Gleichungssystems berechnen. Die Koeffizienten b_0 und b_n, die zur Lösung des Gleichungssystems benötigt werden, entsprechen der halben Krümmung im ersten und im letzten Punkt und können frei vorgegeben werden. Der für diese Arbeit entwickelte Algorithmus zur Berechnung kubischer Splines sieht für beide Koeffizienten den Wert $b_0 = b_n = 0$ vor. Die Koeffizienten a_i und c_i gewinnt man anschließend durch Einsetzen der aus der Lösung des Gleichungssystems ermittelten Koeffizienten b_i in die Gln. 2.17 bzw. 2.19.

2.2.2 B-Splines

Im Zuge der Entwicklung der CAD-Systeme wurde in den siebziger Jahren eine besondere Geometriebeschreibung von Oberflächen entwickelt. Diese Beschreibung ist sehr flexibel, so dass sowohl einfache dreidimensionale Geometrieformen wie z.B. Kugeln oder Kegel als auch komplexe dreidimensionale Freiformflächen mit derselben mathematischen Theorie behandelt werden können. Der Vorteil dieser Art der Beschreibung liegt in der Vereinheitlichung der Datenbasen des jeweiligen CAD-Systems. Die Anfänge dieser Technik beruhen auf den Arbeiten von Bezier [5] und sind zu den heute verwendeten NURBS[3] weiterentwickelt worden.

Vor allem bei der Parametrisierung von Strömungsmaschinengeometrien wird in dem zu entwickelden System auf die B-Spline Technik zurückgegriffen. Im Folgenden soll deshalb die zugrundeliegende mathematische Theorie kurz beschrieben werden. Eine umfassendere Behandlung der Thematik findet man beispielsweise bei Hoschek und Lasser [20].

2.2.2.1 Theorie der B-Spline Kurven

Eine B-Spline Kurve setzt sich in der Regel aus mehreren Kurvenintervallen zusammen. Eine Krümmungsstetigkeit an den Anschlussstellen, im Weiteren Knoten

[3]NURBS = Non Uniform Rational Basis Splines

genannt, erhält man durch Verwendung kubischer Polynome. Der geometrische Verlauf der B-Splines wird durch charakteristische Punkte, den sog. Kontrollpunkten bzw. Describern, und den zugehörigen Basis-Funktionen beschrieben.

Die mathematische Formulierung eines Punktes $\vec{P}(u)$ lautet wie folgt:

$$\vec{P}(u) = \sum_{i=1}^{n} \vec{D}_i \cdot N_{i,k}(u) \quad , \quad u_{min} \leq u \leq u_{max} \quad , \quad 2 \leq k \leq n \ . \tag{2.22}$$

In Gl. 2.22 sind $\vec{D}_i$ die n beschreibenden Kontrollpunkte, u der Kurvenparameter, k die Ordnung der Polynome und $N_{i,k}$ die Basisfunktionen der Kontrollpunkte. Sie berechnen sich rekursiv nach der Formel von Cox-de Boor zu, s. [20]:

$$N_{i,1}(u) := \begin{cases} 1 & : \text{ für } t_i \leq u < t_{i+1} \\ 0 & : \text{ sonst} \end{cases} , \tag{2.23}$$

$$N_{i,k}(u) := \frac{u - t_i}{t_{i+k-1} - t_i} N_{i,k-1}(u) + \frac{t_{i+k} - u}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(u) \ , \tag{2.24}$$

wobei t_i die geometrische Lage des jeweiligen Knoten in Form der normierten Bogenlänge darstellt und im sog. Träger- bzw. Knotenvektor $\vec{T}(t_1, t_2, \ldots, t_s)$ definiert ist. Die Anzahl der Elemente des Knotenvektors berechnet sich durch $s = n + k$. Aus Gl. 2.24 wird deutlich, dass der Wert der Knotenelemente, d.h. die Lage der Polynomanschlüsse, die Basisfunktionen beeinflussen und somit den geometrischen Verlauf des B-Splines mitbestimmt.

Soll der erste und letzte Kontrollpunkt mit dem Kurvenanfangs- und -endpunkt zusammenfallen, so muss das erste und letzte Element des Knotenvektors k-fach belegt werden. In diesem Fall ist im Kurvenanfangs- und -endpunkt die Tangente durch die Gerade der beiden ersten und letzten Kontrollpunkte bestimmt. Im Verlauf der Kurve kann die Tangente bzw. erste Ableitung wie folgt berechnet werden:

$$\vec{P}'(u) = \sum_{i=1}^{n} \vec{D}_i \cdot N'_{i,k}(u) \ , \tag{2.25}$$

$$\vec{P}''(u) = \sum_{i=1}^{n} \vec{D}_i \cdot N''_{i,k}(u) \ , \tag{2.26}$$

wobei sich die Ableitungen der Basisfunktionen folgendermaßen ermitteln lassen:

$$N'_{i,k}(u) = \frac{N_{i,k-1}(u) + (u - t_i) N'_{i,k-1}(u)}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - u) N'_{i+1,k-1}(u) - N_{i+1,k-1}(u)}{t_{i+k} - t_{i+1}} \ , \tag{2.27}$$

$$N''_{i,k}(u) = \frac{2N'_{i,k-1}(u) + (u - t_i)N''_{i,k-1}(u)}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - u)N''_{i+1,k-1}(u) - 2N'_{i+1,k-1}(u)}{t_{i+k} - t_{i+1}} . \quad (2.28)$$

Für B-Splines mit äquidistanten Kurvensegmenten ergibt sich ein *uniformer* B-Spline. Haben die Kurvensegmente unterschiedliche Längen, spricht man von einem *nicht-uniformen* B-Spline.

Bezüglich der Kurvenmodifikation ist eine weitere wichtige Eigenschaft der B-Splines, dass für $n > k$ die Basisfunktionen der Kontrollpunkte nur im Bereich t_i bis t_{i+k} ungleich Null sind. Eine Verschiebung des Kontrollpunktes $\vec{D}_i$ beeinflusst demnach nur denjenigen Kurvenbereich für den

$$t_i \leq u \leq t_{i+k} \quad (2.29)$$

gilt, s. Bild 2.2. Ist die Anzahl der Kontrollpunkte n gleich der Polynomordnung k wird der B-Spline durch sog. Bernstein-Polynome als Basisfunktionen gebildet und man spricht von einer Bézier-Kurve. In diesem Fall beeinflusst eine Kontrollpunktverschiebung den gesamten Kurvenverlauf. Lokale Kurvenmodifikationen werden dadurch unmöglich.

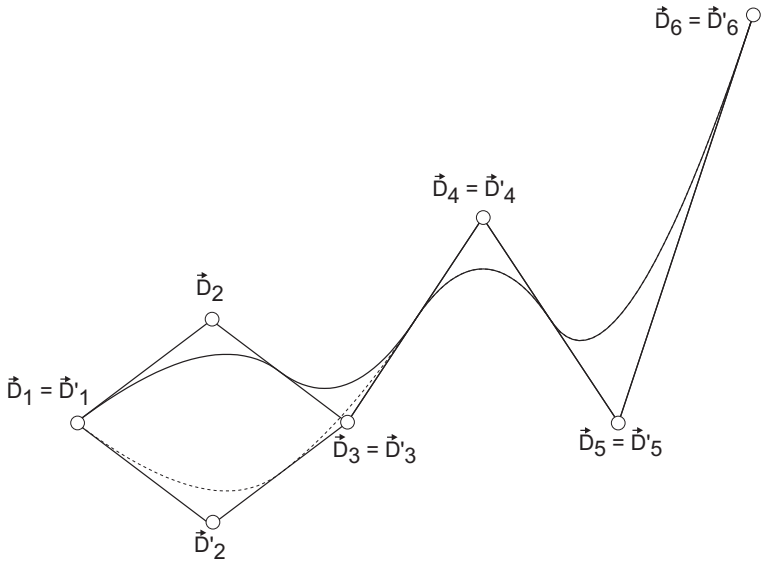


Bild 2.2: Lokale Änderung des Kurvenverlaufs durch Modifikation eines Kontrollpunktes

In Gl. 2.22 wird anhand der Basisfunktionen und der Kontrollpunktkoordinaten der Verlauf einer B-Spline Kurve berechnet. Sind jedoch Kurvenpunkte gegeben und die zugehörige Splinekurve gesucht, muss das inverse Problem gelöst werden. Die in Gl. 2.22 noch unbekannten Kontrollpunktkoordinaten müssen berechnet und

die gegebene Kurve dekodiert werden. Dazu lässt sich für die r Kurvenpunkte die jeweilige normierte Bogenlänge u_i berechnen:

$$u_i = \frac{\sum_{j=2}^{i} |(\vec{P}_j - \vec{P}_{j-1})|}{\sum_{j=2}^{r} |(\vec{P}_j - \vec{P}_{j-1})|} t_{max} \ . \tag{2.30}$$

Aus den Parameterwerten u_i und einer vorgegebenen Anzahl n an Kontrollpunkten kann der Trägervektor definiert werden. Mit diesem Trägervektor können des Weiteren für jeden Kurvenpunkt $\vec{P}_i(u_i)$ die Basisfunktionen nach Gl. 2.24 bestimmt werden. Nach Gl. 2.22 ergibt sich somit für jeden gegebenen Punkt:

$$\vec{P}_i(u_i) = \vec{D}_1 N_{1,k}(u_i) + \vec{D}_2 N_{2,k}(u_i) + \cdots + \vec{D}_n N_{n,k}(u) \ . \tag{2.31}$$

Für alle r Kurvenpunkte ergibt sich ein lineares inhomogenes Gleichungssystem, das in Matrixschreibweise wie folgt formuliert wird:

$$\mathbf{P} = \mathbf{C} \cdot \mathbf{D} \ . \tag{2.32}$$

Für den allgemeinen dreidimensionalen Fall ist $\mathbf{P}$ die $r \times 3$ Matrix der Kurvenpunkte, $\mathbf{C}$ die $r \times n$ Koeffizientenmatrix der Basisfunktionen und $\mathbf{D}$ die $n \times 3$ Matrix der gesuchten Kontrollpunkte. Durch Umformen der Matrixgleichung 2.32 ergeben sich die gesuchten Kontrollpunkte zu:

$$\mathbf{D} = \left(\mathbf{C}^T\mathbf{C}\right)^{-1} \mathbf{C}^T\mathbf{P} \ . \tag{2.33}$$

Ist die Anzahl der Kontrollpunkte gleich der Anzahl der gegebenen Kurvenpunkte $n = r$, ergibt sich ein Interpolationsspline, der durch die vorgegebenen Punkte läuft. Für $n < r$ ist das Gleichungssystem 2.32 überbestimmt und kann nicht mehr direkt gelöst werden, sondern nur mit Hilfe einer Zusatzbedingung, z.B. minimaler Abweichung. Der durch die Kontrollpunkte definierte Spline approximiert die vorgegebene Kurve.

Durch die Eigenschaft des Splines zu jedem Parameter u_i die Summe der Fehlerquadrate minimal zu halten, ist die Approximationsgenauigkeit für die meisten Anwendungsfälle ausreichend genau. Eine Vorgehensweise zur Verbesserung der Genauigkeit ist bei Fernández [12] beschrieben.

2.2.2.2 Flächenbeschreibung mit B-Splines

Analog zur Beschreibung von dreidimensionalen Kurven kann die B-Spline-Technik auch für die Beschreibung von Flächen verwendet werden. Erst durch diese Technik können Freiformflächen mathematisch definiert werden. Ein Flächenpunkt $\vec{P}$ wird mit den jeweiligen Laufparametern u und v beschrieben durch:

$$\vec{P}(u,v) = \sum_{i=1}^{n}\sum_{j=1}^{m} \vec{D}_{i,j} N_{i,k}(u) M_{j,l}(v) \ . \tag{2.34}$$

Die Basisfunktionen für die u- und v-Richtung bilden ein Tensor-Produkt. Man spricht in diesem Zusammenhang auch von Tensor-Produkt-Flächen. Die Basisfunktionen werden analog zu Gl. 2.24 wie folgt definiert:

$$N_{i,1}(u) := \begin{cases} 1 & : \text{ für } t_i \le u < t_{i+1} \\ 0 & : \text{ sonst} \end{cases} , \tag{2.35}$$

$$N_{i,k}(u) := \frac{u - t_i}{t_{i+k-1} - t_i} N_{i,k-1}(u) + \frac{t_{i+k} - u}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(u) \ , \tag{2.36}$$

$$M_{j,1}(v) := \begin{cases} 1 & : \text{ für } s_i \le v < s_{i+1} \\ 0 & : \text{ sonst} \end{cases} , \tag{2.37}$$

$$M_{j,l}(v) := \frac{v - s_j}{s_{j+l-1} - s_j} M_{j,l-1}(v) + \frac{s_{j+l} - v}{s_{j+l} - s_{j+1}} M_{j+1,l-1}(v) \ . \tag{2.38}$$

Hierbei sind s_i und t_i die Elemente der Trägervektoren $\vec{S}$ und $\vec{T}$. Die Ordnung der B-Spline Fläche in v-Richtung wird durch l und in u-Richtung durch k dargestellt. Die Fläche wird dann von einem $n \times m$ Netz von Kontrollpunkten aufgespannt, s. Bild 2.3.

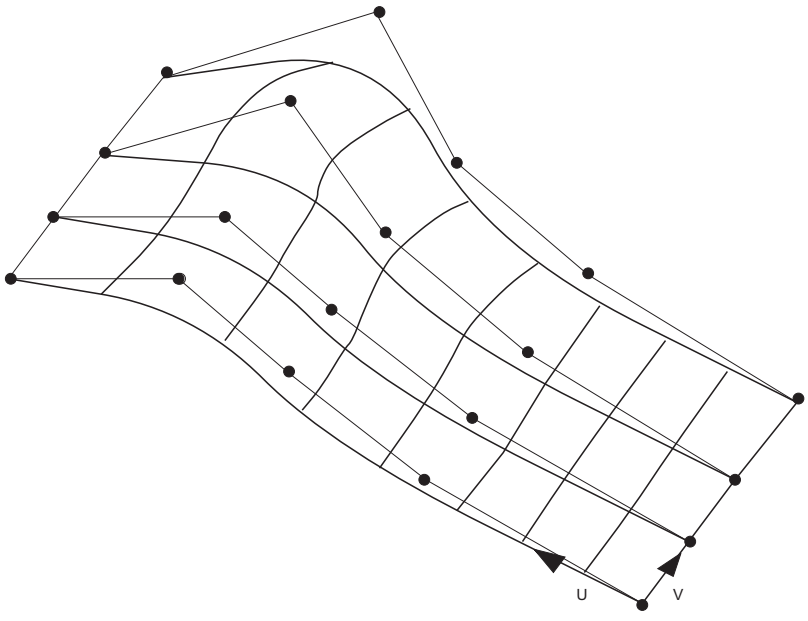


Bild 2.3: B-Spline Fläche mit zugehörigen Kontrollpunkten

Für eine Fläche erfolgt die Dekodierung analog zu den B-Spline Kurven. Die vorgegebene Fläche, bestehend aus einem Satz $r \times q$ Punkten $\vec{P}(u,v)$, kann nach Gl. 2.34 wie folgt formuliert werden:

$$\begin{aligned}\vec{P}(u,v) &= N_{1,k}(u)\left[\vec{D}_{1,1}M_{1,l}(v)+\cdots+\vec{D}_{1,m}M_{m,l}(v)\right]+ \\ &\cdots \\ &N_{n,k}(u)\left[\vec{D}_{n,1}M_{1,l}(v)+\cdots+\vec{D}_{n,m}M_{m,l}(v)\right]\,.\end{aligned} \tag{2.39}$$

In Matrixschreibweise lässt sich das lineare inhomogene Gleichungssystem folgendermaßen formulieren:

$$\mathbf{P} = \mathbf{C}\cdot\mathbf{D}\ . \tag{2.40}$$

Die Ausgangspunkte $\mathbf{P}$ bilden hierbei eine $(r\cdot q)\times 3$ Matrix, das Produkt der Koeffizienten ist in der $(r\cdot q)\times(n\cdot m)$ Matrix $\mathbf{C}$ dargestellt und die gesuchten Kontrollpunkte bilden die $(n\cdot m)\times 3$ Matrix $\mathbf{D}$. Durch Umformen der Matrixgleichung 2.40 erhält man analog zu Gl. 2.32 folgenden Ausdruck:

$$\mathbf{D} = \left(\mathbf{C}^T\mathbf{C}\right)^{-1}\mathbf{C}^T\mathbf{P}\ . \tag{2.41}$$

Ist die Feldgröße der vorgegebenen Punkte gleich der Größe des Kontrollpunktfeldes $r = n$ und $q = m$, so erhält man durch die Dekodierung eine Flächengeometrie, die die Ausgangspunkte interpoliert. Eine approximierte Flächengeometrie erhält man bei $n \leq r$ und $m \leq q$. Eine Verbesserung der Approximationsgenauigkeit ist durch die iterative Berechnung der Laufparameter u und v möglich, s. Rogers und Fog [39].

Zur Dekodierung von Schaufelskelettflächen wird in dem zu entwickelnden System eine spezielle Technik eingesetzt, die sich auch in den Entwurfssystemen am FLM durchgesetzt hat. Hierbei handelt es sich um ein Verfahren, das in der Literatur unter dem Namen *Cross Sectional Design* Verbreitung findet, s. Piegl [32]. Sind Flächenpunkte auf Schnittkurven gegeben, so werden zunächst entlang dieser Schnitte B-Spline Kurven erzeugt. Dabei müssen alle Splines die gleiche Ordnung k und die gleiche Anzahl an Kontrollpunkten aufweisen. Der Trägervektor aller Kurven ist somit identisch, und nach Gl. 2.22 gilt für die Kurvenpunkte in u-Richtung der folgende Ausdruck:

$$\vec{P}(u, v = \text{const.}) = \sum_{i=1}^{n}\vec{D}'_i\cdot N_{i,k}(u)\ . \tag{2.42}$$

Anschließend wird über die berechneten Kontrollpunkte $\vec{D}'_i$ in v-Richtung dekodiert und man erhält die endgültigen Kontrollpunkte der B-Spline Fläche:

$$\vec{D}'_i(v) = \sum_{j=1}^{m}\vec{D}_j\cdot M_{j,l}(v) \qquad \text{für} \qquad i = \text{const.}\ . \tag{2.43}$$

Soll gewährleistet werden, dass die Kontrollpunkte $\vec{D}'_i$ Kurvenpunkte der B-Splines in v-Richtung sind, so ist zur Dekodierung ein Interpolationspline zu verwenden, d.h. es ist $m = r$ zu setzen.

Die geschilderte Art der Flächendekodierung bietet den Vorteil, dass nur in u-Richtung eine Approximation durchgeführt werden muss. Die Abweichung der B-Spline Fläche von den vorgegebenen Punkten ist damit geringer als bei einer vollständigen Dekodierung. Voraussetzung dieser Methode ist jedoch, dass die vorgegebenen Flächenpunkte auf Schnitten $v = \text{const.}$ zur Verfügung stehen.

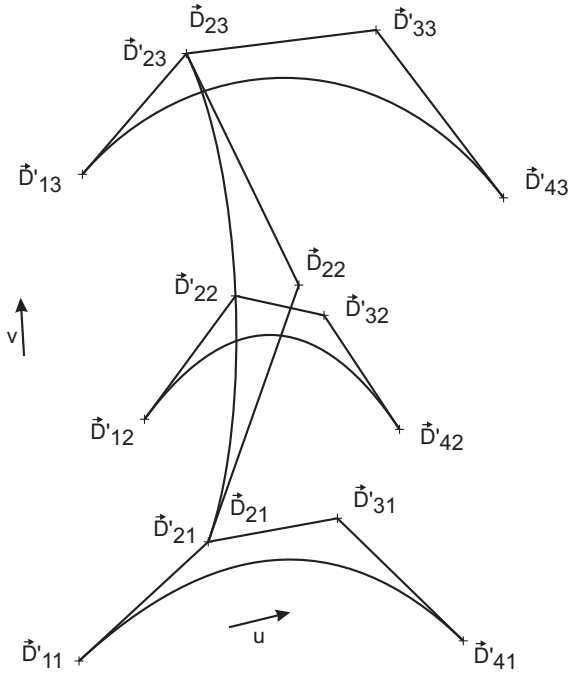


Bild 2.4: Cross Sectional Design von B-Spline Flächen

2.2.3 Aufbau der strömungsführenden Geometrie

Zur Erläuterung des geometrischen Aufbaus der verschiedenen Elemente einer Strömungsmaschine bedient man sich zweckmäßigerweise des klassischen Entwurfs von Beschaufelungen hydraulischer Maschinen. Dieser beginnt mit dem Entwurf der Nabe und Deckscheibe sowie der Festlegung der Schaufelein- und Austrittskanten in der Meridianebene. Anschließend wird die Beschaufelung in der konformen Abbildung entworfen. Durch die Kombination der konformen Abbildung mit der Meridiankontur lässt sich die Schaufelgeometrie in ihre eigentliche dreidimensionale Form überführen.

2.2.3.1 Aufbau der Meridiankontur

Bei gegebenem Volumenstrom gilt es zur Vorauslegung der Meridiankontur, eine möglichst gleichmäßige Geschwindigkeitsverteilung der absoluten Meridiangeschwindigkeit durch das Laufrad zu erreichen, wobei eine Schichtung quer zur Strömungsrichtung in diesem Entwurfsstadium vernachlässigt wird. Des Weiteren sind starke Krümmungsänderungen entlang der Konturen zu vermeiden, damit keine kritischen Beschleunigungen bzw. Verzögerungen in der Strömung auftreten können. Die Modellvorstellung des Meridiankonturentwurfs basiert auf der Auswertung des Flächenverlaufs innerhalb des Laufrades, der wie folgt berechnet wird und in Bild 2.5 für eine Kreiselpumpe dargestellt ist:

$$\frac{A_i}{A_0} = \frac{2\pi\, r_i \cdot \delta_i}{2\pi\, r_0 \cdot \delta_0}\ . \tag{2.44}$$

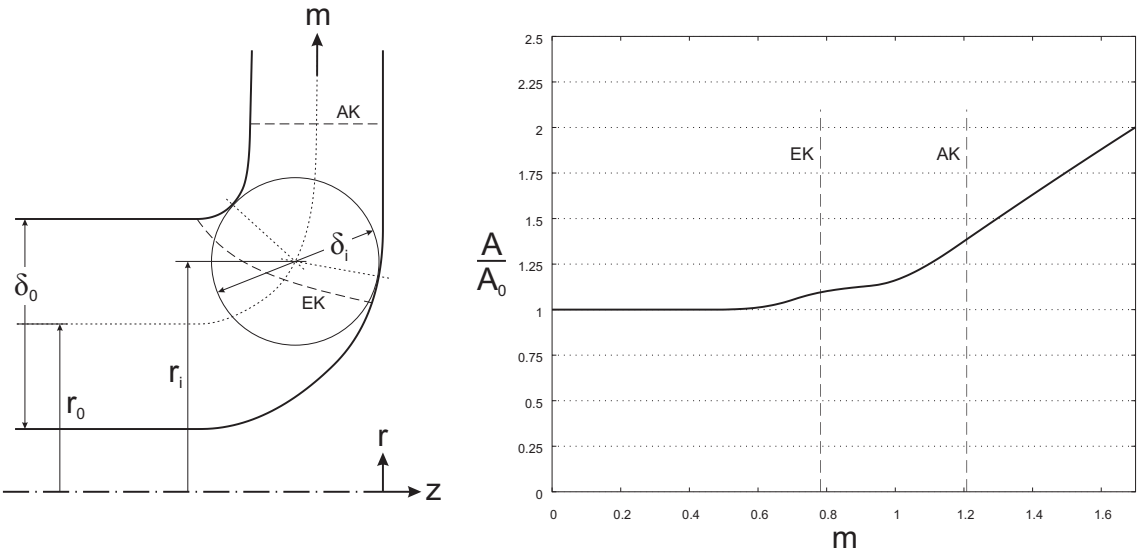


Bild 2.5: Definition des Flächenverlaufs in der Meridianebene

Hierbei ist A_0 eine repräsentative Querschnittsfläche vor dem Laufrad. Die verschiedenen Kreisdurchmesser δ_i und der dazugehörige Kreismittelpunktsradius r_i müssen in einem iterativen Verfahren durch sog. Auskreisen der Meridiankontur bestimmt werden. Ausgehend von dieser initialen Meridiangeometrie kann eine Optimierung erfolgen.

2.2.3.2 Aufbau der Schaufelflächen

Die Beschaufelung setzt sich aus einer Skelettfläche und der zu überlagernden Profilierung zusammmen. Dabei werden unterschiedliche Profilierungen als Initialgeometrie verwendet, z.B. konstante Dicke bei radialen Kreiselpumpen, einfache Profilierungen bei Francis Turbinen und Dickenverteilungen aus der NACA-Tabelle für Ventilatoren. Die Initialgeometrie der Schaufelskelettfläche erhält man aus der Vorgabe von Ein- und Austrittswinkeln entsprechend der Stromfadentheorie. Durch die

Vorgabe eines repräsentativen Strömungswinkelverlaufs können die Skelettlinien der Skelettfläche unter Berücksichtigung der endlichen Schaufelzahl auf Teilflutbahnflächen generiert werden. Durch Approximation der Skelettlinien mittels B-Splines und der Interpolation einer B-Spline Fläche auf der Basis des Cross Sectional Design entsteht die Schaufelskelettfläche. Durch die Überlagerung der jeweiligen Dickenverteilung auf die Skelettlinien und nachfolgender Approximation erhält man die Schaufeloberfläche auf der Basis einer NURBS-Fläche.

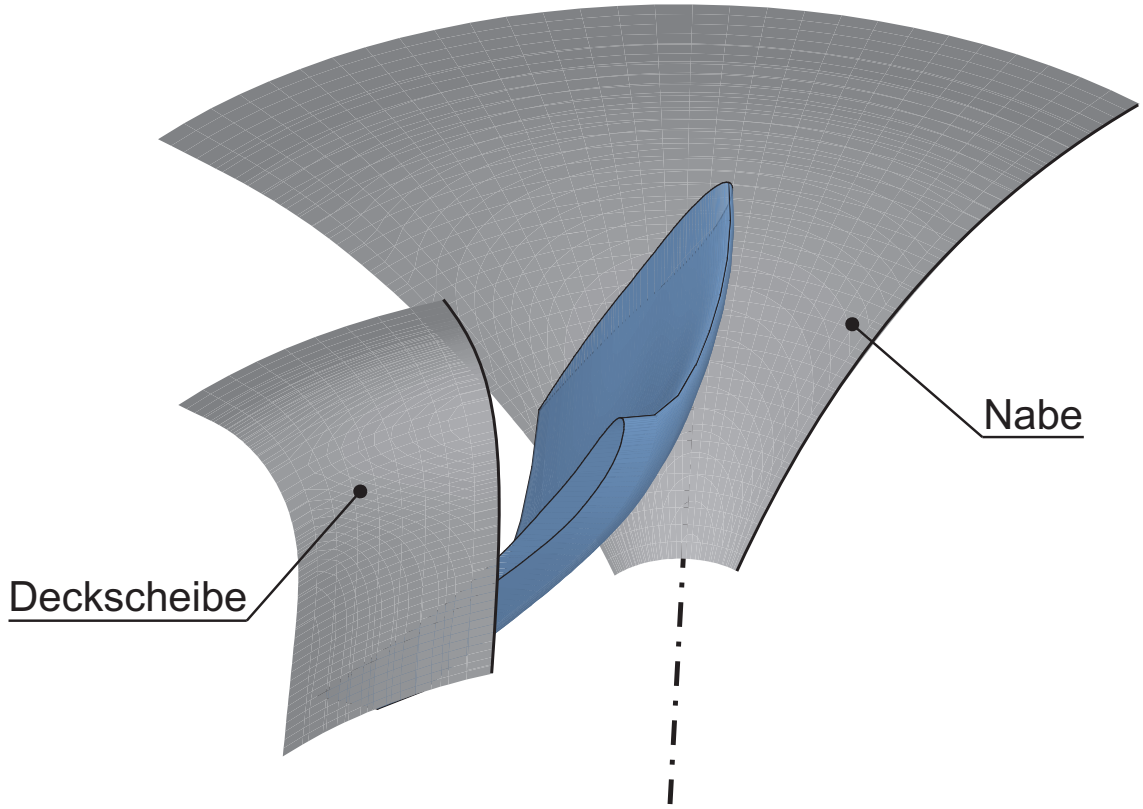


Bild 2.6: Darstellung der Schaufeloberfläche sowie der Rotationsflächen von Deckscheibe und Nabe einer Francis Turbine

2.3 Künstliche neuronale Netze

In dieser Arbeit soll ein Verfahren mit künstlichen neuronalen Netzen entwickelt werden, das durch Eingabe von Betriebspunktparametern in der Lage ist, eine entsprechende Strömungsmaschinengeometrie vorzuschlagen. Der dargestellte Ansatz dient zum schnellen Strömungsmaschinenentwurf und als Möglichkeit zur Bestimmung einer Startgeometrie in einer sich anschließenden Optimierung. In diesem Abschnitt soll deshalb ein Einblick in die Thematik der künstlichen neuronalen Netze gegeben werden.

2.3.1 Biologischer Hintergrund

Um den Aufbau und die Funktionsweise künstlicher neuronaler Netze besser verstehen zu können, wird an dieser Stelle das biologische Vorbild kurz beschrieben.

Die grundlegende Einheit im menschlichen Gehirn ist die Nervenzelle, das sog. *Neuron*. Jedes Neuron ist ein einfacher Prozessor, der Informationen verarbeitet und an andere Neuronen weiterleitet. Trotz der unterschiedlichen Funktionsweisen und Aufgaben lassen sich die Neuronen auf eine grundlegende Struktur zurückführen, wie in Bild 2.7 dargestellt.

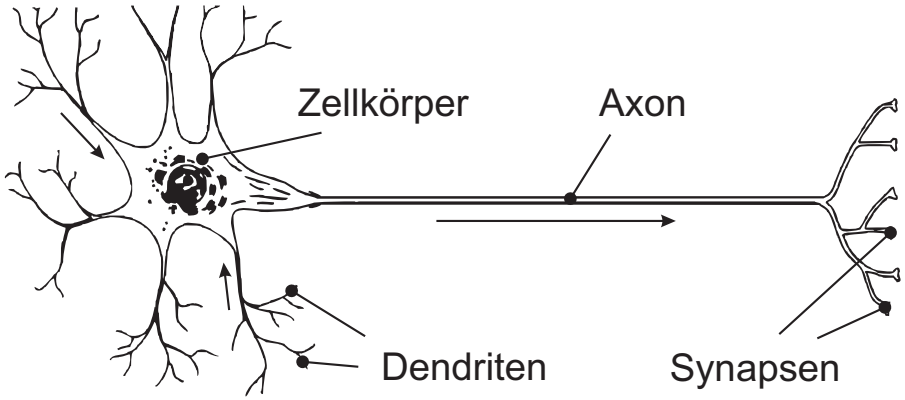


Bild 2.7: Aufbau einer Nervenzelle mit *Zellkörper*, *Axon* und *Dendriten*, die über *Synapsen* Signale von Vorgängerzellen erhalten

Im Wesentlichen besteht ein Neuron aus einem *Zellkörper* sowie den vom Zellkörper ausgehenden Fortsätzen. Diese Fortsätze lassen sich unterteilen in die vielfach aufgefaserten *Dendriten* und eine normalerweise viel längere Nervenfaser, das *Axon*. Die Dendriten der Zelle bzw. der Zellkörper dienen hierbei als Eingänge der Signale anderer Neuronen. Überschreitet die Summe aller Eingangssignale, der sog. Generatorpotentiale, einen bestimmten Schwellwert, so *feuert* das Neuron, d.h. die Übertragung des Signals bzw. der Reizinformation geschieht in Form von kurzzeitigen Potentialsprüngen, den sog. Aktionspotentialen, die das Axon entlanglaufen. Die Übertragung des Signals zu anderen Neuronen erfolgt über die an den Endpunkten des Axons befindlichen *Synapsen*. Diese Synapsen liegen entweder an den Dendriten oder am Zellkörper benachbarter Nervenzellen an.

Zwischen der präsynaptischen Zelle und der postsynaptischen Zelle gibt es in der Regel keinen direkten Kontakt. Die präsynaptische Zelle produziert auf eine Erregung hin Überträgerstoffe, sog. *Neurotransmitter*, die über den synaptischen Spalt zur postsynaptischen Zelle diffundieren und dort wiederum ein Generatorpotential hervorrufen. Bezüglich des Zellkontakts unterscheidet man zwischen *excitatorischen*[4] und *inhibitorischen*[5] Synapsen. Der Kontakt des Axons mit einem Dendriten des nachfolgenden Neuron ist excitatorisch, d.h. die Signale werden weiter geleitet, dagegen ist der Kontakt mit dem Zellkörper inhibitorischer Natur, wodurch eine Weiterleitung der Signale blockiert wird.

[4] excitatorisch = anregend
[5] inhibitorisch = hemmend

Für die Informationsübertragung zwischen Nervenzellen ist es jedoch nicht nur bedeutsam, ob sie erregender oder hemmender Natur ist, sondern es kann auch ein unterschiedlich großer Übertragungswiderstand bestehen, den ein Aktionspotential überwinden muss, um auf der postsynaptischen Seite ein Potential zu erzeugen. Die variable Stärke von synaptischen Verbindungen spielt vor allem bei Gewöhnungs- und Lernvorgängen eine große Rolle. Ein Beispiel hierfür ist das Phänomen der *chemischen Ermüdung*, bei der die präsynaptische Zelle aufgrund starker Stimulation mit der Zeit immer weniger Neurotransmitter in den Spalt absondert und somit zu einer Vergrößerung des Übertragungswiderstands führt. Neben der Konzentration des Neurotransmitters im synaptischen Spalt wird der Übertragungswiderstand bzw. die *Effizienz* der Synapsen u.a. auch durch die Größe der Synapse sowie die Rezeptorendichte zur Aufnahme der Neurotransmittersubstanzen an der postsynaptischen Zelle bestimmt. Es lässt sich unschwer erkennen, welchen Nutzen ein neuronales Netzwerk aus der variablen Verbindungsstärke zieht: Informationen lassen sich in den Synapsen durch ihre Effizienz speichern, worauf z.B. die Funktionsweise des Langzeitgedächtnis zurückzuführen ist, s. Kruse et al. [24]. In anderen Bereichen wird dieser Effekt genutzt, um bei häufiger Benutzung einer Synapse einen bestimmten Übertragungsweg zu bahnen, sei es durch Wachstum oder durch Neustrukturierung der Verbindungen zwischen den Neuronen.

Die *Veränderung* der synaptischen Effizienz bzw. der Verbindungsstruktur zum Erreichen eines bestimmten Informationsmuster oder Übertragungsweges wird in der Literatur allgemein als *Lernen* bezeichnet.

Bezüglich des Aufbaus des Gehirns fällt auf, dass die vielfach miteinander vernetzten Nervenzellen nicht gleichmäßig über das Gehirnvolumen verteilt sind, sondern dass man Ansammlungen von Neuronen in bestimmten Gebieten findet. Dies spricht für die Annahme eines modularen Aufbaus des Gehirns, bei dem jedes Modul seine eigene Funktion erfüllt. Die Neuronen der einzelnen Module sind dabei nicht selten in *Schichten* angeordnet, d.h. die Neuronen einer Schicht sind dabei nur mit den Neuronen der nächsten Schicht verbunden. Dies ermöglicht einen gerichteten Informationsfluss von der ersten Neuronenschicht zur letzten Neuronenschicht, was vor allem zur Abbildung eines bestimmten Eingangssignals auf ein zugehöriges Ausgangssignal von Nutzen ist.

Beim Vergleich der Arbeitsweise eines herkömmlichen Computers und eines Gehirns ergeben sich grundlegende Unterschiede. Während man im Computer eine komplexe Zentraleinheit und einen Arbeitsspeicher findet, gibt es im Gehirn eine große Menge sehr einfacher Prozessoren, die miteinander vernetzt sind. Im Rechner müssen Programmanweisungen nacheinander vom Speicher in die Zentraleinheit übermittelt werden, was einen Engpass bei der Verarbeitung darstellt, während die Prozessoren im Gehirn hochgradig parallel arbeiten.

2.3.2 Aufbau künstlicher neuronaler Netze

Das Ziel bei der Anwendung künstlicher neuronaler Netze ist es, sich von der Arbeitsweise konventioneller Rechner zu entfernen und sich an der Arbeitsweise des menschlichen Gehirns zu orientieren. Gerade bei der zeitaufwändigen Auslegung

neuer Strömungsmaschinen ist die rechneradäquate Nachbildung menschlicher Denkmuster und Erfahrungen von großem Nutzen. Obwohl die in den fünfziger Jahren hochgesteckten Ziele zur Entwicklung intelligenter Programme in keinem Fall auch nur annähernd erreicht worden sind, bilden künstliche neuronale Netze eine wichtige Möglichkeit, nichtlineare Zusammenhänge herzustellen. Hierbei sind insbesondere ihre Lernfähigkeit, ihre Generalisierungsfähigkeit zur Verallgemeinerung von Information und ihre Adaptierbarkeit zur schnellen Anpassung an neue Gegebenheiten sowie ihre Fehlertoleranz bezüglich verrauschter oder unvollständiger Informationen hervorzuheben.

In den folgenden Abschnitten wird die Überführung des biologischen Vorbilds in ein datentechnisches Verarbeitungsmodell beschrieben.

2.3.2.1 Idealisierte Darstellung eines Neurons

Bei der Entwicklung künstlicher neuronaler Netze beschränkt man sich im Allgemeinen darauf, mit vereinfachten Neuronen zu arbeiten. Ein Schema der Funktionsweise eines Neurons zeigt Bild 2.8.

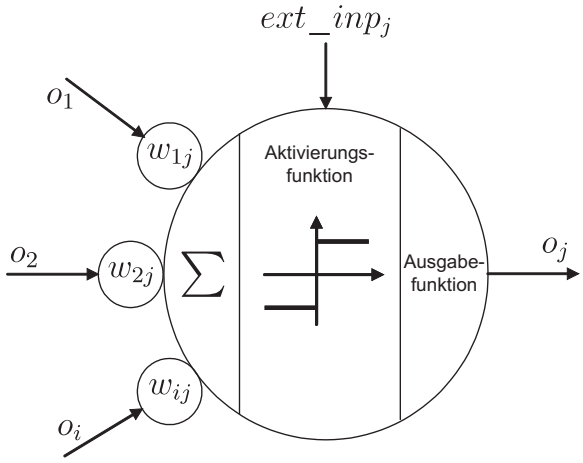


Bild 2.8: Schema eines vereinfachten Neurons

Die elektrischen Potentiale biologischer Nervenzellen entsprechen bei künstlichen Neuronen sog. Aktivierungszuständen. Über die an einem Neuron j einlaufenden Verbindungen werden Aktivierungen von anderen Neuronen o_i herbeigeführt. Die Aktivierungen müssen den Übertragungswiderstand an den Verbindungen zwischen zwei Neuronen überwinden, d.h. sie werden gemäß der Stärke w_{ij} ihrer Verbindungen gewichtet. Die gewichteten Aktivierungen werden zum Nettoinput net_j des empfangenden Neurons aufsummiert:

$$net_j = \sum_i o_i \cdot w_{ij} \ . \tag{2.45}$$

Diese Formel ist in der Literatur auch unter dem Namen *Propagierungsfunktion* bekannt.

Zum Nettoinput kann ein externer Input ext_inp_j hinzukommen, der als Aktivierungszufluss von außen anzusehen ist. Der neue Aktivierungszustand a_j des Neurons zum Zeitpunkt $t+1$ ergibt sich unter Anwendung der *Aktivierungsfunktion* aus dem Nettoinput und dem externen Input zum Zeitpunkt $t+1$ sowie der alten Aktivierung zum Zeitpunkt t:

$$a_j(t+1) = f_{act}(net_j(t+1),\, ext\_inp_j(t+1),\, a_j(t)) \;. \tag{2.46}$$

Als Aktivierungsfunktionen werden häufig lineare Funktionen, Schwellwertfunktionen oder Sigmoidfunktionen eingesetzt, wie in Bild 2.9 dargestellt. Eine besondere Bedeutung kommt dabei der sigmoiden Aktivierungsfunktion zu, auf die in Kap. 3.3 näher eingegangen wird.

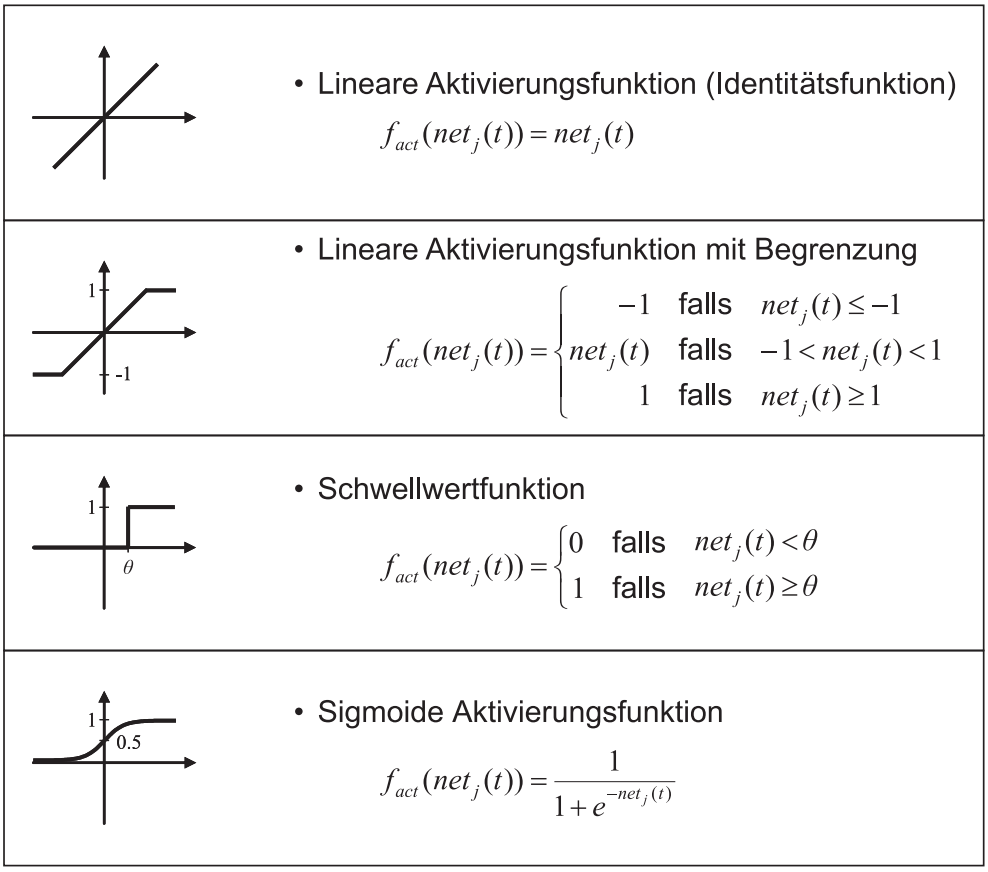


Bild 2.9: Häufig verwendete Aktivierungsfunktionen

In einem letzten Schritt wird der (innere) Aktivierungszustand durch die *Ausgabefunktion* in den eigentlichen Ausgabewert o_j transformiert, der an benachbarte Neuronen weitergegeben wird:

$$o_j = f_{out}(a_j) \;. \tag{2.47}$$

Die Ausgabefunktion *Winner-take-all* ermöglicht z.B. einen Wettbewerb unter den einzelnen Neuronen einer Schicht, indem nur das Neuron mit der größten Aktivierung

seine Information weiter geben darf. In der Regel wird der Ausgabezustand jedoch über die *Identitätsfunktion* dem inneren Aktivierungszustand gleichgesetzt:

$$o_j = a_j \ . \tag{2.48}$$

Die bisher angestellten Überlegungen gelten nur für einzelne Neuronen. Jedes Neuron entspricht dabei einem sehr einfachen Prozessor-Element, das immer nach dem gleichen Schema arbeitet. Die Lösung komplexer Aufgaben erfordert allerdings eine systematische Anordnung mehrerer Neuronen in einer Netzwerkstruktur.

2.3.2.2 Topologie neuronaler Netze

In der Regel ist in einem Netzwerk nicht jedes Neuron mit jedem anderen Neuron verbunden. Vielmehr sind die Neuronen wie beim biologischen Vorbild in Schichten angeordnet, sog. *neuron layers*. Bei der Simulation der Aktivierungsausbreitung wird der Zustand der Neuronen schichtenweise berechnet, d.h. man bestimmt zunächst die Aktivierung der Neuronen einer Schicht, bevor zur nächsten Ebene übergegangen wird. Allgemein besteht ein neuronales Netz aus einer Eingabeschicht (*input layer*), einer Ausgabeschicht (*output layer*) sowie einer beliebigen Anzahl von verdeckten Zwischenschichten (*hidden layers*). Der Anstoss zur Verarbeitung von Informationen in neuronalen Netzen wird durch das Anlegen externer Eingabewerte an die Neuronen der Eingabeschicht gegeben. In dieser Schicht werden die Eingabewerte aber in der Regel nicht weiterverarbeitet, vielmehr ist der Ausgabewert eines Neurons der Eingabeschicht mit dem externen Eingabewert identisch. In Bild 2.10 ist ein zweistufiges[6] Netz mit drei Neuronenschichten dargestellt. Mathematisch betrachtet beschreibt eine solche Netztopologie einen gerichteten und gewichteten Grafen aus Knoten (Neuronen) und Kanten (Verbindungen zwischen den Neuronen).

In einem Netzwerk können Verbindungen sowohl zwischen Neuronen unterschiedlicher Schichten als auch zwischen Neuronen der gleichen Schicht bestehen. Im letzteren Fall handelt es sich häufig um Verbindungen, mit denen ein Wettbewerb unter den Neuronen einer Schicht simuliert werden kann. Dabei wird beispielsweise der Aktivierungszustand eines Neurons dazu genutzt, die anderen Neuronen der gleichen Schicht zu hemmen.

Bezüglich der Richtung der Aktivierungsausbreitung klassifiziert man zwei Arten von neuronalen Netzen.

Netze ohne Rückkopplung (Feedforward-Netze)

Bei dieser Verbindungsstruktur breitet sich die Aktivierung in den Neuronen vorwärts gerichtet von der Eingabeschicht über die versteckten Schichten zur Ausgabeschicht aus. Dabei existiert keine Verbindung, die von einem Neuron direkt oder über zwischengeschaltete Neuronen wieder zum gleichen Neuron zurückführt, s. Bild 2.11.

[6] n-stufig = n Schichten trainierbarer Verbindungen bei $n+1$ Neuronenschichten

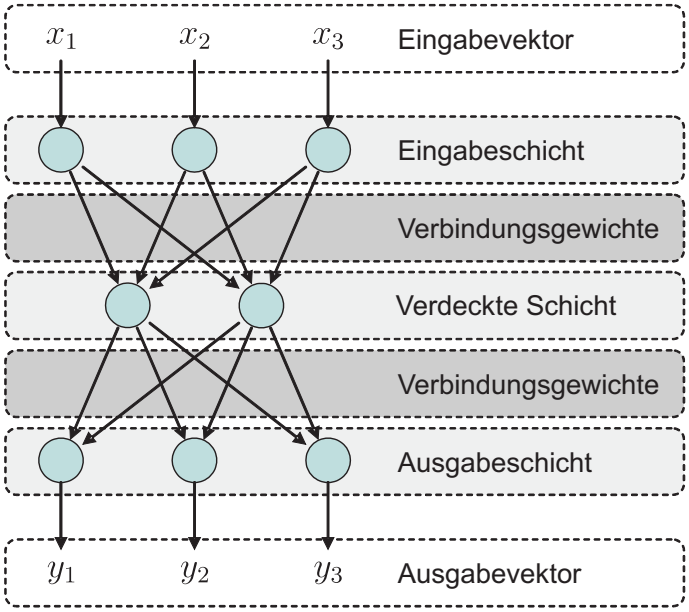


Bild 2.10: Zweistufiges neuronales Netz mit 3-2-3 Topologie

Mathematisch betrachtet stellt diese Topologie einen azyklischen Grafen dar. Während bei *ebenenweise verbundenen* Feedforward-Netzen, wie in Bild 2.10 dargestellt, nur Verbindungen von einer Schicht zur nächsten existieren, gibt es bei *allgemeinen* Feedforward-Netzen auch Verbindungen, die mehrere Schichten überspringen können, s. Bild 2.11. Diese Verbindungen werden in der Literatur als *Shortcuts* bezeichnet.

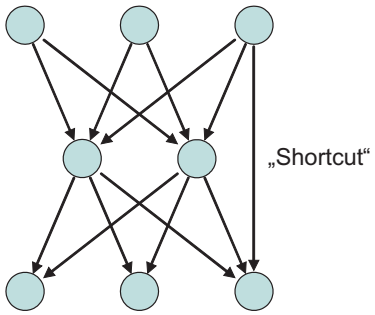


Bild 2.11: Feedforward-Netz mit Shortcut

Netze mit Rückkopplung (Feedback-Netze)

Bei dieser Netztopologie ist die Richtung der Aktivierungsausbreitung nicht vorgeschrieben. Die Aktivierung kann zwischen den Neuronen sowohl hin als auch zurück gegeben werden (Rückkopplung). Man unterscheidet drei Arten der Rückkopplung, s. Bild 2.12:

- Netze mit direkter Rückkopplung (*direct feedback*) erlauben es, dass ein Neuron seine eigene Aktivierung über eine Verbindung von seinem Ausgang zu

seinem Eingang verstärkt oder abschwächt. Solche Neuronen erreichen häufig die Grenzzustände ihrer Aktivierung.

- Bei Netzen mit indirekter Rückkopplung (*indirect feedback*) wird Aktivierung von nachfolgenden Schichten auf vorgelagerte Schichten zurückgegeben. Mit dieser Art der Rückkopplung will man eine Aufmerksamkeitssteuerung auf bestimmte Bereiche von Eingabeneuronen durch das Netz erreichen.
- Netze mit Rückkopplung innerhalb einer Schicht (*lateral feedback*) werden oft für Wettbewerbsaufgaben eingesetzt, bei denen nur ein einziges Neuron in ein und derselben Schicht aktiv werden soll. Das Neuron mit der stärksten Aktivierung hemmt dabei die anderen Neuronen der Schicht, während es sich oft selbst durch eine direkte Rückkopplung stärkt.

Zusätzlich lassen sich Feedback-Netze noch in Netze mit *symmetrischen* bzw. *asymmetrischen* Verbindungen unterteilen, je nachdem ob die Verbindungsgewichte in beide Richtungen gleich bzw. unterschiedlich sind.

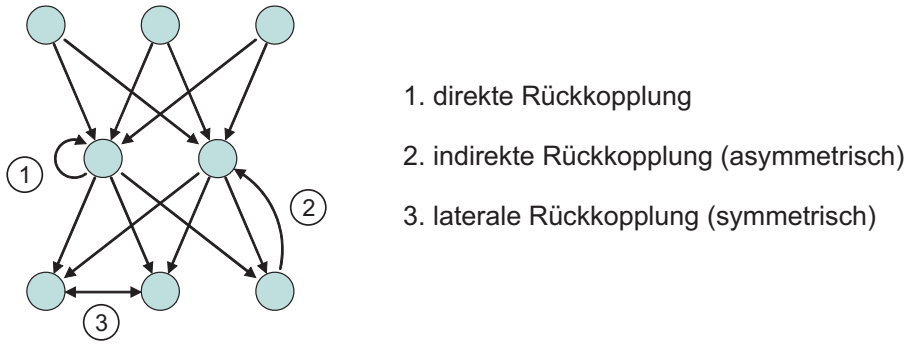


Bild 2.12: Feedback-Netz mit Rückkopplung

Eine besondere Art der Rückkopplung findet man bei den *vollständig verbundenen Netzen*, die auch unter dem Namen Hopfield-Netze bekannt geworden sind. In diesen Netzen existieren Verbindungen zwischen allen Neuronen. Es gibt allerdings zwei Restriktionen: Die Topologie erlaubt keine direkte Rückkopplung und alle Verbindungen müssen symmetrisch sein.

Das für diese Arbeit entwickelte Verfahren zur Simulation neuronaler Netze ist in der Lage, beliebige Netzwerktopologien zu realisieren. Allerdings kommen zur Approximation von Pumpen- bzw. Turbinengeometrien im grafischen Entwurfssystem ausschließlich ebenenweise verbundene Feedforward-Netze zum Einsatz. Eine detaillierte Beschreibung für die Verwendung und Programmierung der unterschiedlichen Netztopologien findet man z.B. bei Kruse et al. [24] und Sauer [41].

2.3.3 Der Lernprozess

Künstliche neuronale Netze haben gegenüber konventionellen Berechnungsprogrammen den Vorteil, eine gegebene Aufgabe weitgehend selbständig aus Beispielen zu

erlernen. In diesem Abschnitt soll deshalb ein allgemeiner Überblick über den Lernprozess in neuronalen Netzen gegeben werden.

2.3.3.1 Definition des Begriffs „Lernen“

Die Informationsverarbeitung in neuronalen Netzen bedeutet in vielen Fällen die Abbildung von Mustern, s. Bild 2.13. Dabei gehört zu jedem Eingabemuster das entsprechende Ausgabemuster. Ein optimal beschaffenes Netzwerk wird zu allen Eingabemustern korrekt das zugehörige Ausgabemuster produzieren. Ob es dazu in der Lage ist, hängt allerdings nicht nur von der gewählten Netztopologie ab.

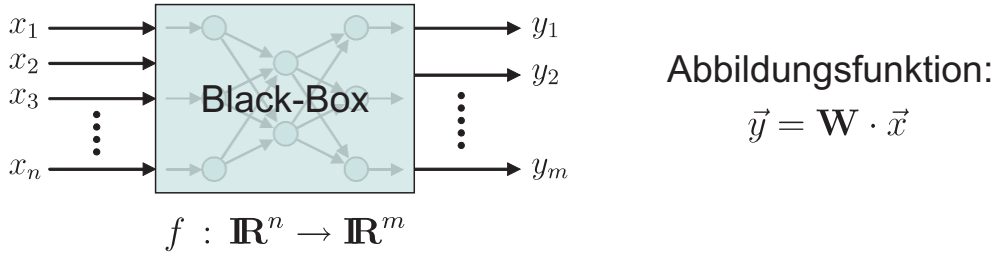


Bild 2.13: Abbildung von Mustern in neuronalen Netzen

Führt die Abbildung eines bestimmten Eingabemusters nicht zum gewünschten Ausgabemuster, so muss die Informationsausbreitung im Netz verändert werden. Im Wesentlichen kann diese Veränderung durch die folgenden Mechanismen realisiert werden:

1. Veränderung der Anzahl Neuronen im Netzwerk (Topologie-Aspekt),
2. Entwicklung neuer bzw. Entfernung existierender Neuronenverbindungen (Topologie-Aspekt),
3. Modifikation der Verbindungsstärken (Veränderung der Gewichte),
4. Modifikation des Aktivierungszuflusses (Schwellwert),
5. Modifikation der Aktivierungs- bzw. Ausgabefunktion.

Die Anwendung dieser Veränderungsmechanismen bezeichnet man, wie auch schon in Kap. 2.3.1 erwähnt, als *Lernen* bzw. *Trainieren*. Das Lernen durch Veränderung der Gewichte ist bei der Simulation künstlicher neuronaler Netze die am häufigsten verwendete Methode. Erst in letzter Zeit haben auch Verfahren, die eine Veränderung der Topologie beeinhalten, eine zunehmende Bedeutung erlangt.

Mit dem für diese Arbeit entwickelten Simulator ist das Lernen durch Veränderung der Netztopologie prinzipiell möglich, allerdings ist diese Lernmethode nicht Gegenstand des implementierten Lernalgorithmus. Eine interaktive Topologieänderung kann im Entwurfsbaustein aus diesem Grund nur vor oder nach dem Lernen durchgeführt werden. Die weiteren Betrachtungen schließen den Topologie-Aspekt beim Trainieren künstlicher neuronaler Netze aus.

2.3.3.2 Lernen in künstlichen neuronalen Netzen

Die Informationsausbreitung wird im Wesentlichen durch die gewichteten Verbindungen zwischen den Neuronen beeinflusst. Bei einer gegebenen Netzwerkarchitektur müssen die für eine fehlerfreie Abbildung von Mustern notwendigen Gewichte aber erst bestimmt werden. Bei der üblicherweise hohen Zahl von Verbindungen in neuronalen Netzwerken kann die günstige Kombination von Gewichtswerten in der Regel nicht in einem einzigen Schritt festgelegt werden, sondern muss mittels eines iterativen Verfahren berechnet werden. Hierfür durchläuft ein neuronales Netz abwechselnd zwei unterschiedliche Phasen, wie Bild 2.14 zeigt.

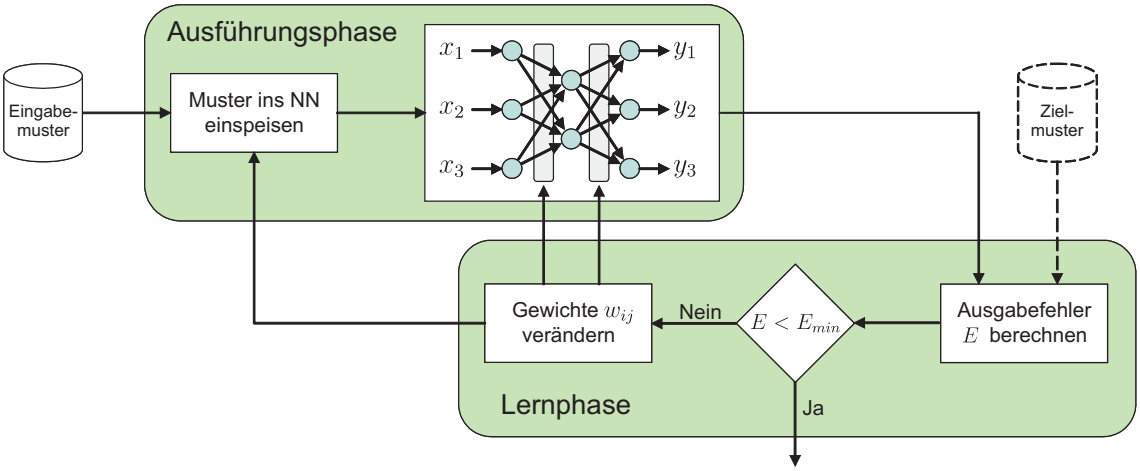


Bild 2.14: Lernprozess in einem neuronalen Netz

Ausführungsphase (Recall Mode)

In dieser Phase werden dem Netzwerk Eingabemuster präsentiert. Das Netz produziert abhängig von seinen Gewichten das entsprechende Ausgabemuster, wobei die Qualität der Ausgabe entscheidend von der Anzahl und der Qualität der vorangegangenen Lernphasen abhängt.

Lern-/Trainingsphase (Learning Mode)

In dieser Phase werden die Gewichte des neuronalen Netzes modifiziert. Mit Hilfe einer fest vorgegebenen Vorschrift, der *Lernregel*, werden die Verbindungsgewichte so verbessert, dass der Ausgabefehler des Netzes in der nächsten Ausführungsphase geringer wird.

Ausführungs- und Lernphase wechseln sich gegenseitig so lange ab, bis der Ausgabefehler für alle zu erlernenden Muster unter einen vorgegebenen Wert sinkt.

Hinsichtlich der Art des Lernens definiert man drei unterschiedliche Lernparadigmen:

- **Überwachtes Lernen** (*supervised learning*)
  Beim überwachten Lernen gibt es einen „externen“ Lehrer, der dem Netzwerk zum Vergleich die richtige Lösung anbietet. Die Selbstmodifikation orientiert

sich dann an der Differenz der Netzwerkausgabe zur erwarteten Ausgabe. Diese Technik setzt allerdings die Existenz von Trainingsdaten voraus, die aus Eingabe- und Zielmusterdaten bestehen. Eine sehr häufig verwendete Lernregel beim überwachten Lernen ist die *Delta-Regel*, auch *Widrow-Hoff-Regel* genannt, s. Rumelhart und McClelland [40]:

$$\Delta w_{ij} = \eta \cdot o_i \cdot (t_j - o_j) = \eta \cdot o_i \cdot \delta_j \ . \tag{2.49}$$

Hierbei ist Δw_{ij} die Änderung des Gewichtes w_{ij} zwischen dem Vorgängerneuron i und dem Nachfolgerneuron j, η ein Proportionalitätsfaktor, der in der Literatur als *Lernrate* bezeichnet wird, und δ_j die Differenz des Ausgabewerts o_j zum gewünschten Zielwert t_j.

- **Bestärkendes Lernen** (*reinforcement learning*)
  Bei diesem Lernparadigma tritt anstelle des Lehrers der Bewerter. Dem Netzwerk werden lediglich Informationen über die Qualität des Ergebnisses mitgeteilt, also ob die Netzwerkausgabe korrekt oder inkorrekt ist. Das gewünschte Zielmuster liegt dem Netz als Vergleichsmöglichkeit nicht vor.

- **Unüberwachtes Lernen** (*unsupervised learning*)
  Das unüberwachte Lernen benötigt weder Lehrer noch Bewerter. Das Netzwerk bildet selbstständig eine interne Repräsentation der Eingabedaten in Form von Ähnlichkeitsklassen. Aus diesem Grund spricht man bei diesen Netzwerken auch von *selbst-organisierenden* Netzen. Der Abgleich der Gewichte erfolgt nach der *Hebbschen Lernregel*:

  $$\Delta w_{ij} = \eta \cdot o_i \cdot a_j \ . \tag{2.50}$$

  Diese besagt, dass die Verbindung zwischen zwei Neuronen i und j verstärkt wird, wenn beide gleichzeitig aktiviert sind. Hierbei gehen in die mathematische Form der Hebbschen Regel sowohl die Ausgabe o_i der Vorgängerzelle als auch der Aktivierungszustand a_j der Nachfolgerzelle ein.

Bei dem für diese Arbeit entwickelten Lernalgorithmus handelt es sich um ein überwachtes Lernverfahren für mehrstufige Feedforward-Netze, das unter dem Namen *Backpropagation* bekannt ist. Die dabei eingesetzte Backpropagation-Lernregel ist eine Verallgemeinerung der oben erwähnten Delta-Regel und wird in Kap. 3 ausführlicher behandelt.

Eine detaillierte Beschreibung und Beispiele zu bestärkenden bzw. unüberwachten Lernverfahren sind u.a. bei Kruse et al. [24] und Zell [68] zu finden.

2.3.3.3 Anpassen von Schwellwerten durch Bias-Neuronen

Wie beim biologischen Äquivalent haben auch die künstlichen Neuronen in den meisten Fällen einen Schwellwert (*Bias*), der die Aktivierung des Neurons regelt. Oftmals

kann es während des Lernens erforderlich sein, diese Schwelle zu verändern. Ein Beispiel hierfür stellt das Trainieren des Nullvektors in einstufigen Netzen dar. Da jedes Neuron der Eingabeschicht den Wert $o_i = 0$ an die Ausgabeschicht weitergibt, berechnet sich die Gewichtsänderung nach Gl. 2.49 zu $\Delta w_{ij} = \eta \cdot 0 \cdot \delta_j = 0$. Ein Trainieren der Gewichte ist in diesem Fall nicht möglich. Durch das Anpassen des erwähnten Schwellwertes lässt sich die Neuronenaktivität dennoch manipulieren, um somit das Lernziel zu erreichen.

In künstlichen neuronalen Netzen wird dieser einstellbare Schwellwert unterschiedlich realisiert, s. Bild 2.15. Zum einen kann er gemäß Gl. 2.46 als Aktivierungszufluss in die Aktivierungsfunktion eingehen. Damit ergibt sich folgender Aktivierungszustand eines Neurons j:

$$a_j(t+1) = f_{act}(net_j(t+1) - \theta_j(t+1),\, a_j(t)) \ . \tag{2.51}$$

Andererseits lässt sich der Schwellwert auch mit Hilfe eines sog. Bias-Neurons über ein zusätzliches Verbindungsgewicht $w_{0j} = -\theta_j$ berücksichtigen. Das Bias-Neuron liefert dabei immer den Ausgabewert 1. Im Gegensatz zu Gl. 2.51 ändert sich hierbei nicht die Aktivierungsfunktion, sondern der Nettoinput nach Gl. 2.45:

$$net_j = (\sum_i o_i \cdot w_{ij}) + 1 \cdot w_{0j} = (\sum_i o_i \cdot w_{ij}) - \theta_j \ . \tag{2.52}$$

Mathematisch gesehen sind beide Varianten gleich. Lediglich bei der Programmierung erweisen sich Bias-Neuronen als nützlicher, da lediglich Verbindungsgewichte trainiert werden müssen, und man sich somit auf ein einziges Lernverfahren beschränken kann. Vor allem beim implementierten Backpropagation-Verfahren werden Schwellwerte wie Gewichte behandelt, um die Herleitung des Verfahrens bzw. den Lernalgorithmus nicht unnötig kompliziert zu machen.

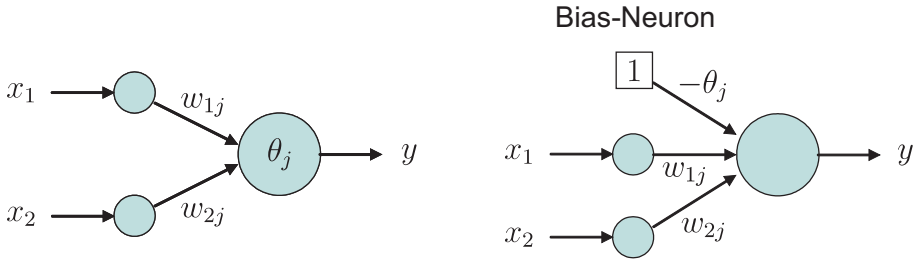


Bild 2.15: Netzwerk mit Schwellwert (links) und Bias-Neuron (rechts)

Kapitel 3

Backpropagation Netzwerke

In diesem Kapitel wird das Lernverfahren vorgestellt, das in dem für diese Arbeit entwickelten Entwurfssystem zum Trainieren von Pumpen- und Turbinengeometrien eingesetzt wird. Das unter dem Namen *Backpropagation* bekannt gewordene Verfahren wurde 1974 von Werbos [63] entwickelt, erlangte allerdings erst im Jahre 1986 durch die Arbeiten von Rumelhart und McClelland [40] seine große Bedeutung. Bis heute zählt Backpropagation zu einem der bekanntesten Lernverfahren für mehrstufige Feedforward-Netze, da es sich im Vergleich zu anderen Lernverfahren um ein sehr schnelles und robustes Verfahren handelt.

3.1 Aufbau von Backpropagation Netzwerken

Minsky und Papert [27] zeigten, dass einstufige Feedforward-Netze (*Perzeptron-Modelle*), die lediglich aus einer Eingabe- und einer Ausgabeschicht bestehen, nur bedingt zum Abbilden von Mustern geeignet sind. Dies liegt darin begründet, dass ein einstufiges Perzeptron ausschließlich *linear trennbare* Funktionen repräsentieren kann. So ist es zwar möglich, die logischen Verknüpfungen AND und OR abzubilden, nicht aber die logische Verknüpfung XOR, s. Bild 3.1. Ausgehend von Gl. 2.52 liefert ein einstufiges Perzeptron mit zwei Eingabe- und einem Ausgabeneuron, vgl. Bild 2.15, genau dann den Wert 1, wenn gilt:

$$net_j = o_1 \cdot w_{1j} + o_2 \cdot w_{2j} = x_1 \cdot w_{1j} + x_2 \cdot w_{2j} \geq \theta_j \ . \tag{3.1}$$

Für einen konstanten Schwellwert θ_j ergibt sich somit eine Gerade in der durch x_1 und x_2 aufgespannten Ebene, wie in Bild 3.1 für die logischen Verknüpfungen AND und OR dargestellt. Dagegen ist eine Klassifizierung der Netzwerkausgabe der logischen Verknüpfung XOR durch die Trennung des Eingaberaumes mit nur einer einzigen Geraden nicht möglich, d.h. das Problem ist nicht linear trennbar.

Im Gegensatz zu den einstufigen Feedforward-Netzen, die in der Regel nur als binäre Musterassoziatoren eingesetzt werden, besitzen Backpropagation Netzwerke drei oder mehr Neuronenschichten und sind in der Lage auch komplexe Muster bzw.

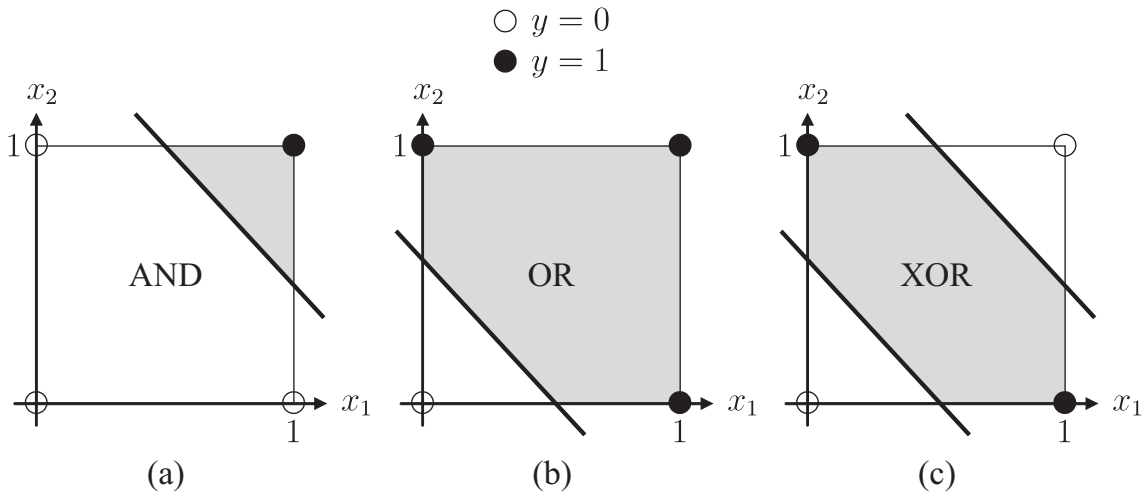


Bild 3.1: Lineare Trennbarkeit logischer Funktionen: Die Klassifizierung der Ausgabewerte der logischen Verknüpfung (a) AND bzw. (b) OR lässt sich durch die Separation des Eingaberaumes mit Hilfe einer einzigen Geraden realisieren. Bei der logischen Verknüpfung (c) XOR können die Ausgabewerte dagegen nicht linear voneinander getrennt werden

nichtlineare Zusammenhänge abzubilden. Dabei dient die erste Neuronenschicht zur Aufnahme von externen Eingabewerten, während die letzte Schicht für die Wiedergabe von Ausgabewerten vorgesehen ist. Die Neuronen der verdeckten Zwischenschichten sind verantwortlich für die interne Repräsentation von Eingabewerten und ermöglichen es, Probleme zu lösen, die eine interne Aufbereitung von Eingabewerten erfordern. So kann z.B. das oben erwähnte XOR-Problem durch Hinzufügen eines einzigen verdeckten Neurons gelöst werden, s. RUMELHART UND MCCLELLAND [40]. Backpropagation Netzwerke gehören zu den vollständig ebenenweise verbundenen Feedforward-Netzen, d.h. jedes Neuron eines Neuronenschicht ist mit jedem Neuron der vorgelagerten und nachfolgenden Schicht verbunden. Shortcut-Verbindungen über mehr als zwei aufeinanderfolgende Schichten hinweg sind ebenfalls zulässig, vgl. Kap. 2.3.2.2. Sinnvollerweise ist zudem jedes Neuron der Ausgabeschicht und der verdeckten Zwischenschichten mit einem Bias-Neuron verbunden, um einen Schwellwert zu simulieren, der unabhängig von der Aktivierungsfunktion verändert werden kann. In Bild 3.2 ist ein typisches Backpropagation Netzwerk dargestellt.

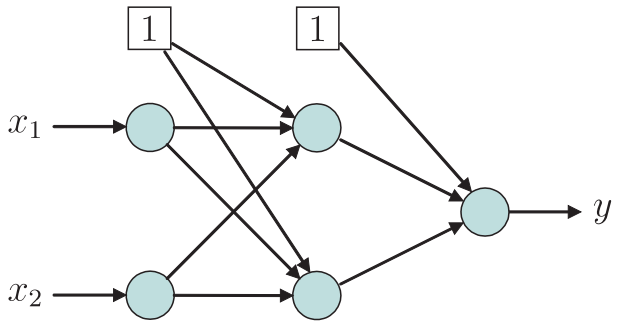


Bild 3.2: Zweistufiges Backpropagation Netzwerk mit 2-2-1 Topologie

3.2 Informationsverarbeitung

Die Informationsausbreitung in Backpropagation Netzwerken erfolgt wie bereits in Kap. 2.3.2 beschrieben. Über die Eingabeschicht wird ein beliebiges Eingabemuster $\vec{x} = (x_1, x_2, \ldots, x_n)$ ins Netz eingespeist, wobei die Ausgabe eines Neurons i der Eingabeschicht identisch mit dem jeweiligen Eingabewert des Musters ist:

$$o_i = x_i \ . \tag{3.2}$$

Eine Weiterverarbeitung der Eingabewert findet also nur in den Neuronen der verdeckten Schichten und der Ausgabeschicht statt. Hierfür seien in den folgenden Ausführungen i und j die Neuronen zweier miteinander verbundener Neuronenschichten, wobei das Neuron i in der dem Neuron j vorgelagerten Schicht liegt. Der durch die Propagierungsfunktion bestimmte Nettoinput net_j des Neurons j berechnet sich gemäß Gl. 2.52 zu:

$$net_j = (\sum_i o_i \cdot w_{ij}) - \theta_j \ . \tag{3.3}$$

Zur Vereinfachung der Herleitung der Backpropagation-Regel wird der Schwellwert $-\theta_j$ in den folgenden Betrachtungen durch Erweiterung des Indexbereichs i als Produkt $o_0 \cdot w_{0j}$ in die Summation aufgenommen, wobei o_0 als Bias-Neuron stets den Wert 1 liefert.

Mit dem Nettoinput net_j lässt sich anschließend der aktuelle Aktivierungszustand a_j des Neurons j berechnen. Es ist für mehrstufige neuronale Netze nicht sinnvoll, lineare Aktivierungsfunktionen zu verwenden, da die gleiche Abbildungsfunktion auch von einem einstufiges Netzwerk realisiert werden kann, s. Zell [68]. Im Allgemeinen wird für Backpropagation Netzwerke eine *sigmoide* Aktivierungsfunktion wie in Bild 3.3 gewählt, woraus sich mehrere Vorteile ergeben. Durch den asymptotischen Verlauf der sigmoiden Aktivierungsfunktion wird z.B. sichergestellt, dass sich ein extremes Verhalten einzelner Neuronen durch eine sehr starke Aktivierung nicht über das gesamte Netz fortpflanzen kann. Zudem besitzen sigmoide Aktivierungsfunktionen im Bereich um den Arbeitspunkt θ_j die größte Steigung, wodurch das Netzwerk in der Lage ist, auf schwache Signale wesentlich sensibler zu reagieren als auf starke. Als Letztes sei noch zu bemerken, dass sigmoide Funktionen anders als beispielsweise die binäre Schwellwertfunktion an jeder Stelle stetig und damit überall differenzierbar sind, was für das Backpropagation Verfahren eine zwingende Voraussetzung ist.

Die bei Backpropagation Netzwerken am häufigsten verwendeten Aktivierungsfunktionen sind derzeit die *logistische* Aktivierungsfunktion, wie in Bild 3.3 dargestellt, und die Funktion *tangens hyperbolicus*. Das für diese Arbeit programmierte Backpropagation Verfahren beschränkt sich auf die Verwendung der logistischen Aktivierungsfunktion, deren Wertebereich im offenen Intervall $]0, 1[$ liegt und die folgende Form hat:

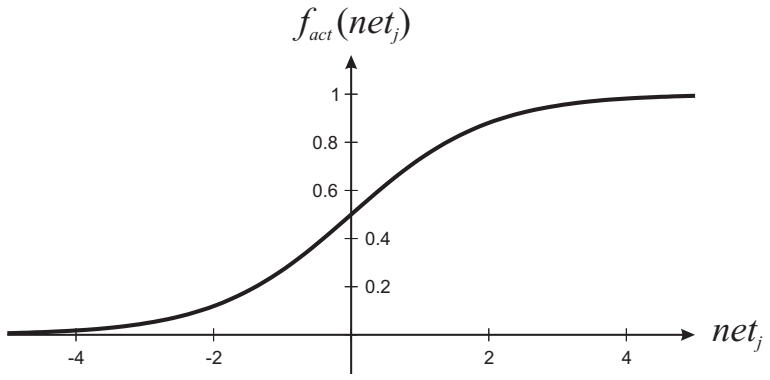


Bild 3.3: Sigmoide Aktivierungsfunktion (logistische Funktion)

$$f_{act}(net_j) = \frac{1}{1 + e^{-net_j}} \ . \tag{3.4}$$

Die Ableitung der logistischen Funktion wird ebenfalls für das Backpropagation Verfahren benötigt und berechnet sich zu:

$$\begin{aligned} \frac{df_{act}(net_j)}{dnet_j} &= -\frac{1}{(1 + e^{-net_j})^2} \cdot (-e^{-net_j}) \\ &= \frac{1}{1 + e^{-net_j}} \cdot \frac{e^{-net_j}}{1 + e^{-net_j}} \\ &= f_{act}(net_j) \cdot \frac{1 + e^{-net_j} - 1}{1 + e^{-net_j}} \\ &= f_{act}(net_j) \cdot (1 - f_{act}(net_j)) \ . \end{aligned} \tag{3.5}$$

Üblicherweise wird bei Backpropagation Netzwerken auf eine zusätzliche Ausgabefunktion f_{out} verzichtet, wodurch die Ausgabe eines Neurons j gleich dem inneren Aktivierungszustand ist:

$$o_j = f_{act}(net_j) = \frac{1}{1 + e^{-net_j}} \ . \tag{3.6}$$

Sind alle Ausgabewerte der Neuronen einer Schicht berechnet, so wird mit der nächsten Neuronenschicht fortgefahren. Dieser Prozess, der unter dem Namen *Forward-propagation* bekannt ist, wird solange wiederholt, bis die Ausgabe der letzten Schicht bekannt ist. Der Algorithmus wechselt dann von der Ausführungsphase in die Lernphase, in der das berechnete Ausgabemuster des Netzes mit dem vorgegebenen Zielmuster verglichen wird, und die Verbindungsgewichte w_{ij} zwischen den Neuronen entsprechend verändert werden.

3.3 Das Backpropagation Lernverfahren

In diesem Abschnitt soll die Arbeitsweise des von RUMELHART UND MCCLELLAND [40] vorgestellten Lernverfahren für Backpropagation Netzwerke beschrieben werden.

3.3.1 Prinzip des Lernverfahrens

Ziel des Backpropagation Lernverfahrens ist es, eine Kombination von Verbindungsgewichten w_{ij} zu finden, mit denen das Netzwerk die vorgegebene Menge von Eingabemustern auf die entsprechenden Zielmuster möglichst frei abbilden kann. Hierbei ist der Gesamtfehler E über alle Lernmuster p ein Maß für die Leistungsfähigkeit eines Netzwerkes und es gilt:

$$E = \sum_p E_p \tag{3.7}$$

mit

$$E_p = \frac{1}{2} \sum_j \left(t_j - o_j\right)^2 \ , \tag{3.8}$$

wobei E_p der gesamte quadratische Fehler für ein zu lernendes Muster p (*pattern*) ist, der sich aus der Summe der Lernfehler der einzelnen Ausgabeneuronen ergibt. Den Lernfehler eines Neurons j der Ausgabeschicht erhält man dabei aus der quadratischen Differenz zwischen dem Zielwert t_j des Musters und dem tatsächlichen Ausgabewert o_j. Durch das Verändern einzelner Verbindungsgewichte w_{ij} ändert sich der Ausgabewert o_j der Ausgabeneuronen und damit auch der Gesamtfehler des Netzwerkes. Somit lässt sich der Gesamtfehler als Funktion der Gewichte wie in Bild 3.4 darstellen:

$$E = E(w_{ij}) = E(w_{11}, w_{21}, \ldots, w_{nm}) \ . \tag{3.9}$$

Zur Minimierung des Gesamtfehlers verwendet Backpropagation ein Gradientenabstiegsverfahren. Die Änderung Δw_{ij} eines Gewichtes ist dabei proportional zum negativen Gradienten der Fehlerfunktion mit der Lernrate η als Proportionalitätsfaktor:

$$\Delta w_{ij} = -\eta \cdot \frac{\partial E}{\partial w_{ij}} \ . \tag{3.10}$$

Mit dem Ziel, den Gesamtfehler E des Netzwerkes zu minimieren, liegt es nahe, zuerst die Verbindungsgewichte für jedes einzelne zu lernende Muster zu bestimmen, die den Lernfehler E_p des Musters reduzieren.

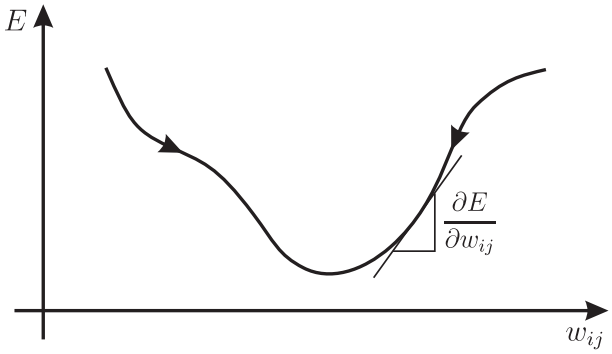


Bild 3.4: Fehler eines Netzwerkes als Funktion eines Gewichtes w_{ij}

3.3.2 Delta-Regel

Ausgehend von Gl. 3.10 und unter Anwendung der Kettenregel folgt für die Gewichtsänderung Δw_{ij} zur Reduzierung des Fehlers E_p eines Musters p:

$$\Delta w_{ij} = -\eta \cdot \frac{\partial E_p}{\partial w_{ij}} = -\eta \cdot \frac{\partial E_p}{\partial net_j} \cdot \frac{\partial net_j}{\partial w_{ij}} \ . \tag{3.11}$$

Aus der Definition des Fehlersignals δ_j für ein Neuron j

$$\delta_j = -\frac{\partial E_p}{\partial net_j} \tag{3.12}$$

und der partiellen Ableitung

$$\frac{\partial net_j}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}} (\sum_i o_i \cdot w_{ij}) = o_i \tag{3.13}$$

folgt für Gl. 3.11:

$$\Delta w_{ij} = \eta \cdot o_i \cdot \delta_j \ . \tag{3.14}$$

Unter der Voraussetzung einer linearen Aktivierungsfunktion, d.h. $o_j = net_j$, ergibt sich das Fehlersignal δ_j im Zusammenhang mit Gl. 3.8:

$$\delta_j = -\frac{\partial E_p}{\partial net_j} = -\frac{\partial E_p}{\partial o_j} = (t_j - o_j) \ , \tag{3.15}$$

und man erhält die bereits in Kap. 2.3.3.2 erwähnte *Delta-Regel*:

$$\Delta w_{ij} = \eta \cdot o_i \cdot (t_j - o_j) \ . \tag{3.16}$$

3.3.3 Backpropagation-Regel

Für Backpropagation Netzwerke kann die in Gl. 3.16 dargestellte Delta-Regel aus zwei Gründen nicht übernommen werden. Zum einen verwenden Backpropagation Netzwerke nichtlineare Aktivierungsfunktionen und zum anderen ist für Neuronen der verdeckten Schichten kein Zielwert t_j definiert. Die Delta-Regel muss deshalb in eine verallgemeinerte Delta-Regel transformiert werden, die unter dem Namen *Backpropagation-Regel* bekannt ist.

Ausgehend von Gl. 3.12 kann das Fehlersignal δ_j eines Neurons j unter Anwendung der Kettenregel folgendermaßen umgeformt werden:

$$\delta_j = -\frac{\partial E_p}{\partial net_j} = -\frac{\partial E_p}{\partial o_j} \cdot \frac{\partial o_j}{\partial net_j} \ . \tag{3.17}$$

Unter der Voraussetzung einer nichtlinearen und differenzierbaren Aktivierungsfunktion folgt für die zweite partielle Ableitung des Produkts in Gl. 3.17 wegen $o_j = f_{act}(net_j)$:

$$\frac{\partial o_j}{\partial net_j} = \frac{\partial f_{act}(net_j)}{\partial net_j} = f'_{act}(net_j) \ . \tag{3.18}$$

Zur Bestimmung der ersten partiellen Ableitung aus Gl. 3.17 muss unterschieden werden, ob es sich bei dem betrachteten Neuron j um ein Neuron der Ausgabeschicht oder einer verdeckten Schicht handelt. Im ersten Fall lässt sich der fehlende Ausdruck direkt aus der in Gl. 3.8 dargestellten Fehlerfunktion E_p ableiten, da sowohl der Zielwert t_j als auch die Ausgabe o_j von Neuron j bekannt ist:

$$-\frac{\partial E_p}{\partial o_j} = (t_j - o_j) \ . \tag{3.19}$$

Insgesamt berechnet sich das Fehlersignal δ_j für Neuronen der Ausgabeschicht durch Einsetzen der Gln. 3.18 und 3.19 in Gl. 3.17 in der folgenden Weise:

$$\delta_j = f'_{act}(net_j) \cdot (t_j - o_j) \ . \tag{3.20}$$

Im zweiten Fall kann die erste partielle Ableitung aus Gl. 3.17 nur indirekt berechnet werden, da dem Neuron j einer verdeckten Zwischenschicht zum Vergleich kein Zielwert t_j zur Verfügung steht. Mit Hilfe der Kettenregel und der Tatsache, dass Neuron j bei allen verbundenen Nachfolgeneuronen einen Fehler verursacht, lässt sich der gesuchte Ausdruck wie folgt ersetzen:

$$-\frac{\partial E_p}{\partial o_j} = -\sum_k \frac{\partial E_p}{\partial net_k} \cdot \frac{\partial net_k}{\partial o_j} \ , \tag{3.21}$$

wobei sich k auf ein Neuron bezieht, das mit dem betrachteten Neuron j verbunden ist und in der nachfolgenden Neuronenschicht liegt. Aus der Definition des Fehlersignals gemäß Gl. 3.12 und der partiellen Ableitung

$$\frac{\partial net_k}{\partial o_j} = \frac{\partial}{\partial o_j} \left(\sum_j o_j \cdot w_{jk} \right) = w_{jk} \tag{3.22}$$

erhält man durch Einsetzen in Gl. 3.21:

$$- \frac{\partial E_p}{\partial o_j} = \sum_k \delta_k \cdot w_{jk} \ . \tag{3.23}$$

Dies bedeutet, dass man den Lernfehler des verdeckten Neurons j für ein Muster p aus den Fehlersignalen δ_k aller Nachfolgeneuronen k und den entsprechenden Verbindungsgewichten w_{jk} berechnen kann.

Für alle Neuronen der verdeckten Zwischenschichten lässt sich somit das Fehlersignal δ_j durch Einsetzen der Gln. 3.18 und 3.23 mit folgender Formel berechnen:

$$\delta_j = f'_{act}(net_j) \cdot \sum_k \delta_k \cdot w_{jk} \ . \tag{3.24}$$

Die Berechnung des Fehlersignals erfolgt hierbei entgegengesetzt der vorwärts gerichteten Informationsausbreitung der Eingabewerte (*Forwardpropagation*), also von der Ausgabeschicht über die verdeckten Zwischenschichten zur Eingabeschicht. Aufgrund dieser rückwärts gerichteten Fehlerverarbeitung wird der Lernprozess *Backpropagation* genannt. Ein Überblick des gesamten Backpropagation Verfahrens ist in Bild 3.5 dargestellt.

Da in dem für diese Arbeit programmierten Backpropagation Lernverfahren die logistische Aktivierungsfunktion gemäß Gl. 3.4 eingesetzt wird, ergibt sich für die Ableitung der Aktivierungsfunktion aus Gl. 3.5 wegen $o_j = f_{act}(net_j)$:

$$f'_{act}(net_j) = f_{act}(net_j) \cdot (1 - f_{act}(net_j)) = o_j \, (1 - o_j) \ , \tag{3.25}$$

und die Backpropagation-Regel lässt sich in der folgenden vereinfachten Form zusammenfassen:

$$\Delta w_{ij} = \eta \cdot o_i \cdot \delta_j \tag{3.26}$$

mit

$$\delta_j = \begin{cases} o_j \, (1 - o_j) \cdot (t_j - o_j) & \text{falls } j \text{ Ausgabeneuron ist} \\ o_j \, (1 - o_j) \cdot \sum_k \delta_k \cdot w_{jk} & \text{falls } j \text{ verdecktes Neuron ist} \ . \end{cases} \tag{3.27}$$

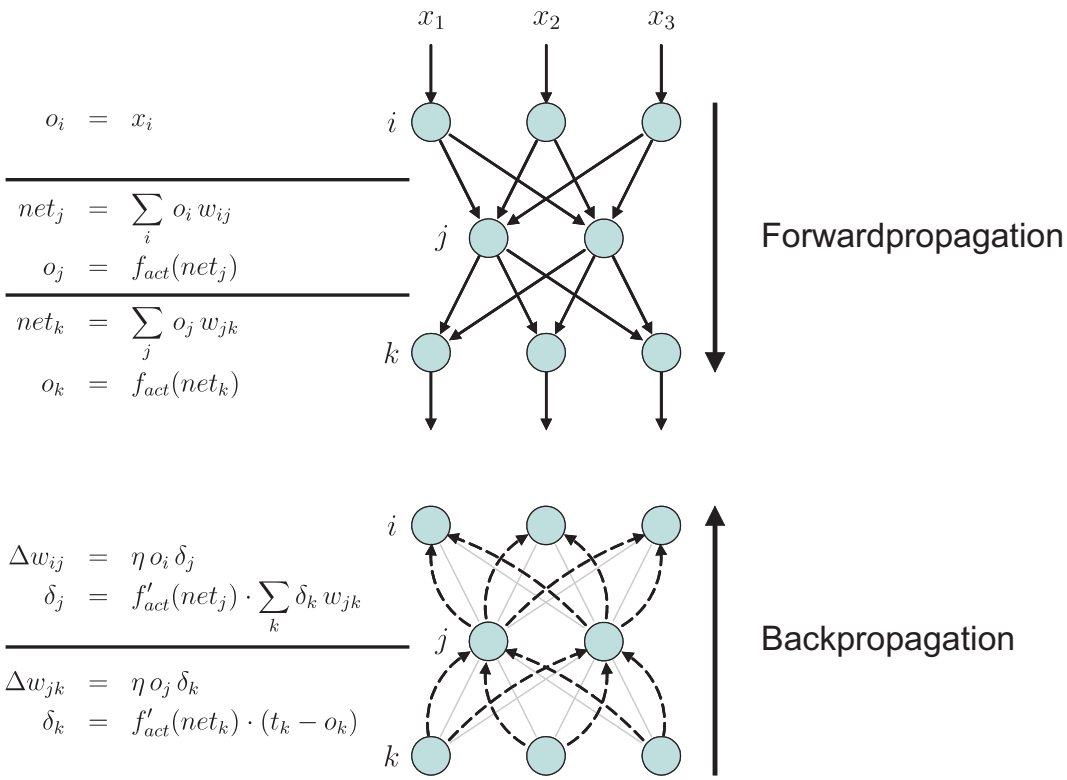


Bild 3.5: Backpropagation am Beispiel eines zweistufigen Netzwerks

Nachdem alle Gewichtsänderungen Δw_{ij} des Netzwerks für ein bestimmtes Lernmuster p bestimmt sind, erfolgt die Anpassung der Verbindungsgewichte w_{ij} durch einfache Addition:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t) \ . \tag{3.28}$$

Unter Verwendung der neuen Gewichte beginnt der Lernprozess mit der Informationsverarbeitung des nächsten Lernmuster von vorne. Diese Art des Lernens wird in der Literatur als *musterweises Lernen* oder *Online-Training* bezeichnet. Haben alle zu lernenden Muster den Lernprozess durchlaufen, so ist der Lernzyklus für eine *Epoche* abgeschlossen. Dieser Prozess wird solange wiederholt, bis der Gesamtfehler E des Netzwerks unter einen vorgegebenen Grenzwert sinkt.

Alternativ zum musterweisen Lernen kann eine Gewichtsanpassung auch erst nach dem Durchlauf aller Lernmuster erfolgen. In diesem Fall spricht man von *epochenweisem Lernen* oder *Batch-* bzw. *Offline-Training*. Hierbei werden lediglich die Gewichtsänderungen Δw_{ij} nach jedem gelernten Muster p aufsummiert, während das Anpassen der Verbindungsgewichte w_{ij} einmalig am Ende des Lernzyklus erfolgt:

$$w_{ij}(t+1) = w_{ij}(t) + \sum_p (\Delta w_{ij}(t))_p \ . \tag{3.29}$$

Die Unterschiede beider Lernverfahren werden in Kap. 3.5.2 ausführlicher beschrieben.

3.4 Probleme von Backpropagation

Da es sich bei Backpropagation um ein Gradientenverfahren und somit um ein lokales Verfahren handelt, d.h. es liegen keine Informationen über die Fehlerfläche insgesamt vor, muss ein Minimum allein aus der Kenntnis der lokalen Umgebung gesucht werden. Dadurch entstehen eine Reihe von Problemen, die im Folgenden beschrieben werden. Zudem werden Lösungsansätze für diese Probleme vorgestellt, die in dem für diese Arbeit entwickelten Verfahren umgesetzt worden sind.

3.4.1 Symmetry Breaking

Dieses Problem bezieht sich auf die Initialisierung der Startgewichte vollständig ebenenweise verbundener Feedforward-Netze. Werden die Verbindungsgewichte zu Beginn des Lernprozesses alle auf den gleichen Wert initialisiert, so lässt sich beweisen, dass sich durch das Lernverfahren Backpropagation keine gänzlich unterschiedlichen Gewichte in einer Verbindungsschicht bilden können, s. Zell [68]. So sind beispielsweise die Gewichte zwischen den Verbindungen eines Neurons der Ausgabeschicht und allen Neuronen der Vorgängerschicht immer gleich groß. Diese sog. *symmetrischen* Verbindungsgewichte prägen sich im gesamten Netz aus und das Backpropagation Verfahren ist nicht in der Lage, diese durch die Initialisierung eingeführte Symmetrie zu brechen.

Eine einfache Lösung dieses Problems liegt in der zufälligen Initialisierung der Gewichtswerte. Dabei sollten die Zufallswerte im Bereich der größten Steigung der Aktivierungsfunktion liegen, damit sich das System schnell adaptiert. In der Regel werden die Gewichte, wie auch im entwickelten Verfahren, im Bereich von -1 bis 1 initialisiert.

3.4.2 Lokale Minima

Ein großes Problem aller Gradientenverfahren ist die Tatsache, in einem lokalen Minimum der Zielfunktion, hier der Fehlerfunktion E, hängen zu bleiben, s. Bild 3.6a. Je größer ein neuronales Netz ist, desto größer ist die Anzahl an Verbindungen zwischen den Neuronen und desto zerklüfteter wird die zugrunde liegende Fehlerfläche. Die Chance, in einem suboptimalen lokalen Minimum zu landen, wird dabei immer wahrscheinlicher. In welchem Umfang sich lokale Minima ausbilden, hängt auch sehr stark von der Anwendung und von der Art der Lernmuster ab. Aus diesem Grund existieren für dieses Problem nur sehr wenige allgemeingültige Verfahren. Die praktische Erfahrung zeigt allerdings, dass das Backpropagation Verfahren durch eine geeignete Wahl der Lernrate η in den meisten Anwendungen ein Minimum findet, das vom Niveau her nah genug am globalen Minimum liegt und zur Musterabbildung akzeptabel ist.

3.4.3 Flache Plateaus

Die treibende Kraft bei Backpropagation ist der Gradient der Fehlerfunktion. Je steiler die Fehlerkurve ist, desto weniger Schritte sind notwendig, um das Minimum zu erreichen. Ist die Fehlerfunktion, wie in Bild 3.6b dargestellt, in einem Bereich nahezu steigungsfrei, so stagniert Backpropagation, d.h. das Lernverfahren braucht extrem viele Iterationsschritte zum Verlassen dieses flachen Plateaus. Im schlimmsten Fall, wenn der Gradient gleich null ist, kommt das Verfahren vollständig zum Erliegen. Leider lässt sich in dieser Situation auch nicht erkennen, ob man sich auf einem flachen Plateau oder in einem lokalen bzw. globalen Minimum befindet.

Dieses Problem lässt sich durch eine kleine Modifikation bei der Anpassung der Gewichte durch den sog. *Momentum-Term* lösen, der in Kap. 3.5.5 ausführlicher beschrieben ist.

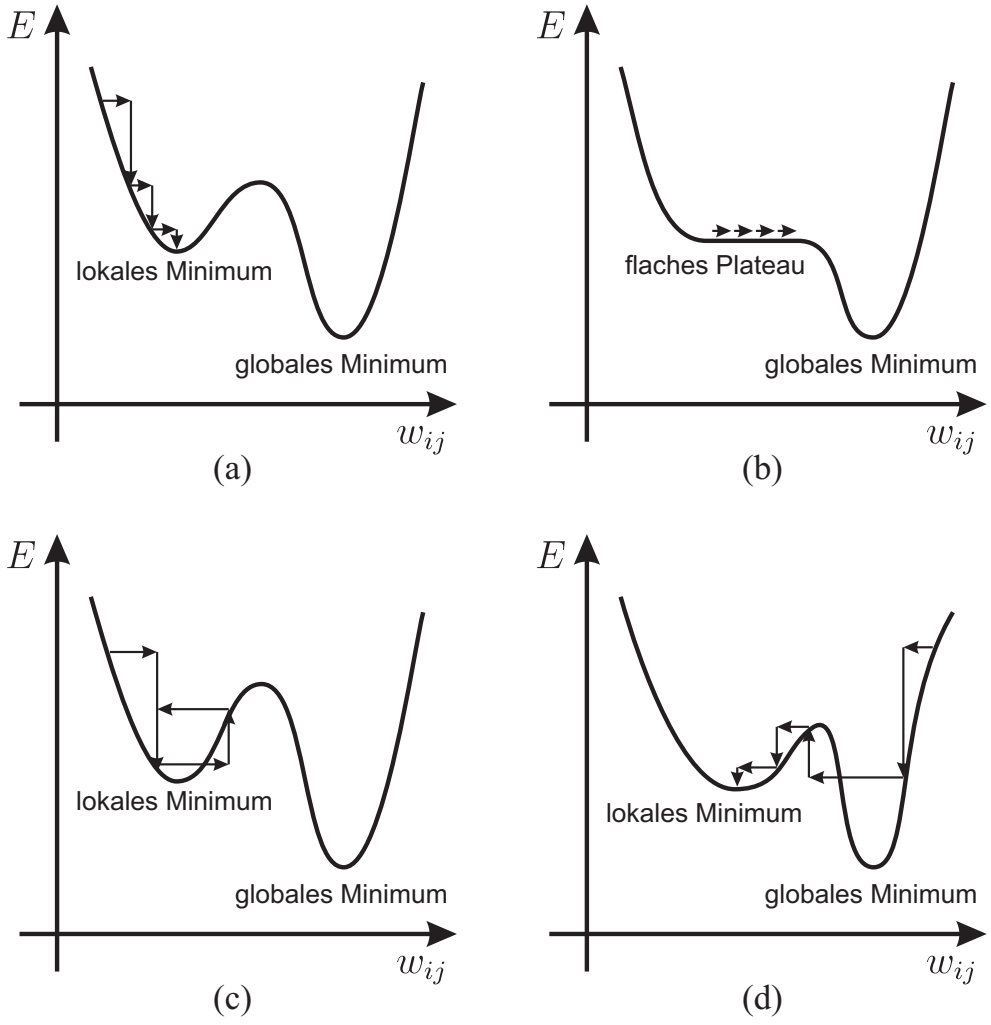


Bild 3.6: Probleme von Backpropagation: (a) lokales Minimum, (b) flaches Plateau, (c) Oszillation, (d) Verlassen guter Minima

3.4.4 Oszillation

Oszillation des Lernverfahrens tritt in Senken der Fehlerfläche durch zu hohe Lernraten oder durch sehr große Gradienten an den Rändern auf. Wie Bild 3.6c zeigt, ist die Gewichtsänderung dabei so groß, dass ein Sprung auf die andere Seite der Senke erfolgt. Ist die Steigung an dieser Stelle betragsmäßig genauso groß, so bewirkt dies wiederum ein Sprung zurück auf die erste Seite.

Durch die Verwendung des bereits im letzten Abschnitt erwähnten Momentum-Terms lässt sich diese Oszillation dämpfen oder sogar eliminieren.

3.4.5 Verlassen guter Minima

Ist die Gewichtsänderung durch einen großen Gradienten bei sehr engen Senken der Fehlerfläche so stark, kann Backpropagation aus einem optimalen Minimum herausspringen und in einem suboptimalen Minimum landen, s. Bild 3.6d. Zwar begünstigt sowohl die Erhöhung der Lernrate sowie die Verwendung des Momentum-Terms das Herausspringen aus einem guten Minimum, allerdings tritt dies in der Praxis nur sehr selten auf.

3.5 Einflussgrößen auf das Lernverhalten

An dem für diese Arbeit programmierten Backpropagation Lernverfahren wurden Modifikationen vorgenommen, die sich auf das Lernverhalten auswirken. Neben den fest implementierten Verbesserungen des Verfahrens, lässt sich das Lernverhalten auch über mehrere variierbare Parameter von außen beeinflussen. Dieser Abschnitt beschäftigt sich deshalb mit den implementierten Einflussgrößen.

3.5.1 Einstellen der Lernrate

Die richtige Wahl der Lernrate η, in der Literatur auch als Lernfaktor bzw. Schrittweite bekannt, ist entscheidend für das Lernverhalten des Algorithmus. Um ein schnelles Absinken des Fehlers zu erreichen, sollte versucht werden, η möglichst groß zu wählen. Große Werte von η bewirken allerdings auch starke Sprünge auf der Fehlerfläche, wobei das globale Minimum übersprungen werden kann oder das Verfahren zu oszillieren beginnt. Zu kleine Werte von η haben den Nachteil, dass das Lernen entweder in einem suboptimalen lokalen Minimum endet oder die Zeitspanne für das Training unakzeptabel lang wird. Da die Wahl der Lernrate sowohl von dem betrachteten Problem und den Trainingsdaten als auch von der Größe und Topologie des Netzwerkes abhängig ist und die in der Literatur gegebenen Hinweise meist widersprüchlich sind, lässt sich ein optimaler Wert meist nur versuchsweise ermitteln. Bei den für diese Arbeit trainierten Geometriedaten hat sich ein konstanter Wert von $\eta = 0.3$ für alle getesteten Netztopologien als sinnvoll erwiesen. Er wird auch als Standardwert im entwickelten Entwurfssystem vorgeschlagen.

3.5.2 Auswahl des Lernverfahrens

Das implementierte Backpropagation Verfahren sieht eine wahlweise Anpassung der Verbindungsgewichte nach jedem gelernten Musterpaar (*musterweises Lernen*) oder nach einem vollständigen Durchlauf aller zu lernenden Muster (*epochenweises Lernen*) vor, vgl. Kap. 3.3.3. Je nach Auswahl des Lernverfahrens wird das Lernverhalten des Backpropagation Netzwerks unterschiedlich beeinflusst.

Beim musterweisen Lernen bzw. Online-Training erfolgt die Anpassung der Gewichte stets in Richtung des steilsten Abstiegs der Fehlerfunktion des jeweiligen zu lernenden Musters. Um eine nach jedem Lernzyklus wiederkehrende Veränderung der Gewichte, bedingt durch die Reihenfolge der Lernmuster, zu vermeiden, ist es sinnvoll die Reihenfolge der Lernmuster in jedem Lernzyklus zu variieren. Im entwickelten Backpropagation Algorithmus dieser Arbeit werden die Lernmuster deshalb vor jedem Lernzyklus per Zufallsgenerator untereinander ausgetauscht.

Beim epochenweisen Lernen bzw. Batch-Training hat die Reihenfolge der Lernmuster keinen Einfluss auf das Lernverhalten des Netzwerks. Durch die Kumulation der einzelnen Gewichtsveränderungen aller Lernmuster ergibt sich die Anpassung der Gewichte in Richtung des steilsten Abstiegs der Fehlerfunktion aller Lernmuster. Insbesondere bei einer hohen Anzahl von Lernmustern ist eine epochenweise Anpassung der Gewichte zu bevorzugen, da die Anpassung nur einmal pro Lernzyklus ausgeführt wird und somit erheblich Rechenzeit eingespart wird.

Der Vorteil der höheren Geschwindigkeit des Batch-Trainings wird beim Online-Training allerdings durch die höhere Wahrscheinlichkeit, nicht in einem lokalen Minimum zu enden, und die schnellere Adaption des Netzes beim Nachtrainieren eines neuen Lernmusters sowie den niedrigeren Bedarf an Speicherplatz wieder ausgeglichen. Die Auswahl des Verfahrens ist nach Cichocki und Unbehauen [8] im höchsten Maße vom betrachteten Problem abhängig, wobei sich das Online-Training in den meisten Fällen als das bessere Lernverfahren erweist.

3.5.3 Transformieren von Ein- und Ausgabewerten

Auch der Wertebereich von Ein- und Ausgabewerten der Trainingsmuster hat einen großen Einfluss auf das Lernverhalten. Durch eine ungünstige Auswahl der Musterwerte kann es zu einer Konvergenz des Fehlers gegen ein lokales Minimum der Fehlerfunktion oder gar zum Abbruch des Lernverfahren Backpropagation kommen. Damit ein optimales Lernverhalten gewährleistet ist, müssen sowohl die Eingangs- als auch die Ausgangswerte in ein vorgegebenes Intervall transformiert werden. Hierbei bietet sich der Arbeitsbereich der Aktivierungsfunktion an, also der Bereich mit der größten Steigung, da das Netzwerk dort am stärksten auf Veränderungen reagieren kann.

Für die in Bild 3.3 dargestellte logistische Aktivierungsfunktion werden die Eingabewerte für das entwickelte Verfahren in das Intervall $[-1, 1]$ des Definitionsbereichs und die Ziel- bzw. Ausgabewerte in das Intervall $[0.2, 0.8]$ des Wertebereichs der Aktivierungsfunktion transformiert. Zur Berechnung der transformierten Werte müssen

hierzu in den Datensätzen stets die möglichen Grenzwerte der Ein- und Ausgabemuster angegeben werden.

3.5.4 Verzerrung des Ausgabefehlers

Durch eine Verzerrung des Fehlers, den das Backpropagation Netzwerk in den Neuronen der Ausgabeschicht macht, wird das Lernen nach FAHLMAN [11] weiter beschleunigt. Diese Verzerrung lässt sich mit Hilfe einer nichtlinearen Fehlerfunktion realisieren, so dass Ausgabeneuronen, die stark vom gewünschten Ziel abweichen, ein sehr großes Fehlersignal liefern und dadurch die entsprechenden Verbindungsgewichte schneller angepasst werden können. Hierzu wird die Fehlerdifferenz $(t_j - o_j)$ aus Gl. 3.19 durch den hyperbolischen Arcustangens $\operatorname{arctanh}(t_j - o_j)$ ersetzt. Wie in Bild 3.7 ersichtlich, ist die Funktion bei kleiner Differenz $(t_j - o_j)$ nahezu linear und die Gewichte werden in gewohnter Weise verändert. Erreicht die Differenz dagegen -1 bzw. $+1$, so geht der Fehler gegen $-\infty$ bzw. $+\infty$, was zu einer sehr starken Veränderung der Gewichte führt. Um dabei dem Problem von Überläufen bei der Simulation entgegen zu wirken, empfiehlt FAHLMAN [11], die Ausgabe der Funktion $\operatorname{arctanh}(t_j - o_j)$ auf das Intervall $[-17, +17]$ zu begrenzen. Aufgrund der verwendeten logistischen Aktivierungsfunktion und des Transformationsintervalls $[0.2, 0.8]$ für die Zielwerte werden diese Grenzwerte allerdings nicht erreicht.

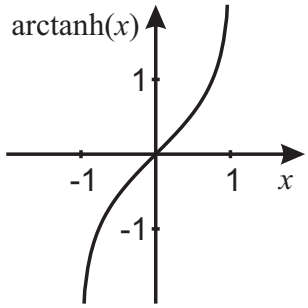


Bild 3.7: Arcustangens hyperbolicus

3.5.5 Momentum-Term

Eine häufig benutze Methode zur Vermeidung von Oszillation bzw. der Stagnation von Backpropagation auf flachen Plateaus, s. Kap. 3.4, beruht auf der Verwendung des sog. Momentum-Terms, der bei RUMELHART UND MCCLELLAND [40] beschrieben wird. Durch diesen Term fließt die bereits vollzogene Gewichtsänderung zum Zeitpunkt t in die Berechnung der neuen Gewichtsänderung zum Zeitpunkt $t + 1$ ein. Gl. 3.26 erweitert sich damit zu:

$$\Delta w_{ij}(t+1) = \eta\, o_i\, \delta_j + \alpha\, \Delta w_{ij}(t)\ . \tag{3.30}$$

Der konstante Faktor α wird in der Literatur als *Momentum* bezeichnet und gibt an, wie stark die neue Gewichtsänderung von der vorhergehenden Gewichtsänderung beeinflusst wird. Durch diesen zusätzlichen Term werden Schwankungen zweier aufeinander folgender Gewichtsveränderungen ausgeglichen bzw. gedämpft, während gleich gerichtete Gewichtsveränderungen verstärkt werden und somit das Minimum der Fehlerfunktion schneller erreicht wird.

Wie in der Literatur vorgeschlagen, hat sich bei den simulierten Backpropagation Netzwerken in dieser Arbeit und der eingestellten Lernrate von $\eta = 0.3$ ein Momentum von $\alpha = 0.9$ als sinnvoll erwiesen. Dieser Wert kann im entwickelten Entwurfssystem ebenfalls geändert werden.

3.5.6 Weight Decay

Zur Verbesserung der Generalisierungseigenschaften von Backpropagation Netzwerken wurde die entwickelte Backpropagation-Regel in eine Lernregel mit Gewichtsabnahme (*weight decay*) modifiziert. Diese Modifikation geht auf die Arbeiten von Werbos [64] zurück. Große Gewichte erschweren die Minimierung der Fehlerfunktion E, was sich häufig durch Oszillationen und unkontrollierte Sprünge auf der Fehlerfläche bemerkbar macht. Die Forderung nach kleinen Gewichten lässt sich als Term mit in die Fehlerfunktion nach Gl. 3.8 aufnehmen

$$E'_p = E_p + \frac{d}{2} \cdot \sum_{i,j} (w_{ij})^2 \ , \tag{3.31}$$

und es ergibt sich eine neue Lernregel zum Verändern der Gewichte:

$$\Delta w_{ij}(t+1) = \eta \, o_i \, \delta_j - d \cdot w_{ij}(t) \ . \tag{3.32}$$

Der zweite Term „bestraft“ zu große Gewichte, und das Verfahren ist gezwungen, die Gewichte gleichmäßiger im Netz zu verteilen. Dadurch ergibt sich zur Abbildung von Mustern auch eine weniger zerklüftete Abbildungsfunktion und somit eine höhere Generalisierungsleistung des neuronalen Netzes.

Die Verwendung des Weight Decay Parameters d lässt jedoch nur sehr kleine Werte zu, da sonst leicht alle Gewichte permanent geschwächt werden und das Verfahren nicht mehr konvergiert. Der beim Trainieren der Pumpen- und Turbinengeometrien angestrebte quadratische Netzfehler E_{min} konnte für Weight Decay Werte $d > 10 \cdot E_{min}$ bereits nicht mehr erreicht werden.

3.5.7 MultiBPG

Backpropagation Netzwerke werden in der Regel so lange trainiert, bis der Gesamtfehler E des Netzwerkes unter einen vorgegebenen Wert E_{min} sinkt. Werden so wie in

dieser Arbeit alle Geometriedaten einer Pumpen- oder Turbinenreihe in einem einzigen Netz gelernt, so ist es für bestimmte Geometrieparameter wünschenswert, eine genauere Approximation anzustreben als bei anderen. Aus diesem Grunde wurde für diese Arbeit eine Methode mit dem Namen *MultiBPG* (Multi-BackPropaGation) entwickelt, um ein einziges Backpropagation-Netzwerk mit n Eingabeneuronen und m Ausgabeneuronen in m gleich große Netze mit n Eingabeneuronen und 1 Ausgabeneuron umzuwandeln und umgekehrt. Es ist dann im Entwicklungssystem möglich, für jedes der Netze, also für jeden zu lernenden Geometrieparameter, einen eigenen Fehlerwert anzugeben. Die anderen Lernparameter bleiben dabei für alle Netze gleich. Wird für eines der Netze kein separates zu erreichendes Fehlerminimum angegeben, so verwendet der Lernalgorithmus für dieses Netz einen global einstellbaren Fehlerwert. Die einzelnen Netze werden beim Starten des Lernvorgangs im Entwurfssystem nacheinander trainiert.

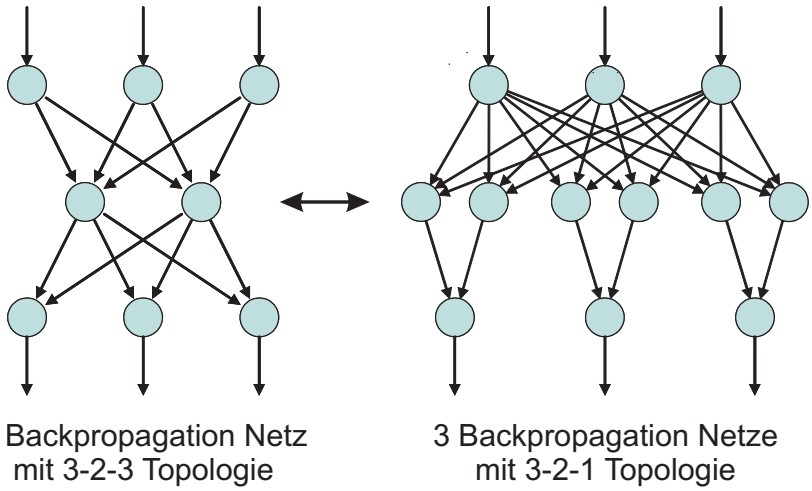


Bild 3.8: Aufsplittung eines Backpropagation Netzes mittels MultiBPG

In der Praxis zeigte sich jedoch, dass die Aufteilung in mehrere Teilnetze nur bei einer großen Anzahl zu lernender Geometrien sinnvoll ist, bei denen eine systematische Aufbereitung nur schwer möglich ist.

Kapitel 4

Entwurfssystem

In diesem Kapitel wird das grafische Entwurfssystem beschrieben, das die interaktive Aufbereitung von Pumpen- und Turbinengeometrien mit dem Simulationsbaustein für neuronale Netze verbindet.

4.1 Übersicht

Das für diese Arbeit entwickelte Entwurfssystem dient in erster Linie zum Generieren initialer Geometrien von Beschaufelungen hydraulischer Maschinen. Dabei werden die Geometrien allerdings nicht wie in anderen Erstentwurfsprogrammen aus geometrischen Zusammenhängen bzw. physikalischen Gesetzmäßigkeiten erzeugt. Ein großes Problem dieser Systeme ist der zeitliche Aufwand für die Generierung einer nahezu optimalen Initialgeometrie für einen vorgegebenen Betriebspunkt. Aus diesem Grund verfolgt das hier vorgestellte Programmsystem eine andere Strategie. Hierfür werden bereits vorhandene Strömungsmaschinengeometrien in Abhängigkeit von ihrem Betriebspunkt analysiert und zur Abbildung durch ein künstliches neuronales Netz aufbereitet. Unter Verwendung des in Kap. 3 vorgestellten Backpropagation Lernverfahren wird das Netz mit den aufbereiteten Betriebspunkt- und Geometriedaten trainiert. Nach dem Lernen ist das System in der Lage zu jedem beliebigen Betriebspunkt eine entsprechende Strömungsmaschinengeometrie durch einfaches Abrufen des neuronalen Netzes vorzuschlagen. Damit der gesamte Prozess, sowohl das Aufbereiten und Lernen als auch das Abrufen neuer Geometrien, schnell und anwenderfreundlich zu handhaben ist, wurde das System modular mit einer großen Anzahl zusätzlicher Funktionen ausgestattet:

- Automatische Datenaufbereitung von Pumpen bzw. Turbinengeometrien,
- mehrere interaktive Modifikationsbausteine zum Anpassen von Lernmustern sowie von Strömungsmaschinengeometrien unterschiedlicher Baureihen in der Meridianebene und in der konformen Abbildung,
- Tools zur Analyse der Geometrien durch die Darstellung des meridionalen Flächenverlaufs sowie des Schaufelwinkelverlaufs,

- Steuerung des implementierten Backpropagation Lernverfahrens zum Trainieren und Abrufen beliebiger Muster,
- grafische Darstellung der Netzwerkausgabe zur Beurteilung der gelernten Daten.

Eine generelle Struktur des entwickelten Entwurfssystems mit dem Namen *KNN-Designsystem* zeigt Bild 4.1.

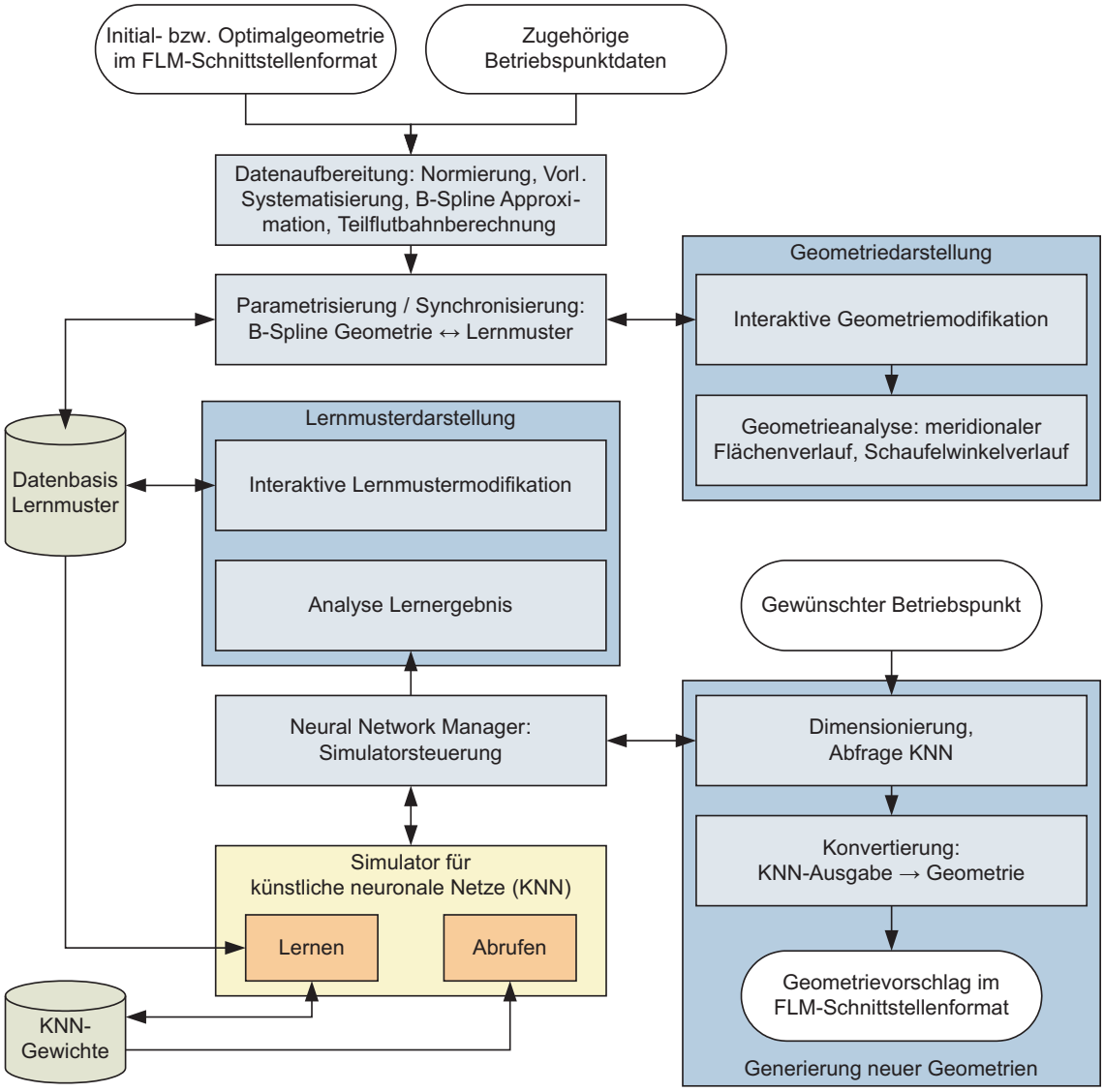


Bild 4.1: Systemstruktur des KNN-Designsystems

Ausgehend von einer initialen bzw. bereits optimierten Beschaufelung, die in dem FLM internen Schnittstellenformat vorliegt, und dem dafür vorgesehenen Betriebspunkt, erfolgt die Datenaufbereitung der Geometrie, auf die in Kap. 4.2 ausführlich eingegangen wird. Die verarbeiteten Daten werden anschließend dem nächsten Modul zur Parametrisierung übergeben, s. Kap. 4.3. Aus den daraus anfallenden Informationen wird ein Lernmuster gebildet, das in einer Lerndatenbank abgelegt

werden kann. Um die Strömungsmaschinengeometrie im System nachträglich interaktiv modifizieren zu können, ist das Parametrisierungsmodul ebenfalls für die Synchronisation zwischen Geometriedaten und Lernmustern verantwortlich.

Damit ein neuronales Netz sinnvolle Ergebnisse generieren kann, muss schon in dem zu lernenden Datensatz eine bestimmte Systematik vorhanden sein. Mit diesem Thema beschäftigt sich Kap. 4.4. Hierfür stehen dem Anwender die interaktiven Module zum Modifizieren der Lernmuster sowie der Geometrie zur Verfügung.

Sind ausreichend viele Trainingsdaten in der Datenbank vorhanden, so kann das Lernen beginnen, s. Kap. 4.5. Die Steuerung des ausgegliederten Neuronale-Netze-Simulators wird im Entwurfssystem vom sog. *Neural Network Manager* übernommen.

Ist das neuronale Netz trainiert, so lässt sich das System zum Generieren neuer Geometrien verwenden. Hierfür wird der gewünschte Betriebspunkt abgefragt und als Eingabemuster in das trainierte Netz eingespeist. Die Netzwerkausgabe muss dann lediglich in eine Strömungsmaschinengeometrie konvertiert werden, die im System betrachtet werden kann und gleichzeitig im FLM internen Schnittstellenformat ausgegeben wird. Dieser Vorgang wird in Kap. 4.6 genauer dargestellt.

Damit der Anwender sowohl die aufbereiteten bzw. neu erstellten Geometrien als auch das trainierte neuronale Netz bewerten kann, wurden in das System Werkzeuge zur Analyse integriert, die in Kap. 4.7 beschrieben sind.

Um das System so flexibel wie möglich zu gestalten, sind die geometriespezifischen Module nicht fest im System verankert, sondern werden über den sog. *Data Type Manager* als Interfaces zur Laufzeit instanziiert. Dadurch lassen sich im System verschiedene Geometriekonzepte realisieren. Die unterschiedliche Parametrisierung zwischen einer Kreiselpumpe und einer Francis Turbine wurde im System durch diese Möglichkeit genauso umgesetzt wie die Aufsplittung der Strömungsmaschinengeometrie in eine Meridiankontur und eine Schaufelgeometrie. Wird dem Data Type Manager kein Datentyp vorgegeben, so dient das Programm dem Anwender immerhin als grafische Benutzeroberfläche für den Neuronale-Netze-Simulator mit Editiermöglichkeit für die Lernmuster.

Damit das Programmsystem mit seiner grafischen Benutzeroberfläche sowohl unter Windows als auch unter Linux lauffähig ist, wurde für die Entwicklung die plattformunabhängige Programmiersprache Java verwendet. Zur Anzeige und interaktiven Modifikation der Daten auf dem Bildschirm kam das am Lehrstuhl von Kronschnabl [22] entwickelte grafische Toolkit zum Einsatz. Der Simulator für künstliche neuronale Netze ist unabhängig vom Entwurfssystem als eigenständiges Programm entwickelt worden, damit er auch in anderen Anwendung verwendet werden kann. Da die Simulation neuronaler Netze extrem rechenintensiv und somit sehr zeitaufwändig sein kann, wurde der Simulator im Hinblick auf die deutlich bessere Performance als Konsolenapplikation in der Programmiersprache C++ entwickelt und ist somit ebenfalls auf den unterschiedlichen Plattformen problemlos kompilier- und ausführbar.

An dieser Stelle sei darauf hingewiesen, dass für diese Arbeit lediglich Kreiselpumpen- bzw. Francis Turbinenlaufräder aufbereitet und gelernt wurden, um den Umfang dieser Arbeit zu begrenzen. Ebenso blieb die Dickenverteilung der Laufschaufeln beim Lernen unberücksichtigt, da diese bei allen Laufrädern einer Baureihe in gleicher Weise generiert wurde und somit lediglich zur Vergrößerung der Lerndatenbank und des neuronalen Netzes beigetragen hätte. Eine Erweiterung des Systems mit variabler Dickenverteilung sowie mehrerer Pumpen- bzw. Turbinenstufen ist jedoch problemlos möglich.

4.2 Datenaufbereitung

Wie bereits erwähnt, ist ein entscheidender Erfolgsfaktor beim Trainieren neuronaler Netze eine systematisch strukturierte Datenbasis. Hierfür müssen die zu lernenden Strömungsmaschinengeometrien zunächst aufbereitet und anschließend systematisiert werden. Die komplexe Datenaufbereitung wird dabei vom entwickelten Entwurfssystem durchgeführt und soll im Folgenden beschrieben werden.

4.2.1 Geometriebeschreibung

Ausgangslage für die Aufbereitung sind die Geometriedaten unterschiedlicher Beschaufelungen einer Strömungsmaschinenbaureihe im FLM internen Geometrieformat. Hierbei handelt es sich im Wesentlichen um zwei unterschiedliche Dateien, die die strömungsführende Geometrie vollständig beschreiben, s. Bild 4.2.

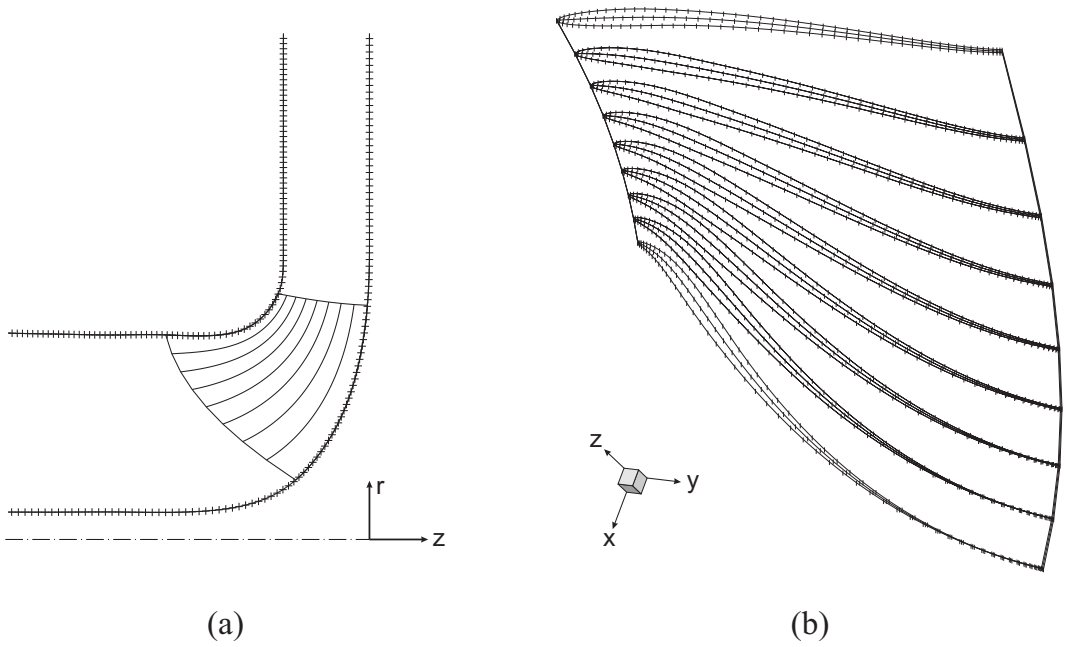


Bild 4.2: Strömungsführende Geometrie einer Francis Turbine im FLM internen Format: (a) Meridiankontur, (b) Laufschaufel

In einer Datei werden die Rotationsflächen der Nabe und der Deckscheibe in diskreten Punkten durch ihre jeweiligen Meridiankonturen repräsentiert, s. Bild 4.2a.

Die andere Datei enthält die Geometrie der dreidimensionalen Laufschaufel, ebenfalls in diskreten Punkten, s. Bild 4.2b. Dabei wird die Schaufeloberfläche durch die Schaufelskelettfläche und einem der Skelettfläche überlagerten Schaufelprofil dargestellt. Die Skelettfläche wird aus räumlichen Skelettlinien gebildet, die auf Rotationsflächen im Schaufelkanal definiert sind. Diese Rotationsflächen werden wiederum aus geometrisch definierten Stromlinien in der Meridianebene bestimmt, wie in Bild 4.2a verdeutlicht.

4.2.2 Normierung

Der erste Schritt der Aufbereitung ist die Normierung der unterschiedlich dimensionierten Geometrien auf eine vergleichbare Größe. Hierbei bietet sich als Referenz der jeweilige Laufraddurchmesser bzw. der zugehörige Radius r_D an, s. Bild 4.3.

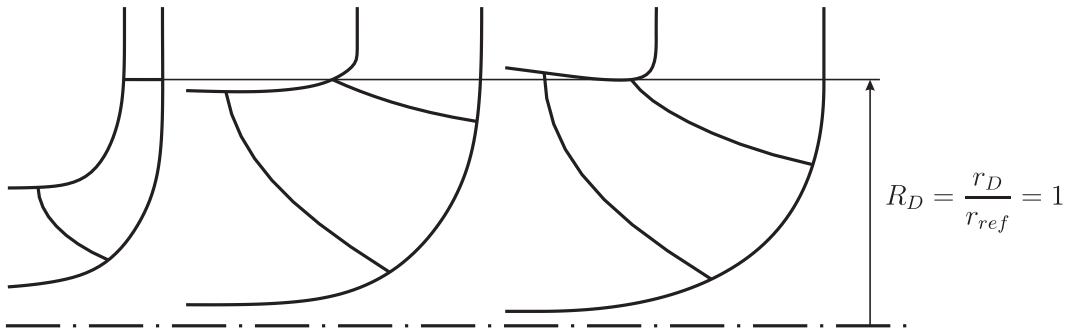


Bild 4.3: Normierung der Geometrie

Durch die Skalierung der Geometrie ergeben sich die normierten Koordinaten zu:

$$R = \frac{r}{r_{ref}}\,,\ \varphi = \varphi\,,\ Z = \frac{z}{r_{ref}} \quad \text{bzw.} \quad X = \frac{x}{r_{ref}}\,,\ Y = \frac{y}{r_{ref}}\,,\ Z = \frac{z}{r_{ref}}\,. \tag{4.1}$$

4.2.3 Geometrie- und Indexkonventionen

Da jeder Konstrukteur seine eigene Philosophie bei der Auslegung der Geometrie bezüglich der Lage im Koordinatensystem, der Drehrichtung oder der zugrunde liegenden Indexkonvention hat, wird die Geometrie im KNN-Designsystem geprüft und gegebenenfalls so angepasst, dass alle Geometrien stets in der gleichen systemeigenen Geometrie- bzw. Indexkonvention vorliegen, s. Bild 4.4.

Die Ausrichtung der Geometrie in der Meridianebene sieht eine axiale Erstreckung des Strömungskanals in negative z-Richtung vor, wobei die Nabenkontur bei $r = r_{ref}$ auf dem Wert $z = 0$ zu liegen kommt, Punkt A.

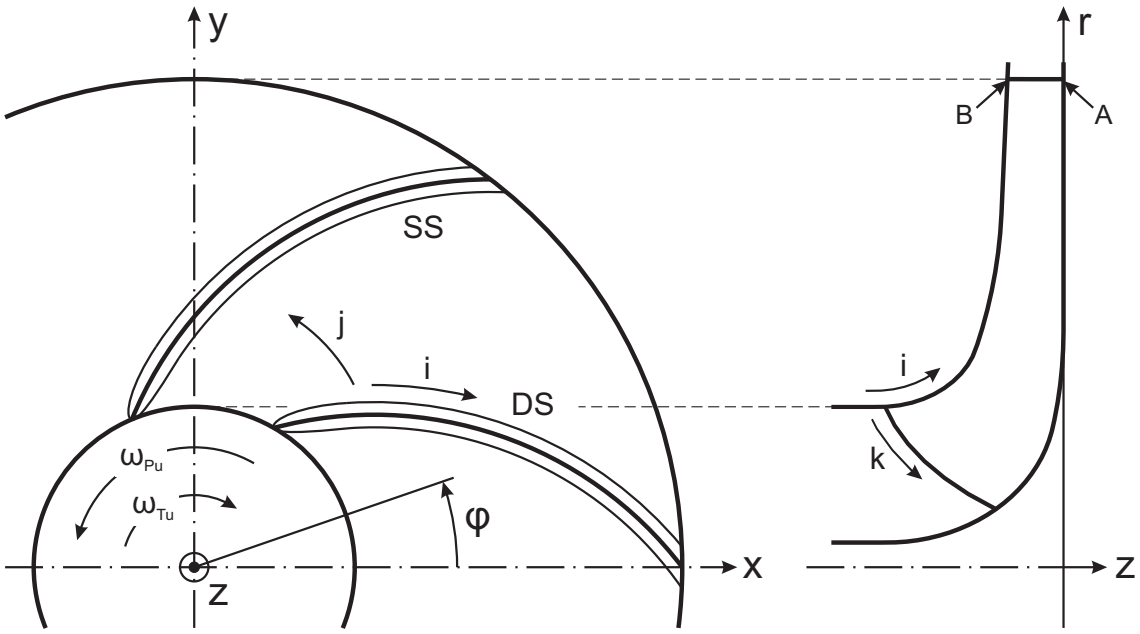


Bild 4.4: Geometrie- und Indexkonventionen im KNN-Designsystem

Die Lage der Schaufel orientiert sich zum einen an der Drehrichtung des Laufrades, die im KNN-Designsystem für eine Pumpe positiv und für eine Turbine negativ definiert ist. Dadurch liegt auch die Druck- bzw. Saugseite sowohl bei Pumpen als auch bei Turbinen stets auf der selben Seite der Schaufel, wie in Bild 4.4 dargestellt.

Zum anderen wird die Außenkante der Schaufel im Bereich der Deckscheibe, Punkt B, durch eine Drehung der Geometrie um die z-Achse bei $\varphi = 0$ fixiert.

Unabhängig ob es sich um eine Pumpe oder Turbine handelt, wird die Geometrie immer in gleicher Weise indiziert. Dabei zeigt die Indexrichtung i stets in Pumpenfließrichtung, j in positive φ-Richtung bzw. von der Druckseite der Schaufel zur Saugseite der nächsten Schaufel und k von der Deckscheibe zur Nabe. Alle betrachteten Koordinatensysteme (x, y, z), (r, φ, z) und (i, j, k) bilden somit jeweils ein Rechtssystem.

4.2.4 Systematisierung der Schaufelrotationsflächen

In einem letzten Schritt müssen noch die Rotationsflächen auf denen die einzelnen Skelettlinien der Schaufel liegen systematisiert werden, da diese je nach spezifischer Drehzahl sowohl in ihrer Anzahl als auch in ihrer Konstruktionsmethode unterschiedlich sein können, s. Bild 4.5. Durch die Verschneidung der dreidimensionalen Schaufel mit den systematisch generierten Rotationsflächen entstehen Schaufelprofile, die vergleichbar sind und somit im späteren Verlauf parametrisiert werden können. Hierbei kommen die in Kap. 2.2.1 erwähnten kubischen Splines zur Glättung der Schaufelskelettfläche und der Profilierung zum Einsatz, damit bei der Verschneidung, bedingt durch die diskrete Darstellung der Geometrie, keine „kantigen“ Profile entstehen. Die äquivalenten Punkte der in k-Richtung, also von der Deckscheibe zur

Nabe, übereinander liegenden Skelettlinien werden mit Hilfe des Splines als glatte Kurve beschrieben. Ebenso werden die Konturen der neuen Rotationsflächen jeweils mit einem kubischen Spline aufbereitet. Die Verschneidung der beiden Splinekurven erfolgt in der zweidimensionalen Meridianebene mit Hilfe eines iterativen Verfahrens. Aus den berechneten Schnittpunkten der Kurven lässt sich anschließend die systematisch aufbereitete, dreidimensionale Schaufelgeometrie erstellen.

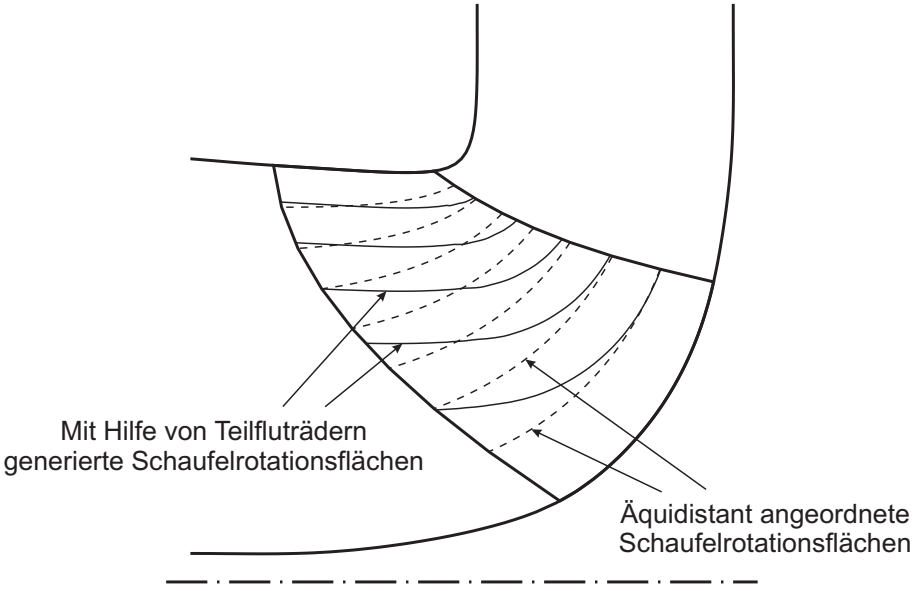


Bild 4.5: Unterschiedliche Schaufelrotationsflächen

Die Generierung der Schaufelrotationsflächen erfolgt mit Hilfe von Teilfluträdern, da sie den tatsächlichen Verlauf des Fluids durch den Strömungskanal besser beschreiben als z.B. die in Bild 4.5 dargestellten, äquidistant über den Querschnitt angeordneten Rotationsflächen und somit auch der Schaufelwinkelverlauf entlang eines Stromfadens plausible Werte annimmt. Hierbei wird angenommen, dass durch jedes Teilflutrad, begrenzt durch jeweils zwei benachbarte Rotationsflächen, der gleiche Teilvolumenstrom fließt. Das zur Generierung der Schaufelrotationsflächen entwickelte Verfahren wird im nächsten Abschnitt vorgestellt.

4.2.5 Generierung von Teilfluträdern

Die Systematisierung der Rotationsflächen geschieht durch eine vollständige Neuberechnung der Flächenkonturen in der Meridianebene. Dabei werden die Konturen durch Stromlinien beschrieben, die unabhängig von der Schaufelgeometrie nur aus der Deckscheiben- und Nabenkontur hervorgehen, s. Bild 4.6. Die Konstruktion einer vorgegebenen Anzahl geometrischer Stromlinien Z_{SL} erfolgt durch die Aufteilung des Strömungskanals in $(Z_{SL}-1)$ sog. Teilfluträder, durch die jeweils der gleiche Teilvolumenstrom $Q_k = Q/(Z_{SL}-1) = \text{const.}$ fließt. Unter der Voraussetzung einer quer zur Strömungsrichtung konstanten Meridianströmungsgeschwindigkeit c_m folgt für die Querschnittsflächen der Teilfluträder entlang einer zu den Stromlinien senkrecht

stehenden Potentiallinie[1], $m = \text{const.}$:

$$A_k = \frac{Q_k}{c_m} = 2\pi\, r_k \cdot \delta_k = \text{const.}\ , \tag{4.2}$$

wobei r_k der mittlere Radius der Flächenkontur und δ_k die Schichtdicke des Teilflutrades ist. Für die in Bild 4.6 dargestellten Kreise bedeutet dies, dass die Kreisdurchmesser δ_k umgekehrt proportional zum jeweiligen Mittelpunktsradius r_k des Kreises sein müssen:

$$\delta_k \sim \frac{1}{r_k}\ . \tag{4.3}$$

Die geometrische Konstruktion der Stromlinien stellt mit den bisher gemachten Überlegungen ein nur schwer zu lösendes Problem dar, da alle geometrischen Komponenten stark voneinander abhängen. So beeinflussen sich die Mittelpunktsradien r_k, die Schichtdicken δ_k, die Lage der Stromlinien und die senkrecht zu den Stromlinien stehenden Potentiallinien wechselseitig. Dieses Problem lässt sich nur mit einem sehr zeitaufwändigen impliziten Potentialverfahren sinnvoll lösen.

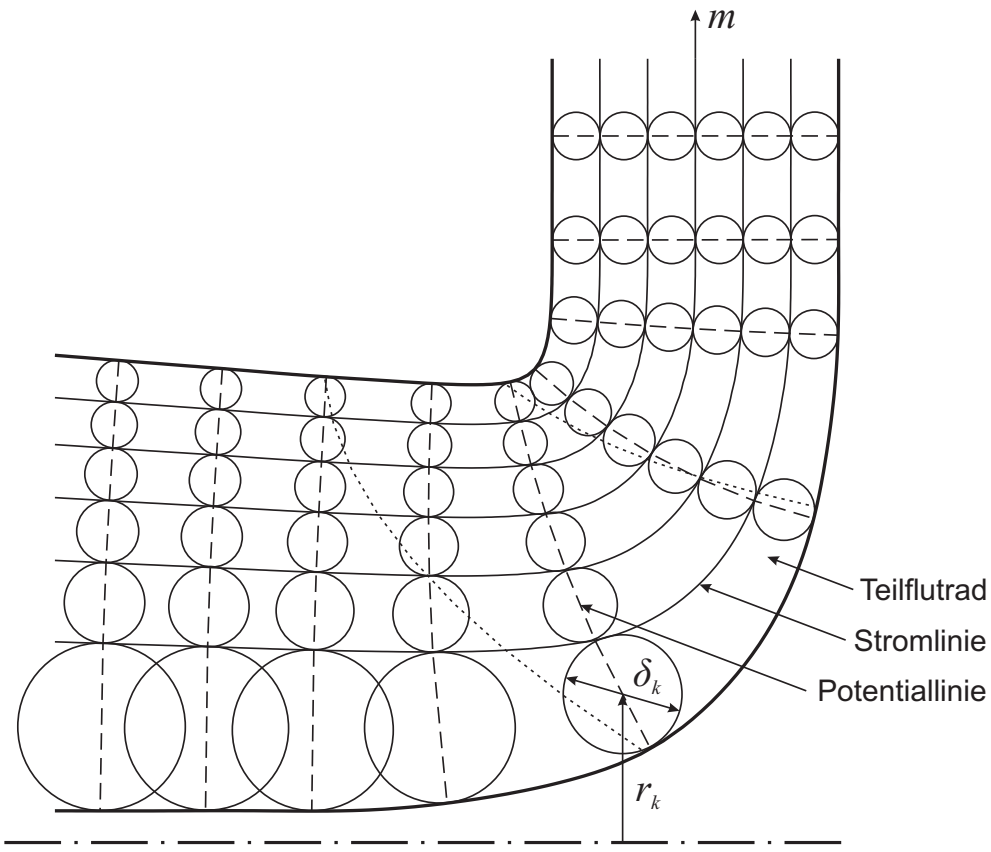


Bild 4.6: Konstruktion von Stromlinien durch Teilfluträder

[1]Dabei ist in beiden Fällen nicht die aus der Potentialtheorie bekannte Definition, sondern die geometrische Definition in Anlehnung an die Potentialtheorie gemeint

Aus diesem Grunde wurde für diese Arbeit ein algebraisches Verfahren entwickelt, mit dem eine schnelle und ausreichend genaue Bestimmung der Stromlinien möglich ist. Die Generierung der Stromlinien erfolgt dabei in zwei Schritten.

Im ersten Schritt werden Hilfskurven konstruiert, die als Ersatz für die eigentlichen Potentiallinien in Bild 4.6 dienen. Die Vorgehensweise ist schematisch in Bild 4.7a dargestellt. Die Konstruktion beginnt mit dem Auskreisen der Meridiankontur. Hierbei wird das gleiche Verfahren verwendet, das auch zum Berechnen des meridionalen Flächenverlaufs bei der Geometrieanalyse eingesetzt wird und das in Kap. 4.7.1 ausführlich beschrieben wird. Durch das Verfahren erhält man die geometrische Mitte zwischen Naben- und Deckscheibenkontur in Form einer repräsentativen Mittelpunktslinie. Zu jedem Mittelpunkt D_2 auf der Linie liefert das Verfahren zusätzlich die beiden Berührungspunkte D_1 und D_3 des jeweiligen Auskugelungskreises mit der Deckscheiben- bzw. Nabenkontur. Die Punkte D_1, D_2 und D_3 dienen anschließend als Kontrollpunkte eines uniformen B-Splines 3. Ordnung mit dem die gesuchten Hilfskurven in einfacher Weise beschrieben werden können. Da die Kurvenenden der B-Splines durch diese Methode stets senkrecht zur Deckscheiben- bzw. Nabenkontur stehen, ist somit eine akzeptable Näherung der zu erwartenden Potentiallinien gegeben.

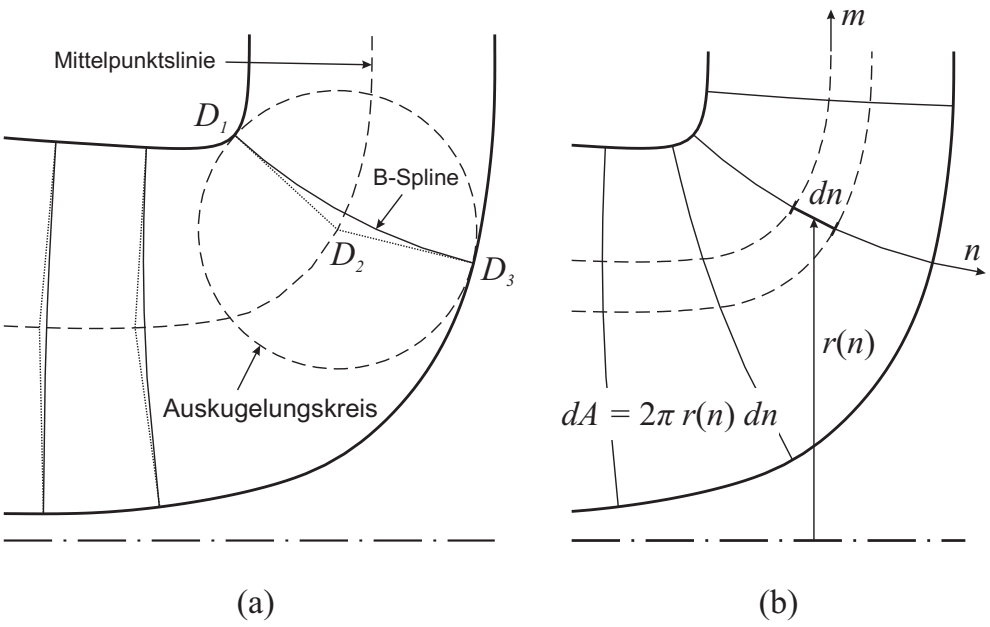


Bild 4.7: Konstruktion von Teilfluträder: (a) Hilfskurven als Ersatz für Potentiallinien, (b) Stromlinienberechnung

Im zweiten Schritt werden die Stromlinien unter Verwendung der zuvor generierten Potentiallinien bestimmt. Wie Bild 4.7b zeigt, gilt für die Querschnittsfläche dA eines Teilflutrades mit der infinitesimalen Schichtdicke dn:

$$dA = 2\pi\, r(n)\, dn \ , \tag{4.4}$$

und die Berechnung der Querschnittsfläche A_k gemäß Gl. 4.2 lässt sich durch Integration von Gl. 4.4 wie folgt ersetzen:

$$A_k = 2\pi \int_{n_{k-1}}^{n_k} r(n)\, dn \ . \tag{4.5}$$

Unter der Voraussetzung gleich großer Querschnittsflächen A_k, können die Schnittpunktskoordinaten n_k der Stromlinien entlang einer Potentiallinie durch folgende Beziehung bestimmt werden:

$$\int_0^{n_k} r(n)\, dn = \frac{k}{Z_{SL}-1} \cdot \int_0^{n_{max}} r(n)\, dn \ . \tag{4.6}$$

Das Flächenintegral wird im Verfahren unter Verwendung kubischer Splines berechnet. Damit es zu keiner Verwechslung mit dem Index k für Stromlinien kommt, wird für die folgenden Betrachtungen der von k unabhängige Index k' eingeführt. Ausgehend von einer diskretisierten Potentiallinie wird zunächst für jeden Punkt $(r_{k'}, z_{k'})$ der Linie die zugehörige Kurvenlänge $n_{k'}$ berechnet mit

$$n_{k'} = n_{k'-1} + \sqrt{(r_{k'} - r_{k'-1})^2 + (z_{k'} - z_{k'-1})^2} \ , \tag{4.7}$$

wobei $n_0 = 0$ ist. Mit Hilfe eines kubischen Splines lässt sich die r-Koordinate der Potentiallinie in Abhängigkeit von n als Kurve darstellen, wobei die einzelnen Splinepolynome zwischen den Stützstellen folgende Form haben:

$$r_{k'}(n) = a_{k'}(n - n_{k'})^3 + b_{k'}(n - n_{k'})^2 + c_{k'}(n - n_{k'}) + d_{k'} \ . \tag{4.8}$$

Die Koeffizienten $a_{k'}$, $b_{k'}$, $c_{k'}$ und $d_{k'}$ der jeweiligen Teilkurven werden mit dem in Kap. 2.2.1 dargestellten Verfahren ermittelt. Durch Integration von Gl. 4.8 erhält man für jedes Teilstück mit $n_{k'} \le n \le n_{k'+1}$:

$$\begin{aligned} \int_{n_{k'}}^{n} r_{k'}(n)\, dn \ &= \ \frac{1}{4} a_{k'}(n - n_{k'})^4 + \frac{1}{3} b_{k'}(n - n_{k'})^3 \\ &\quad + \frac{1}{2} c_{k'}(n - n_{k'})^2 + d_{k'}(n - n_{k'}) \ . \end{aligned} \tag{4.9}$$

Die Summe der einzelnen Integrale über die Teilintervalle $[n_{k'}, n_{k'+1}]$ für $n \ge n_{k'+1}$ bzw. $[n_{k'}, n]$ für $n < n_{k'+1}$ ergibt dann das bestimmte Integral von $r(n)$ im Gesamtintervall $[0, n]$.

Hiermit lässt sich die rechte Seite von Gl. 4.6 für alle Strombahnen k berechnen. Die zugehörige Lage n_k der jeweiligen Stromlinie wird anschließend iterativ über die linke Seite der Gleichung bestimmt.

Die äquivalenten Punktkoordinaten r_k und z_k erhält man durch Einsetzen von n_k in die entsprechenden Splinefunktionen für $r_{k'}(n)$, s. Gl. 4.8, bzw. $z_{k'}(n)$.

Durch Splineinterpolation der berechneten Punkte einer Stromlinie und anschließender Verschneidung mit der Schaufelgeometrie erhält man die neue Schaufelskelettlinie sowie die zugehörige Dickenüberlagerung des Schaufelprofils.

4.3 Parametrisierung der Geometrie

Nachdem die Strömungsmaschinengeometrie aufbereitet und einer vorläufigen Systematisierung unterzogen wurde, muss sie parametrisiert werden, um die zu lernende Datenmenge auf ein Minimum zu reduzieren. Hierbei spielt die Art und Weise der Parametrisierung eine wesentliche Rolle, da sie sowohl Einfluss auf die Qualität der approximierten Geometrie als auch auf das Lernergebnis des neuronalen Netzes hat. Entsprechend der Entwurfsmethodik des Lehrstuhls werden in diesem Programmsystem Meridiankonturen und Schaufelgeometrie getrennt voneinander aufbereitet bzw. parametrisiert und somit auch in unterschiedlichen neuronalen Netzen trainiert. In den folgenden Abschnitten werden die unterschiedlichen Parametrisierungen vorgestellt und diskutiert, die im System getestet wurden.

4.3.1 Meridiangeometrie

Der klassische Erstentwurf einer Strömungsmaschine beginnt mit der Auslegung der strömungsführenden Geometrien in der Meridianebene. Dies betrifft im wesentlichen die Deckscheiben- und Nabenkontur sowie die Lage der Ein- und Autrittskanten der Schaufeln. Bild 4.8 zeigt die Meridiangeometrie einer Francis Turbine und einer Kreiselpumpe.

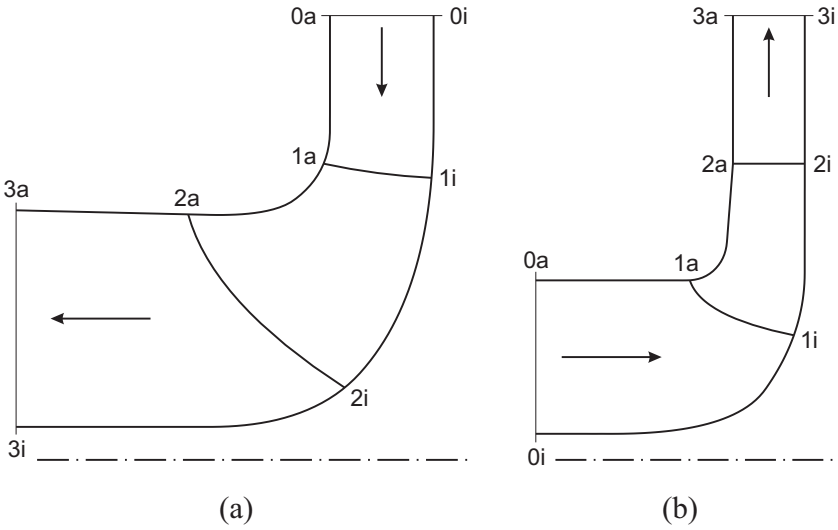


Bild 4.8: Meridiangeometrie: (a) Francis Turbine, (b) Kreiselpumpe

In Strömungsrichtung aufsteigend stellen die Indizes 0 und 3 einen beliebigen Ort vor bzw. nach dem Laufrad und 1 und 2 jeweils die Ein- bzw. Austrittskante der Schaufel dar. Die Bezeichnungen a und i beziehen sich auf die Lage eines Punktes auf der Außen- (Deckscheibe) bzw. Innenkontur (Nabe).

Die Parametrisierung erfolgt mit Hilfe von B-Splines, wobei die Ein- bzw. Austrittskanten der Schaufel jeweils durch einen uniformen B-Spline approximiert werden, während die Deckscheiben- und Nabenkontur aus mehreren uniformen B-Splines

zusammengesetzt werden. Damit erreicht man, dass bestimmte B-Spline Kontrollpunkte an vorgegebenen Stellen auf der Kontur zu liegen kommen. Erschwerend kommt allerdings hinzu, dass der Übergang von einem B-Spline zum nächsten einmal stetig differenzierbar sein muss. Wie Bild 4.9 zeigt, lässt sich dieses Problem durch die Anordnung der betroffenen Kontrollpunkte D_{12}, D_{13}, D_{21} und D_{22} auf einer Geraden relativ leicht lösen, wobei die Endpunkte der B-Splines D_{13} und D_{21} zusammen fallen.

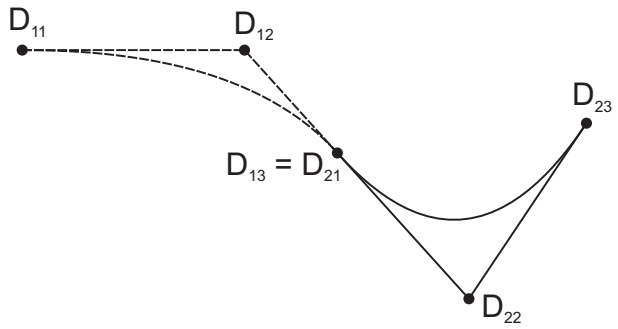


Bild 4.9: B-Spline Übergänge

Damit die unterschiedlichen Parametrisierungen bei der grafischen Darstellung einheitlich behandelt werden können, müssen die einzelnen uniformen B-Spline Segmente einer Kontur zu einem einzigen B-Spline zusammengefasst werden. Dies lässt sich mit Hilfe nicht-uniformer B-Splines realisieren. So ergibt sich z.B. aus den in Bild 4.9 dargestellten uniformen B-Splines 3. Ordnung mit den Trägervektoren $\vec{T} = (0, 0, 0, 1, 1, 1)$ eine äquivalente Darstellung der Kurve unter Verwendung der gleichen Kontrollpunkte und einem nicht-uniformen B-Spline 3. Ordnung mit dem Trägervektor $\vec{T} = (0, 0, 0, \frac{1}{2}, \frac{1}{2}, 1, 1, 1)$. Zur Verkettung n solcher B-Spline Segmente ist demnach ein Trägervektor $\vec{T} = (0, 0, 0, \frac{1}{n}, \frac{1}{n}, \frac{2}{n}, \frac{2}{n}, \ldots, \frac{n-1}{n}, \frac{n-1}{n}, 1, 1, 1)$ erforderlich.

Abhängig von den Kontrollpunkten der vier B-Splines, die die komplette Meridiangeometrie, d.h. Deckscheibe, Nabe, Ein- und Austrittskante, beschreiben, lassen sich geometrische Parameter ableiten, die dem neuronalen Netz als Lernmuster dienen.

Parametrisierung der Meridiangeometrie von Francis Turbinen

In einer ersten Version des KNN-Designsystems wurde versucht, eine für den Anwender praktische Parametrisierung zu verwirklichen, s. Bild 4.10. Hierbei wurde darauf Wert gelegt, dass jeweils ein Kontrollpunkt der approximierenden B-Splines der Deckscheiben- bzw. Nabenkontur auf den vier charakteristischen Punkten D_{1a}, D_{1i}, D_{2a} und D_{2i} zu liegen kommt. Die Kontrollpunkte D_{Aa}, D_{Ai}, D_{Ba} und D_{Bi} werden geometrisch bestimmt und stellen jeweils die Endpunkte gerader Streckenabschnitte dar. Die Ein- und Austrittskanten werden, wie bereits erwähnt, jeweils mit einem uniformen B-Spline 3. Ordnung approximiert.

Aus den so gewonnenen B-Splines lässt sich ein Lernparametersatz generieren, dessen Zielwerte sich aus den in Bild 4.10 dargestellten geometrischen Werten zusammensetzen. Die Kontrollpunkte D_{1i}, D_{2a} und D_{2i} werden im Parametersatz jeweils

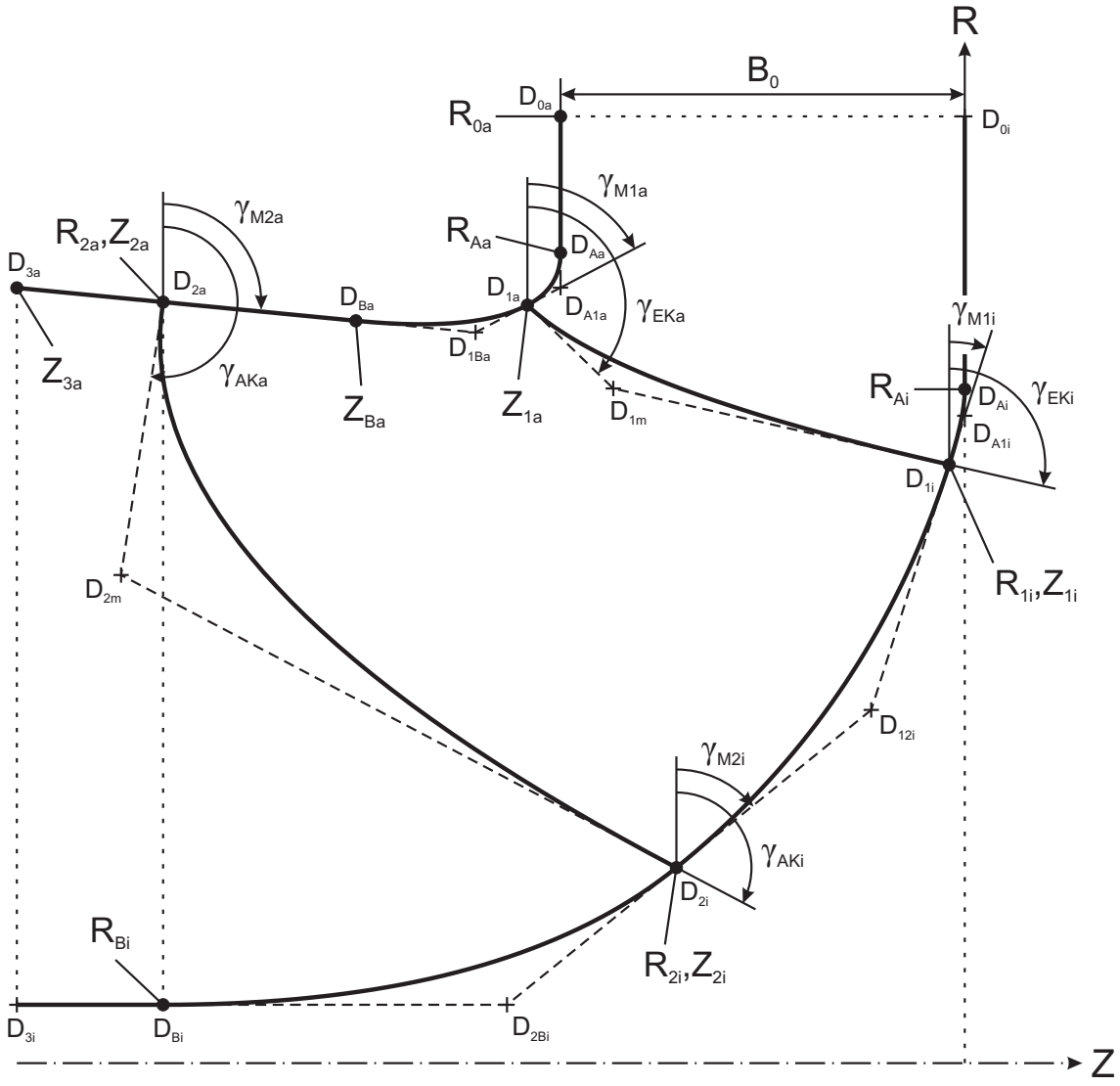


Bild 4.10: Ursprüngliche Parametrisierung der Meridiankontur einer Turbine

durch ihre R- und Z-Werte beschrieben. Für Punkt D_{1a} ist lediglich der Z-Wert erforderlich, da R_{1a} stets den Wert 1 hat. Ebenso werden für die Punkte D_{Ba} und D_{3a} nur die Z-Werte übernommen, da die zugehörigen R-Werte durch die Koordinaten von Punkt D_{2a} und den Winkel γ_{M2a} definiert sind. Wegen $Z_{0i} = Z_{Ai} = 0$, $Z_{0a} = Z_{Aa} = B_0$ und der Annahme, dass $Z_{Bi} = Z_{2a}$ ist, werden für die Punkte D_{0a}, D_{Aa}, D_{Ai} und D_{Bi} ebenfalls nur die R-Werte benötigt. Aus den Kontrollpolygonen der B-Splines ergeben sich zusätzlich die Winkel γ_{M1a}, γ_{M1i}, γ_{M2a} und γ_{M2i}, die die Steigung der Deckscheiben- bzw. Nabenkontur in den jeweiligen Punkten definieren, sowie die Winkel γ_{EKa}, γ_{EKi}, γ_{AKa} und γ_{AKi} zur Beschreibung der Ein- und Austrittskantenneigung.

Leider führte das Lernen dieser Parameter nicht zum gewünschten Ergebnis. Die Tatsache, dass die Simulation neuronaler Netze lediglich eine Approximation der Lernmuster darstellt und sich somit kleine Abweichungen der aus dem Netz abgerufenen Geometrieparameter zum Lernparametersatz ergeben, führen vor allem kleine Schwankungen in den γ Werten zu unbefriedigend großen Geometrieabweichungen

bis hin zu völlig verunstalteten Deckscheiben- bzw. Nabenkonturen im Bereich kleiner spezifischer Drehzahlen n_q. Zusätzlich verhindern vor allem die großen Transformationsintervalle der γ Werte, die für das Lernverfahren benötigt werden, eine ausreichend genaue Approximation dieser Werte. Besonders stark macht sich dies am Wert γ_{M1a} bemerkbar. Während dieser Wert bei Geometrien mit $n_q < 30$ min^{-1} gegen 0° geht, liegt er bei $n_q > 100$ min^{-1} bereits bei fast 90°. Eine vom Lernalgorithmus geforderte Transformation dieses Bereichs in das Intervall $[0.2, 0.8]$ ergibt einen deutlich höheren Approximationfehler als beispielsweise bei den R-Werten, die in einem Bereich zwischen 0 und 1.5 liegen.

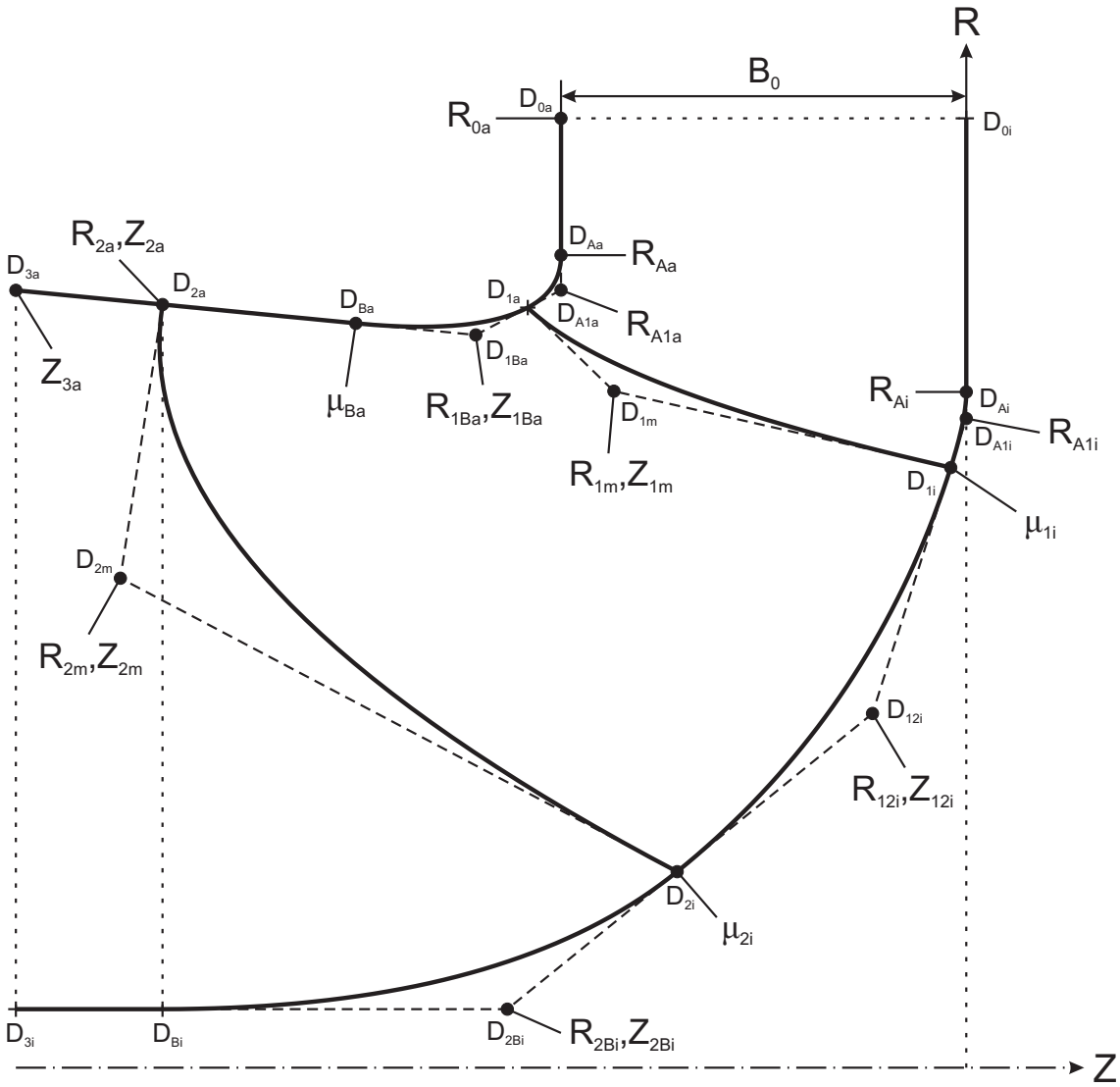


Bild 4.11: Verbesserte Parametrisierung der Meridiankontur einer Turbine

Eine Verbesserung der Ergebnisse stellt die Parametrisierung gemäß Bild 4.11 dar. Hierbei wurde aufgrund der beschriebenen Probleme vollständig auf Winkelparameter verzichtet. Da sich bei einigen Kontrollpolygonabschnitten der verwendeten nicht-uniformen B-Splines zur Approximation der Deckscheiben- bzw. Nabenkontur drei oder mehr Kontrollpunkte auf einer Geraden befinden, bietet sich eine Beschreibung der Punkte über ein Streckenverhältnis μ an, das sich aus dem betrachteten

Kontrollpunkt und den Kontrollpunkten an den Enden der jeweiligen Gerade berechnet und sich somit stets im Intervall $[0, 1]$ befindet. Für die Parametrisierung werden dann lediglich R- und Z-Werte sowie die Streckenverhältnisse μ verwendet, deren Intervallbereiche ähnlich groß dimensioniert sind. Die in Bild 4.11 dargestellten Streckenverhältnisse μ_{Ba}, μ_{1i} und μ_{2i} sind wie folgt definiert:

$$\mu_{Ba} = \frac{|\vec{D}_{Ba} - \vec{D}_{1Ba}|}{|\vec{D}_{2a} - \vec{D}_{1Ba}|} \quad , \quad \mu_{1i} = \frac{|\vec{D}_{1i} - \vec{D}_{A1i}|}{|\vec{D}_{12i} - \vec{D}_{A1i}|} \quad , \quad \mu_{2i} = \frac{|\vec{D}_{2i} - \vec{D}_{12i}|}{|\vec{D}_{2Bi} - \vec{D}_{12i}|} \; . \tag{4.10}$$

Durch diese Parametrisierung wurde die Approximation durch das neuronale Netz insgesamt verbessert, so dass sich bei gleichem Netzfehler im Gegensatz zur ursprünglichen Parametrisierung nahezu identische Meridiankonturen zu den in den Lernmustern gespeicherten Konturen ergaben, s. Bild 4.12.

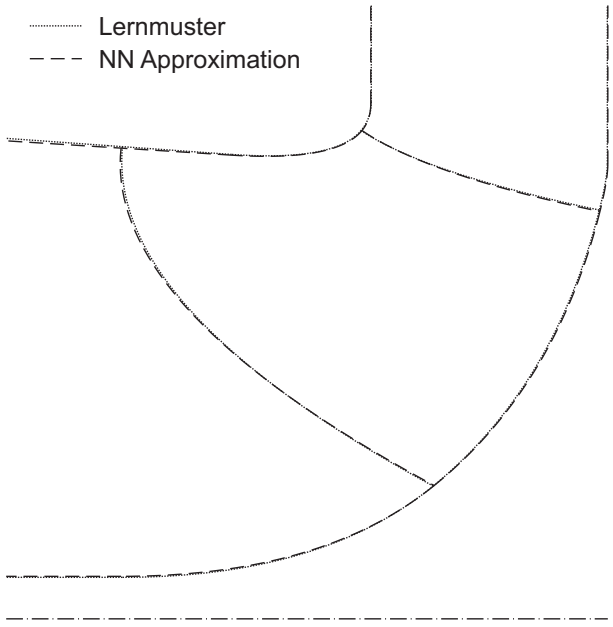


Bild 4.12: Vergleich der aus dem Lernmuster und dem neuronalen Netz generierten Meridiankonturen einer Turbine vom Typ FT60

Ein Problem dieser Parametrisierung ergibt sich allerdings für den Anwender durch die unhandlich gewordenen Geometrieparameter. Während die R- und Z-Werte der auf der Deckscheiben- bzw. Nabenkontur liegenden Kontrollpunkte sowie die Steigung der Kurven an den charakteristischen Punkten D_{1a}, D_{1i}, D_{2a} und D_{2i} bei der ursprünglichen Parametrisierung direkt beeinflusst werden können, lässt sich der Kurvenverlauf mit den neuen Geometrieparametern nur indirekt verändern. Für den Anwender ist der tatsächliche Kurvenverlauf der Deckscheiben- bzw. Nabenkontur durch den neuen Parametersatz daher nur schwer nachvollziehbar.

Ein weiterer Nachteil bei den beiden bisher beschriebenen Parametrisierungen ist die durch die Schaufelkanten fest vorgeschriebene Lage der Kontrollpunkte D_{1a}, D_{1i},

D_{2a} und D_{2i} der beschreibenden B-Splines der Deckscheiben- bzw. Nabenkontur. Für kleiner werdende spezifische Drehzahlen n_q rücken daher vor allem die Punkte D_{Aa}, D_{A1a} und D_{1a} bzw. D_{Ai}, D_{A1i} und D_{1i} immer näher zusammen, bis sie schließlich vollständig übereinander liegen. Die anschließende Krümmung der Deckscheibenkontur wird dann lediglich durch ein einziges B-Spline Segment dargestellt, was unter Umständen für eine Approximation der Originalkontur zu ungenau sein kann. Aus diesem Grunde wurde im entwickelten Programmsystem eine von den Ein- und Austrittskanten unabhängige Parametrisierung der Deckscheiben- und Nabenkontur gewählt, s. Bild 4.13.

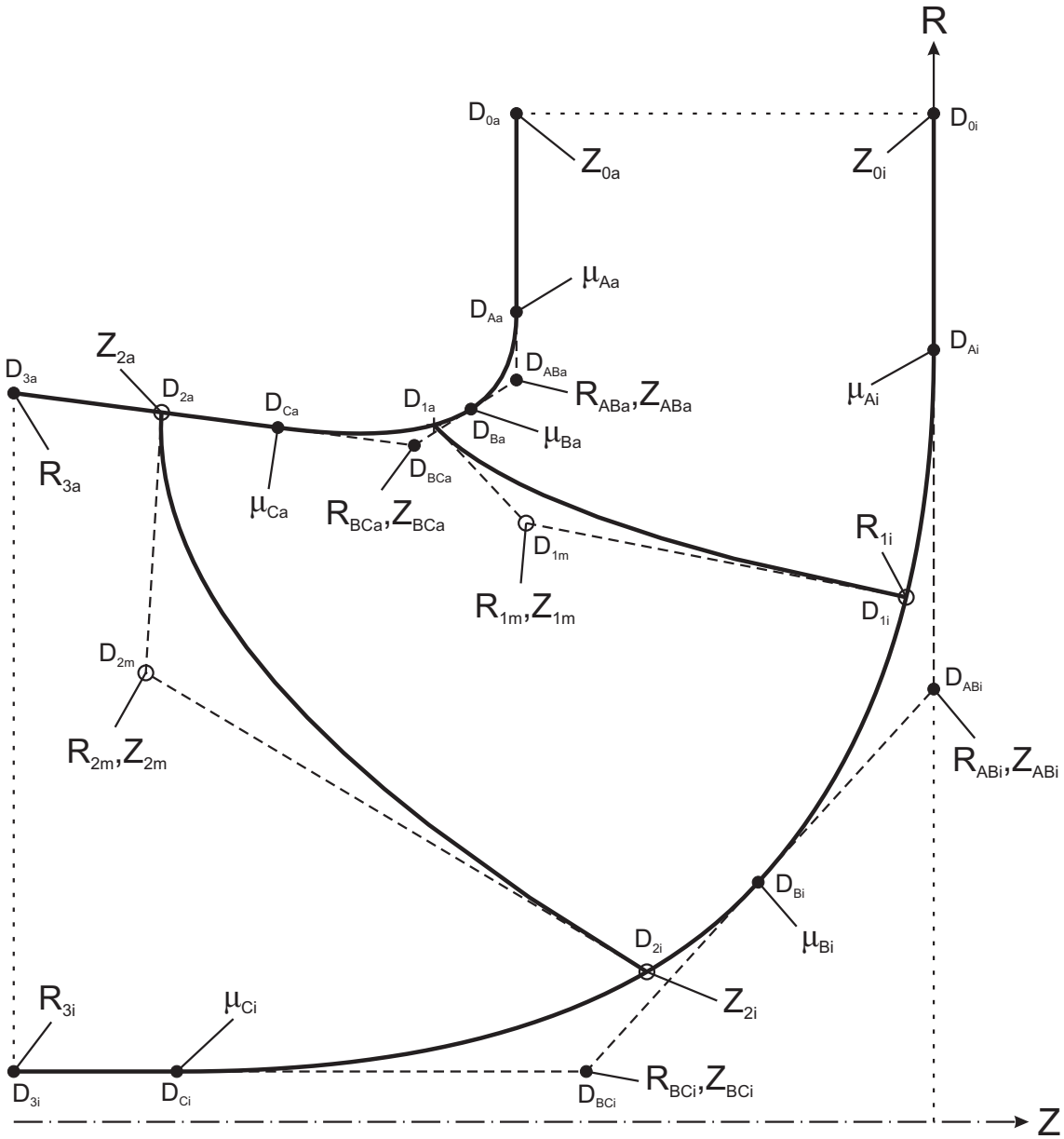


Bild 4.13: Endgültige Parametrisierung der Meridiangeometrie einer Francis Turbine im KNN-Designsystem

Die charakteristischen Punkte D_{1a}, D_{1i}, D_{2a} und D_{2i} stehen dabei nur noch als Kontrollpunkte der Ein- und Austrittskanten B-Splines zur Verfügung, wodurch sich

Deckscheiben- und Nabenkontur wesentlich besser approximieren lassen.

Die Parametrisierung wurde zudem erweitert, so dass auch diagonal durchströmte Turbinen parametrisiert werden können. Für die in dieser Arbeit trainierten Francis Turbinen bleiben jedoch einige der trainierten Geometrieparameter unberücksichtigt und werden vom eingesetzten Parametrisierungmodul aufgrund der radialen Bauweise wie folgt festgelegt: $R_{BCi} = R_{3i}$, $Z_{ABa} = Z_{0a}$, $Z_{ABi} = Z_{0i} = 0$.

Die Werte R_{0a} und R_{0i} bzw. Z_{3a} und Z_{3i} werden bei dieser Parametrisierung nicht mehr wie in den Vorgängerversionen als Zielwerte in das Lernmuster übernommen, sondern werden vom Parametrisierungsmodul mit $R_{0a} = R_{0i} = 2.5$ und $Z_{3a} = Z_{3i} = -2$ vorgegeben, wodurch sich ein genügend langer Eintritts- bzw. Austrittsbereich der Meridiankontur ergibt (im Bild verkürzt dargestellt).

Die Streckenverhältnisse μ sind folgendermaßen definiert:

$$\begin{aligned} \mu_{Aa} &= \frac{|\vec{D}_{Aa} - \vec{D}_{0a}|}{|\vec{D}_{ABa} - \vec{D}_{0a}|} \quad , \quad \mu_{Ba} = \frac{|\vec{D}_{Ba} - \vec{D}_{BCa}|}{|\vec{D}_{ABa} - \vec{D}_{BCa}|} \quad , \quad \mu_{Ca} = \frac{|\vec{D}_{Ca} - \vec{D}_{3a}|}{|\vec{D}_{BCa} - \vec{D}_{3a}|} \ , \\ \mu_{Ai} &= \frac{|\vec{D}_{Ai} - \vec{D}_{0i}|}{|\vec{D}_{ABi} - \vec{D}_{0i}|} \quad , \quad \mu_{Bi} = \frac{|\vec{D}_{Bi} - \vec{D}_{BCi}|}{|\vec{D}_{ABi} - \vec{D}_{BCi}|} \quad , \quad \mu_{Ci} = \frac{|\vec{D}_{Ci} - \vec{D}_{3i}|}{|\vec{D}_{BCi} - \vec{D}_{3i}|} \ . \end{aligned} \tag{4.11}$$

Parametrisierung der Meridiangeometrie von Kreiselpumpen

Die Parametrisierung der Meridiangeometrie von Kreiselpumpenlaufrädern gestaltet sich etwas einfacher als die von Francis Turbinen, s. Bild 4.14.

In der Regel verlaufen sowohl die Deckscheiben- als auch die Nabenkontur im Zuströmbereich achsparallel, wodurch die Beschreibung der Kontrollpunkte D_{0a} und D_{0i} im Parametersatz wegfällt. Auch nach dem Laufrad liegen die beiden Konturlinien meist parallel zueinander, wodurch eine Parametrisierung der Endpunkte D_{3a} und D_{3i} ebenfalls überflüssig wird. Die Grenzwerte der Kontur werden im Parametrisierungsmodul mit $R_{3a} = R_{3i} = 1.5$ und $Z_{0a} = Z_{0i} = -1.5$ festgelegt.

Eine weitere Vereinfachung ergibt sich durch die stets gerade Austrittskante des Pumpenlaufrades. Über den Parameter R_{2i} lassen sich auch Laufräder mit schräger Austrittskante approximieren, was allerdings bei den für diese Arbeit aufbereiteten Pumpengeometrien nicht der Fall ist.

Wie bei der Parametrisierung der Meridiangeometrie von Francis Turbinen ist eine Approximation auch für diagonal durchströmte Pumpen möglich.

Die Streckenverhältnisse μ berechnen sich wie folgt:

$$\begin{aligned} \mu_{Aa} &= \frac{|\vec{D}_{Aa} - \vec{D}_{0a}|}{|\vec{D}_{ABa} - \vec{D}_{0a}|} \quad , \quad \mu_{Ba} = \frac{|\vec{D}_{Ba} - \vec{D}_{ABa}|}{|\vec{D}_{BCa} - \vec{D}_{ABa}|} \quad , \quad \mu_{Ca} = \frac{|\vec{D}_{Ca} - \vec{D}_{2a}|}{|\vec{D}_{BCa} - \vec{D}_{2a}|} \ , \\ \mu_{Ai} &= \frac{|\vec{D}_{Ai} - \vec{D}_{0i}|}{|\vec{D}_{ABi} - \vec{D}_{0i}|} \quad , \quad \mu_{Bi} = \frac{|\vec{D}_{Bi} - \vec{D}_{ABi}|}{|\vec{D}_{BCi} - \vec{D}_{ABi}|} \quad , \quad \mu_{Ci} = \frac{|\vec{D}_{Ci} - \vec{D}_{Di}|}{|\vec{D}_{BCi} - \vec{D}_{Di}|} \ . \end{aligned} \tag{4.12}$$

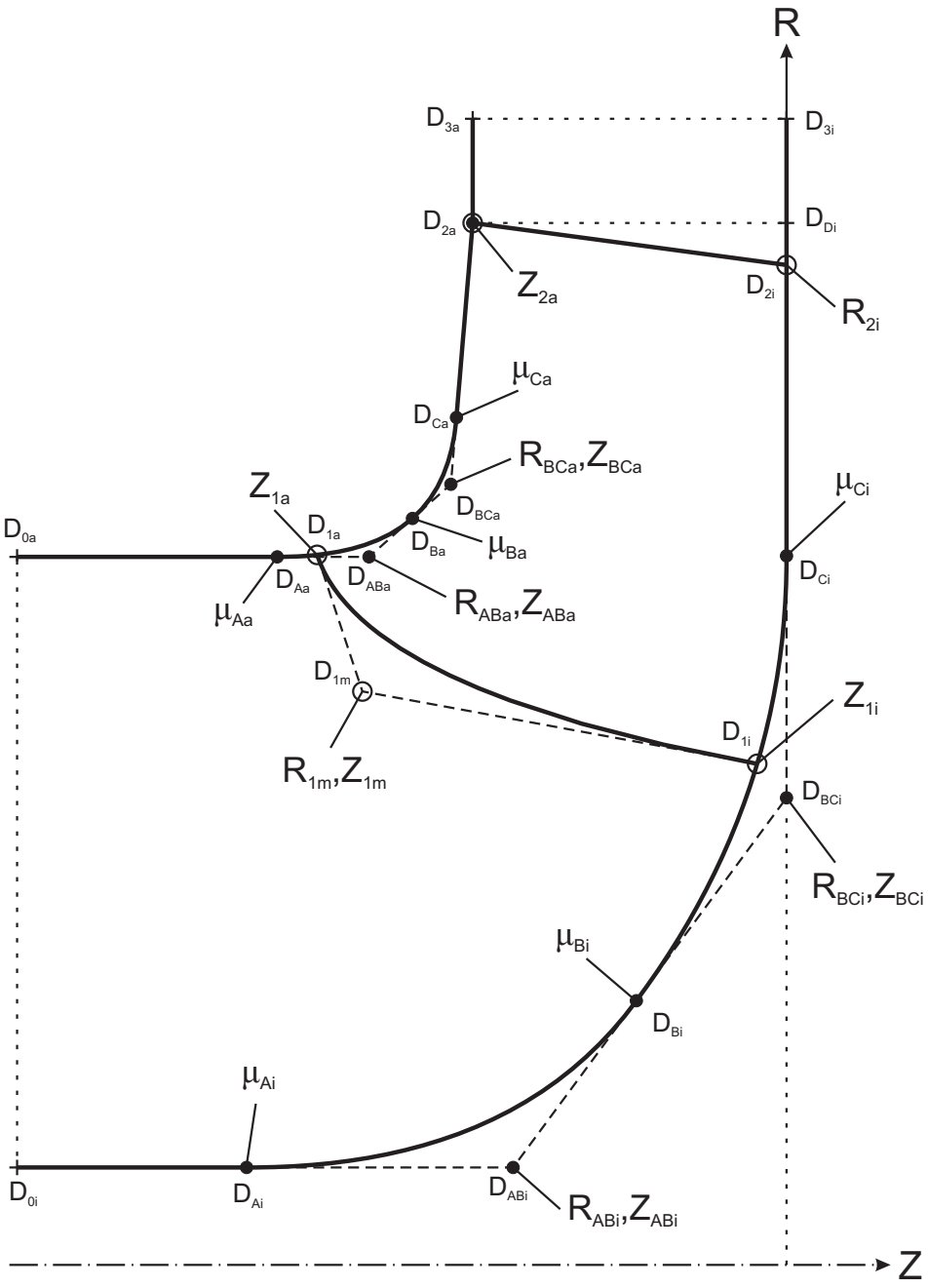


Bild 4.14: Parametrisierung der Meridiangeometrie einer Kreiselpumpe im KNN-Designsystem

4.3.2 Schaufelgeometrie

Die Basis für eine Parametrisierung der Schaufel stellen die auf den räumlich gekrümmten Rotationsflächen verlaufenden Skelettlinien dar, die bereits bei der Datenaufbereitung mit Hilfe von Teilfluträdern generiert worden sind. Zur Reduzierung der zu lernenden Geometrieparameter ist die geometrische Beschreibung der dreidimensionalen Schaufelskelettfläche durch fünf Skelettlinien in den Parametrisierungsmodulen festgelegt. Da die zweifach gekrümmten Rotationsflächen nicht abwickelbar sind, werden sie konform auf eine Zylindermantelfläche mit dem Radius r_{ref} unter

Verwendung der folgenden Berechnungsvorschriften abgebildet:

$$l = r_{ref} \cdot \int_0^m \frac{1}{r}\, dm \;, \tag{4.13}$$

$$u = r_{ref} \cdot \varphi \;. \tag{4.14}$$

Hierbei ist m die Meridiankoordinate entlang der Rotationsfläche, l die zugehörige konforme Länge und u die Umfangskoordinate der Zylindermantelfläche. Durch die konforme Abbildung ist somit eine unverfälschte Darstellung der Schaufelwinkel β_S möglich:

$$\tan \beta_S = -\frac{dl}{r_{ref} \cdot d\varphi} = -\frac{dm}{r \cdot d\varphi} \;. \tag{4.15}$$

Die Parametrisierung der fünf Schaufelskelettlinien erfolgt ebenfalls in der konformen Abbildung, wobei jede Skelettlinie in gleicher Weise behandelt wird.

Bild 4.15 zeigt am Beispiel einer Turbine die Parametrisierung der Skelettlinie entlang der Nabenkontur, Index 5. Da der Referenzradius r_{ref} für alle Geometrien nach der Normierung den Wert 1 hat, gilt mit

$$R = \frac{r}{r_{ref}} \,,\; M = \frac{m}{r_{ref}} \,,\; L = \frac{l}{r_{ref}} \,,\; U = \frac{u}{r_{ref}} = \frac{r_{ref} \cdot \varphi}{r_{ref}} \tag{4.16}$$

für die konform abgebildete Ebene:

$$L = \int_0^M \frac{dM}{R} \;, \tag{4.17}$$

$$U = \varphi \;, \tag{4.18}$$

$$\tan \beta_S = -\frac{dL}{d\varphi} = -\frac{dM}{R \cdot d\varphi} \;. \tag{4.19}$$

Die konforme Länge $L = 0$ bezieht sich dabei auf die Schaufeleintrittskante bei Turbinen bzw. auf die Austrittskante bei Pumpen. In φ Richtung ist die Schaufel bereits während der Datenaufbereitung ausgerichtet worden.

In einem ersten Schritt wird die Skelettlinie mit einem uniformen B-Spline 4. Ordnung approximiert. Zur Bestimmung der zu lernenden Geometrieparameter dient der Kontrollpunkt D_1 des B-Splines an der Eintrittskante mit den Koordinaten $(0, \Delta\varphi_{LE})$ als Referenzpunkt. Dadurch lassen sich die Parameter ΔL_g und $\Delta\varphi_g$ zur Beschreibung von Punkt D_2 sowie ΔL_B und $\Delta\varphi_B$ zur Beschreibung von Punkt D_B ableiten. Die Parametrisierung der Kontrollpunkte D_A und D_C erfolgt mit Hilfe der Schaufelwinkel β_{S1} bzw. β_{S2} am Ein- und Austritt, sowie der Umfangswinkel $\Delta\varphi_1$ bzw. $\Delta\varphi_2$.

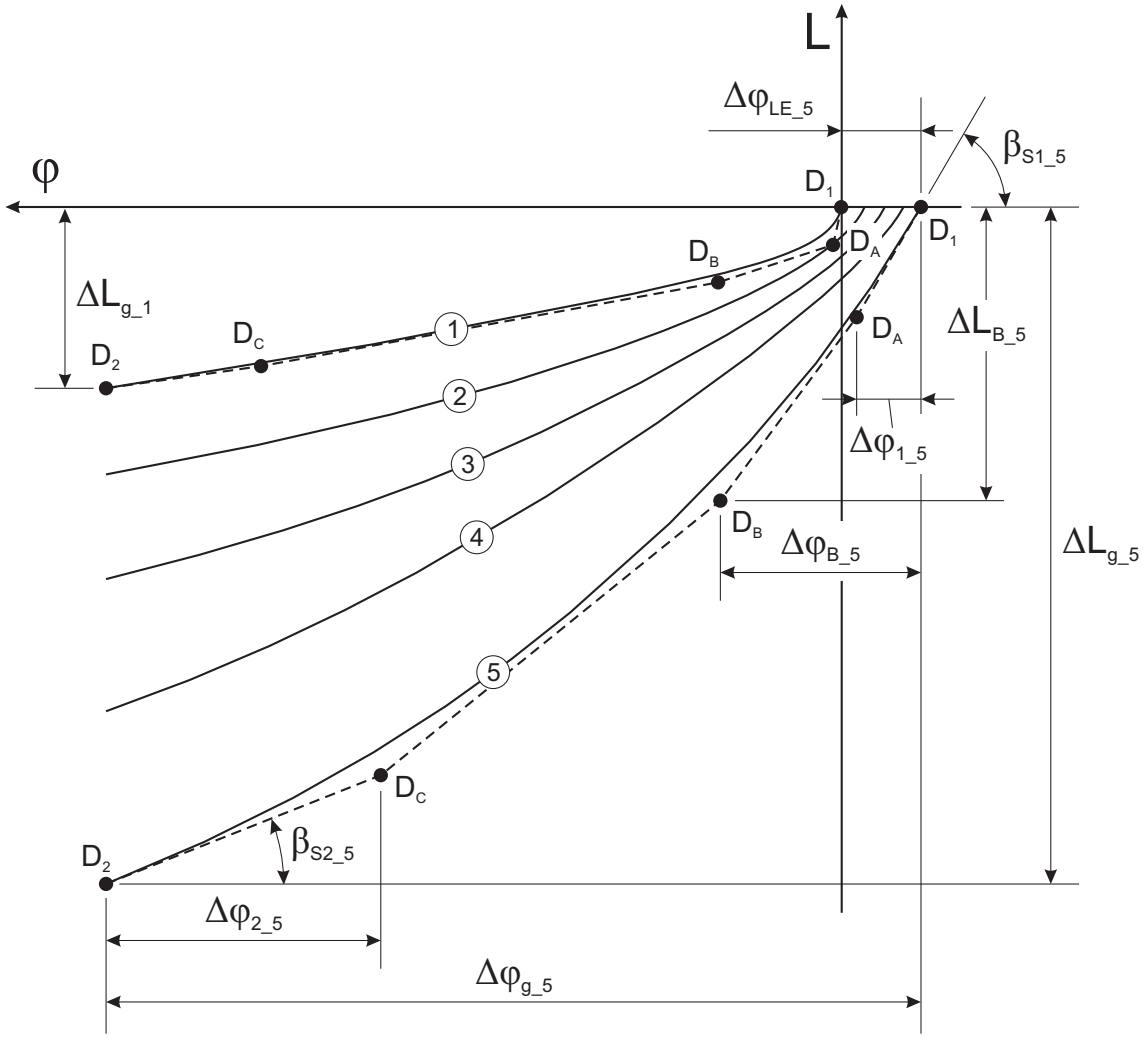


Bild 4.15: Parametrisierung der konformen Abbildung einer Turbinenschaufel in dimensionsloser Darstellung

Für die in dieser Arbeit gelernten Kreiselpumpengeometrien wird eine äquivalente Parametrisierung verwendet, bei der lediglich die Bezeichnung der Kontrollpunkte und der zugehörigen Geometrieparameter entsprechend der Pumpennomenklatur angepasst wurde.

Zusätzlich wurde im KNN-Designsystem für Pumpen noch eine alternative Parametrisierung der konformen Abbildung implementiert, die eine Erweiterung der gerade beschriebenen Parametrisierung darstellt, s. Bild 4.16.

Oftmals ist es bei Pumpenschaufeln erwünscht, den Schaufelwinkel β_S am Ein- bzw. Austritt über eine bestimmte Strecke konstant zu halten, um die Schaufel im Eintrittsbereich entlasten und am Austritt bei konstantem β_{S2} abdrehen zu können. In der konformen Abbildung stellen diese Bereiche jeweils eine Gerade dar, die durch die Punkte D_1 und D_A bzw. D_2 und D_E begrenzt und in den Lernmustern durch die Geometrieparameter β_{S1} und $\Delta\varphi_1$ bzw. β_{S2} und $\Delta\varphi_2$ dargestellt werden. Die B-Spline Approximation der Schaufel erfolgt dann zwischen den Punkten D_A und D_E mit der Bedingung, dass die Kurve tangential in die geraden Endstücke übergeht.

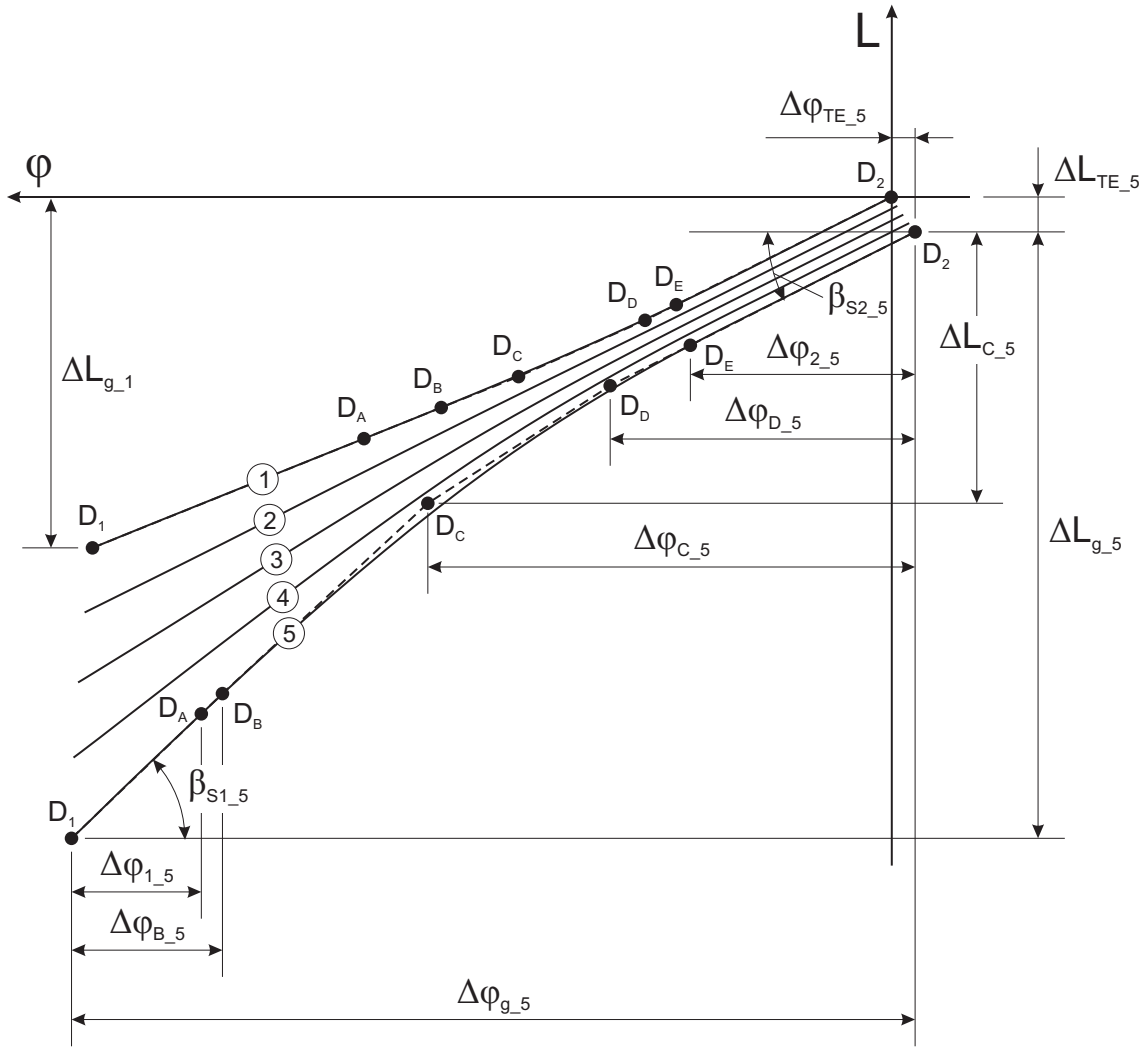


Bild 4.16: Parametrisierung der konformen Abbildung einer Kreiselpumpenschaufel in dimensionsloser Darstellung

Dadurch liegen die Punkte D_B und D_D stets auf einer Verlängerung der Geraden an den Skelettlinienenden und können somit ebenfalls mit den Geometrieparametern $\Delta\varphi_B$ und $\Delta\varphi_D$ sowie dem zugehörigen Schaufelwinkel β_{S1} bzw. β_{S2} beschrieben werden.

Zusätzlich ermöglicht diese Parametrisierung die Darstellung schräger Austrittskanten. Die konforme Länge $L = 0$ bezieht sich dabei auf eine gedachte achsparallele Austrittskante der Schaufel. Der Referenzpunkt D_2 wird somit durch die zwei Geometrieparameter ΔL_{TE} und $\Delta\varphi_{TE}$ beschrieben. Die Punkte D_1 und D_C liefern die restlichen Geometrieparameter ΔL_g, $\Delta\varphi_g$, ΔL_C und $\Delta\varphi_C$.

Zur Visualisierung der parametrisierten Geometrie stehen dem Anwender zwei interaktive Modifikationsbausteine zur Darstellung der Meridiankontur als auch der konformen Abbildung zur Verfügung. Damit der Anwender die Qualität der approximierten Geometrie beurteilen kann, wird die zugehörige Originalgeometrie, sofern vorhanden, ebenfalls im Grafikfenster angezeigt. Eine Modifikation der Geometrie ist

jederzeit durch interaktives Verschieben der parametrisierten Kontrollpunkte möglich.

4.4 Interaktive Systematisierung

Mit den aus dem Parametrisierungsmodul gewonnenen Geometrieparametern (GP) lässt sich zusammen mit einer fest vorgegebenen Anzahl unabhängiger Betriebspunktparameter (BP) der zugrunde liegenden Strömungsmaschine ein Lernmuster generieren, das in einer Datenbank abgespeichert wird. Tab. 4.1 zeigt den Aufbau dieser Datenbank.

Tabelle 4.1: Aufbau der Lernmusterdatenbank

Id	Eingabewerte				Zielwerte			
1	BP_{11}	BP_{12}	...	BP_{1n}	GP_{11}	GP_{12}	...	GP_{1m}
2	BP_{21}	BP_{22}	...	BP_{2n}	GP_{21}	GP_{22}	...	GP_{2m}
⋮	⋮	⋮		⋮	⋮	⋮		⋮
p	BP_{p1}	BP_{p2}	...	BP_{pn}	GP_{p1}	GP_{p2}	...	GP_{pm}

Nachdem genügend Lernmuster in der Datenbank vorhanden sind, beginnt für den Anwender die eigentliche Arbeit.

Beim Trainieren der Lernmuster versucht das neuronale Netz einen funktionalen Zusammenhang zwischen den Eingabe- und den Zielwerten herzustellen. Abhängig von der Art der Lernmuster, den gewählten Lernparametern sowie der Topologie des verwendeten neuronalen Netzes können die Lerndaten mit der ermittelten Abbildungsfunktion nur bis zu einem bestimmten Grad genau approximiert werden. Die Eigenschaft, einen nicht in der Lerndatenbank enthaltenen Eingabevektor im Rahmen der Approximationsgenauigkeit richtig abzubilden, bezeichnet man als *Generalisierung*. Die Generalisierungsfähigkeit des Netzes verschlechtert sich jedoch zunehmend mit höher werdender Approximationsgenauigkeit bezüglich der Lernmuster. Aufgrund der Forderung nach einer hohen Approximationsgenauigkeit bei den zu lernenden Strömungsmaschinengeometrien lässt sich eine optimale Generalisierung somit nur zum Teil durch ein richtig eingestelltes neuronales Netz realisieren. Vor allem durch die hohe Anzahl an Freiheitsgraden während der Auslegung und der Optimierung einer Strömungsmaschine unterliegen die berechneten Geometrieparameter oft sehr großen Schwankungen, die zusammen mit der geforderten Approximationsgenauigkeit zu unbefriedigenden Generalisierungseigenschaften des neuronalen Netzes führen.

Eine gründliche Analyse der Geometrieparameter mit anschließender Systematisierung der Lernmuster ist daher zur Generierung akzeptabler Geometrien aus dem neuronalen Netz zwingend erforderlich. Hierfür existieren im KNN-Designsystem, wie bereits erwähnt, mehrere interaktive Modifikationsbausteine.

Interaktive Lernmustermodifikation

Mit Hilfe des im System integrierten grafischen Lernmustermoduls können die in der Datenbank enthaltenen Lernmuster visualisiert und modifiziert werden, s. Bild 4.17.

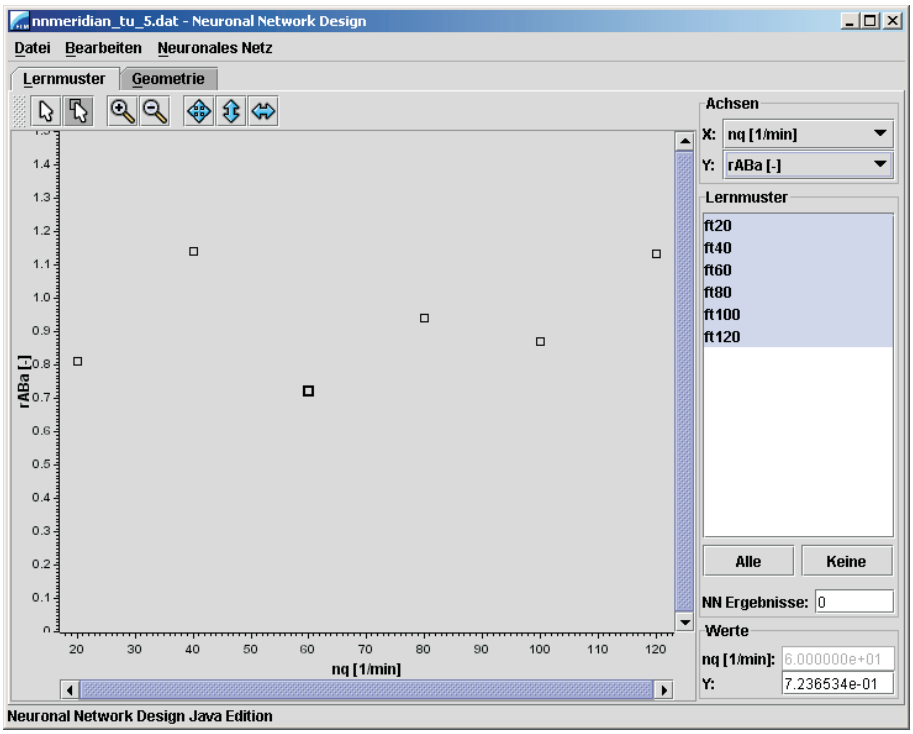


Bild 4.17: Interaktive Lernmustermodifikation im KNN-Designsystem

Durch die Wahl eines Eingabeparameters auf der x-Achse und eines Zielparameters auf der y-Achse lassen sich die jeweiligen Eingabe- und Zielwerte der selektierten Lernmuster beliebig übereinander auftragen und interaktiv oder manuell durch Eingabe exakter Werte anpassen.

Bei der Systematisierung der in den Lernmustern enthaltenen Geometrieparameter besteht bei diesem Modifikationsbaustein allerdings die Gefahr, sich durch größere Änderungen der Werte zu weit von der entsprechenden Originalgeometrie zu entfernen.

Interaktive Geometriemodifikation

Damit die bei der interaktiven Lernmustermodifikation gemachten Änderungen überprüft werden können, werden die Geometrieparameter mit dem ebenfalls im System integrierten Geometriemodul synchronisiert.

Je nachdem, ob es sich bei den zu lernenden Geometriedaten um Meridiangeometrien oder die konformen Abbildungen von Beschaufelungen handelt, wird das entsprechende Geometriemodul gemäß Bild 4.18 bzw. 4.19 vom Programmsystem zur Laufzeit instanziiert. Zum Vergleich werden in den Geometriemodulen sowohl die parametrisierte Geometrie (rot) mit seinen zugehörigen Kontrollpolygonen (schwarz)

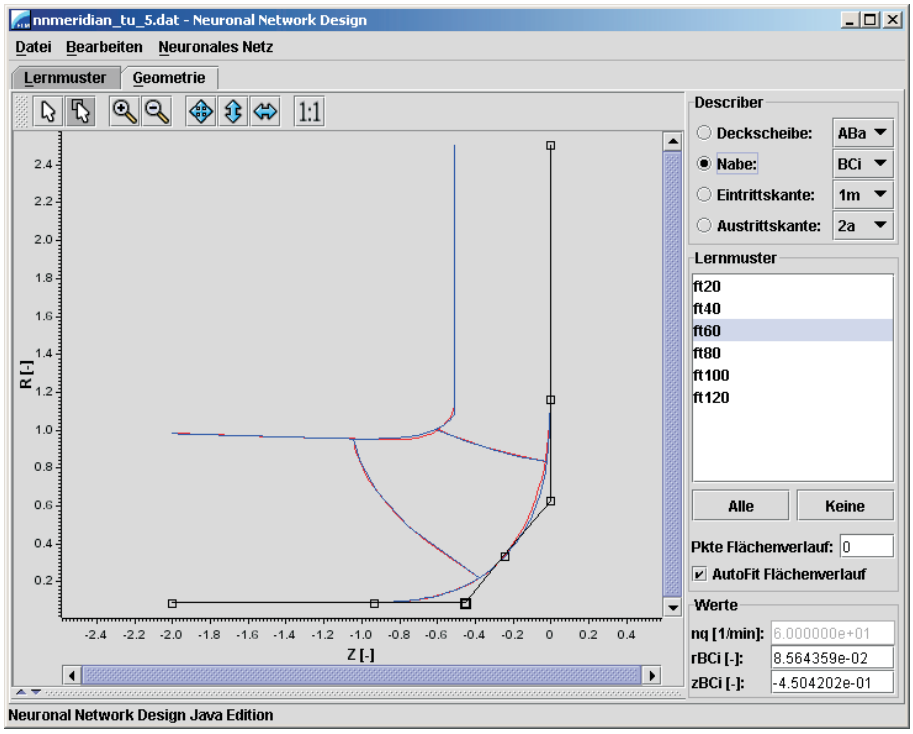


Bild 4.18: Interaktive Geometriemodifikation der Meridiankontur

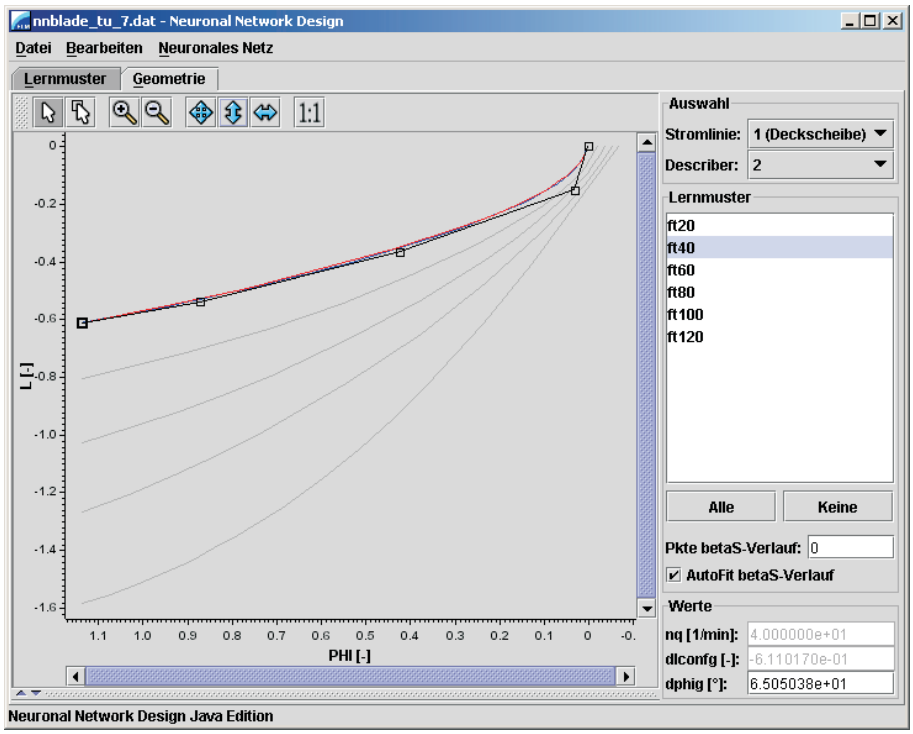


Bild 4.19: Interaktive Geometriemodifikation der konformen Abbildung der Schaufel

als auch die Originalgeometrie (blau) dargestellt. Durch Auswahl der entsprechenden Kontrollpunkte lässt sich auch in diesen Modulen die Geometrie interaktiv bzw. manuell durch Eingabe der zugehörigen Geometrieparameter verändern. Das Geome-

triemodul zur Modifikation von Meridiankonturen ermöglicht zudem die gleichzeitige Darstellung von Meridiankonturen mehrerer Lernmuster.

Durch abwechselndes Verändern der Lernmuster und der Geometrie lassen sich die Daten über die Modifikationsbausteine relativ zuverlässig systematisieren, ohne dass sich auffallend große Abweichungen zur jeweiligen Originalgeometrie ergeben. Eine bereits bei der Generierung und Optimierung der Originalgeometrien systematische Vorgehensweise ist in jedem Fall empfehlenswert und für eine nachträgliche Systematisierung im KNN-Designsystem sehr hilfreich.

4.5 Trainieren des künstlichen neuronalen Netzes

Das KNN-Designsystem wurde primär als grafische Benutzeroberfläche zur Visualisierung und Modifikation beliebiger Lernmustern sowie zum Trainieren und Abrufen eines Backpropagation Netzes entwickelt.

Da der programmierte Simulator zum Trainieren neuronaler Netze als eigenständiges Konsolenprogramm agiert, wurde im KNN-Designsystem ein Modul zur Steuerung des ausgegliederten Simulators entwickelt. Das Modul ermöglicht dem Anwender das Starten und Stoppen des Lernprozesses, das Überspringen des Trainings eines Teilnetzes im Falle von MultiBPG, s. Kap. 3.5.7, das Reinitialisieren der Verbindungsgewichte während des Lernens und das Abrufen eines trainierten Netzwerks.

Damit das KNN-Designsystem während des Lernens weiterhin bedienbar bleibt, wird der Neuronale-Netze-Simulator vom Steuerungsmodul in einem vom KNN-Designsystem unabhängigen Prozess gestartet. Dadurch ist der Anwender auch in der Lage, sowohl ein neues Projekt zu bearbeiten als auch mehrere unterschiedlich konfigurierte Backpropagation Netze gleichzeitig zu trainieren.

Die Lernparameter für den Backpropagation Algorithmus sowie die Topologie des neuronalen Netzes lassen sich bequem über ein entsprechendes Dialogfenster im KNN-Designsystem konfigurieren. Hier werden auch die zu lernende Datenbank und der Dateiname zum Abspeichern der gelernten Verbindungsgewichte des Netzes angegeben. Um dem Anwender die Einstellungen der Lernparameter zu vereinfachen, schlägt das Programmsystem die in Tab. 4.2 aufgelisteten Standardwerte vor, die für die meisten neuronalen Netze gute Lernergebnisse erzielen.

Tabelle 4.2: Standardwerte der Lernparameter

Lernrate η	Momentum α	Weight Decay d	Lernmodus
0.3	0.9	0.0	Batch

Da die Anzahl der Neuronen in der Eingabe- bzw. Ausgabeschicht durch die Lernmuster vorgegeben ist, werden im KNN-Designsystem zur Bestimmung der Netztopologie lediglich die Zahl der verdeckten Schichten sowie die jeweilige Anzahl Neuronen pro Schicht angegeben. Die Wahl der richtigen Topologie ist dabei von vielen

Faktoren abhängig und lässt sich nur empirisch bestimmen. Die Literatur empfiehlt, sich auf eine, maximal zwei verdeckte Neuronenschichten zu beschränken und zur genaueren Approximation der Lernmuster eher die Anzahl Neuronen einer Schicht allmählich zu erhöhen. Beim Festlegen der Topologie sollte darauf geachtet werden, dass ein neuronales Netz nicht zu lange trainiert wird, um den im KNN-Designsystem eingestellten Netzfehler E_{min} zu erreichen. Unter bestimmten Bedingungen lässt sich bei Feedforward-Netzen das Phänomen des *Übertrainierens* beobachten. Dabei werden die Lernmuster zwar immer besser approximiert, die Generalisierungsfähigkeit des Netzes verschlechtert sich allerdings ab einem gewissen Punkt zunehmend, wodurch sich die geforderte Approximationsgenauigkeit eines nicht gelerntes Testmuster eventuell nicht mehr erreichen lässt. Hierfür kann im KNN-Designsystem neben dem zu erreichenden Netzfehler auch die Anzahl der maximal zu durchlaufenden Lernzyklen angegeben werden.

Bevor das Training beginnt, müssen die Eingabe- und Zielwerte der Lernmuster vom Simulator in die jeweiligen Arbeitsintervalle des neuronalen Netzes transformiert werden. Hierfür müssen im KNN-Designsystem die minimal bzw. maximal möglichen Grenzwerte für die in den Lernmustern enthaltenen Eingabe- und Zielwerte angegeben werden.

Während des Lernprozesses werden der Netzfehler des neuronalen Netzes in Abhängigkeit von den bereits durchlaufenen Lernzyklen als auch von der verstrichenen Zeit sowie statistische Informationen über die gelernten Verbindungsgewichte in einer Datei mitprotokolliert.

Nachdem das neuronale Netz trainiert wurde, kann das Lernergebnis anschließend im KNN-Designsystem zur Analyse visualisiert werden, s. Kap. 4.7. Ist das Lernergebnis akzeptabel, so ist das neuronale Netz in der Lage, in Sekundenbruchteilen ein vernünftiges Ausgabemuster zu einem beliebigen Eingabemuster im Bereich der angegebenen Grenzwerte zu generieren.

4.6 Generierung neuer Geometrien

Um eine dreidimensionale Schaufel aus dem KNN-Designsystem heraus zu generieren, müssen die konform abgebildeten Schaufelskelettlinien wieder zurück auf die zugehörigen Rotationsflächen zwischen Deckscheibe und Nabe abgebildet werden. Da die Lage der Rotationsflächen einer aus dem neuronalen Netz abgerufenen Schaufelgeometrie für einen beliebigen Betriebspunkt allerdings nicht bekannt ist, müssen diese zuerst erzeugt werden. Da schon bei der Erstellung der Lernmuster auf eine Systematisierung der Rotationsflächen, vgl. 4.2.4, geachtet wurde, lassen sich die fünf geforderten Rotationsflächen in gleicher Weise aus der zugehörigen Meridiankontur berechnen. Die Generierung einer dreidimensionalen Schaufel im KNN-Designsystem erfolgt daher in zwei Schritten.

In einem ersten Schritt muss der Anwender eine zum gleichen Betriebspunkt passende Meridiangeometrie aus dem in Kap. 4.3.1 beschriebenen neuronalen Netz abrufen. Die erhaltenen Geometrieparameter werden zur weiteren Verarbeitung in einer Datei

abgelegt. Zur weiteren Verwendung der Meridiangeometrie in den lehrstuhleigenen Entwurfs- und Optimierungssystemen lässt sich die Deckscheiben- und Nabenkontur zusätzlich im FLM internen Geometrieformat abspeichern.

Im zweiten Schritt wird die konforme Abbildung aus dem neuronalen Netz für Schaufelgeometrien abgerufen. Zur Konvertierung der Geometrieparameter der Schaufel werden zunächst mit Hilfe der vorher erstellten meridionalen Geometrieparameter die fünf gesuchten Rotationsflächen berechnet. Zusätzlich wird die konforme Länge zwischen der Ein- und Austrittskante entlang der Rotationsflächen bestimmt. Je nachdem wie gut die Approximation sowie die Generalisierung des neuronalen Netzes ist, kann sich die konforme Länge zwischen der Ein- und Autrittskante der Meridiankontur von der konformen Länge ΔL_g der Schaufelgeometrie unterscheiden. Hierbei kommt es zu einem Konflikt bei der Berechnung der dreidimensionalen Skelettlinien. Wird für die Berechnung die konforme Länge der abgerufenen Schaufelgeometrie verwendet, so wird zwar der Schaufelwinkelverlauf korrekt wiedergegeben, allerdings kann sich dadurch eine der Schaufelkanten verschieben, die unter Umständen auch wellig sein kann. Im anderen Fall ergibt sich eine Veränderung des Schaufelwinkelverlaufs, bei gleich bleibender Lage der Schaufelkanten. Da aber die Nachbearbeitung einer falsch liegenden und welligen Schaufelkante ebenfalls Auswirkungen auf den Schaufelwinkelverlauf hat, ist die zweite Variante, also die Beibehaltung der Schaufelkanten in der Meridiankontur vorzuziehen. Da die Parametrisierung der Skelettlinien zudem die Schaufelwinkel β_{S1} und β_{S2} vorsieht, ändert sich der Schaufelwinkelverlauf lediglich im mittleren Bereich der Skelelettlinie. Die in Bild 4.15 bzw. 4.16 dargestellten konformen Längen ΔL_g und ΔL_B bzw. ΔL_C der in Kap. 4.3.2 vorgestellten Parametrisierungen werden zur Einhaltung der Streckenverhältnisse entsprechend skaliert. Danach werden die Skelettlinien auf die dreidimensionalen Rotationsflächen umgerechnet und im FLM internen Geometrieformat abgespeichert.

Die aus den neuronalen Netzen abgerufenen Meridiankonturen und konformen Abbildungen lassen sich im KNN-Designsystem visualisieren und gegebenenfalls nachbearbeiten. Die nachträglich modifizierte Geometrie kann dann zur Verbesserung der Netzwerkausgabe ohne erneute Parametrisierung in die jeweilige Lernmusterdatenbank übernommen und nachtrainiert werden.

4.7 Analysewerkzeuge

Um die parametrisierten Geometrien besser beurteilen zu können, lässt sich im KNN-Designsystem sowohl der meridionale Flächenverlauf zwischen Deckscheiben- und Nabenkontur als auch der Schaufelwinkelverlauf der Skelettlinien in den Geometriemodulen darstellen. Durch die gleichzeitige Anzeige der parametrisierten Geometrie sowie des zugehörigen Originals ist eine genauere Anpassung an die Originalgeometrie während der interaktiven Modifikation möglich.

Ebenso verfügt das KNN-Designsystem über eine Möglichkeit, die Lernergebnisse des neuronalen Netzes zu visualisieren.

4.7.1 Meridionaler Flächenverlauf

Die Bestimmung des bereits in Kap. 2.2.3.1 erwähnten meridionalen Flächenverlaufs erfolgt durch Auskreisen der Meridiankontur, s. Bild 4.20.

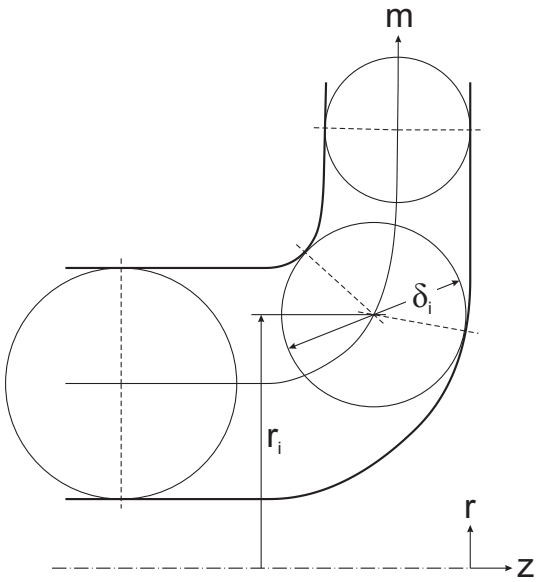


Bild 4.20: Auskreisen der Meridiankontur

Die Kreismittelpunktsradien r_i sowie die zugehörigen Durchmesser δ_i der Kreise, die zur Berechnung der für den Flächenverlauf benötigten Querschnittsflächen

$$A_i = 2\pi\, r_i \cdot \delta_i \tag{4.20}$$

erforderlich sind, werden in einem iterativen Verfahren bestimmt, das auf den Überlegungen von Kronschnabl [23] zur algebraischen Generierung von Strohmbahnen basiert. Wie in Bild 4.21 schematisch dargestellt, werden hierfür in einem ersten Schritt Hilfslinien zwischen der diskreten Deckscheiben- und Nabenkontur konstruiert, auf denen die Mittelpunkte der Kreise ermittelt werden.

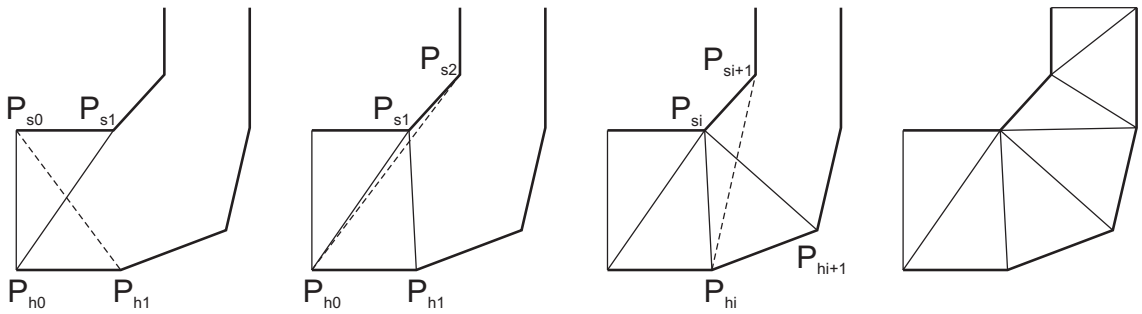


Bild 4.21: Triangulation der Meridiankontur

Ausgehend von der ersten Hilfslinie $P_{s0}P_{h0}$, die sich aus den jeweiligen Anfangspunkten der Deckscheibe und der Nabe ergibt, werden die restlichen Hilfslinien wie

folgt ermittelt: Falls der Abstand zwischen den Punkten P_{si} und P_{hi+1} kleiner ist als der Abstand zwischen P_{hi} und P_{si+1}, dann ist $P_{si}P_{hi+1}$ Hilfslinie und si wird zur Bestimmung der nächsten Hilfslinie um 1 erhöht; andernfalls ist $P_{hi}P_{si+1}$ Hilfslinie und hi wird um 1 erhöht. Diese Prozedur wird solange wiederholt, bis die gesamte Meridiankontur *trianguliert* ist. Der Vorteil der Triangulation liegt zum einen darin, dass stets die Punkte mit dem kürzesten Abstand miteinander verbunden sind, und zum anderen, dass sich die Hilfslinien zwischen Deckscheibe und Nabe nicht überschneiden und somit eine geordnete Lage der Kreismittelpunkte gewährleistet wird. Zudem kann die Anzahl Punkte auf der Deckscheiben- und Nabenkontur unterschiedlich sein.

Danach beginnt die iterative Suche nach dem jeweiligen Mittelpunkt der Auskugelungskreise entlang einer Hilfslinie, s. Bild 4.22. Ausgehend von der Mitte der Hilfslinie, Punkt 1, wird jeweils der kleinste Abstand zur Deckscheibe bzw. zur Nabe berechnet. Sind die Abstände ungleich, so wird der Kreismittelpunkt entlang der Hilfslinie entsprechend der halben Abstandsdifferenz $\frac{1}{2}\Delta a$ verschoben. Dieser Vorgang wird solange wiederholt, bis der Betrag der Differenz $|\Delta a|$ annähernd 0 ist. Zusätzlich zum berechneten Mittelpunkt, Punkt 2, stehen zur weiteren Verwendung auch die beiden Berührungspunkte des Kreises auf der Deckscheiben- bzw. Nabenkontur zur Verfügung.

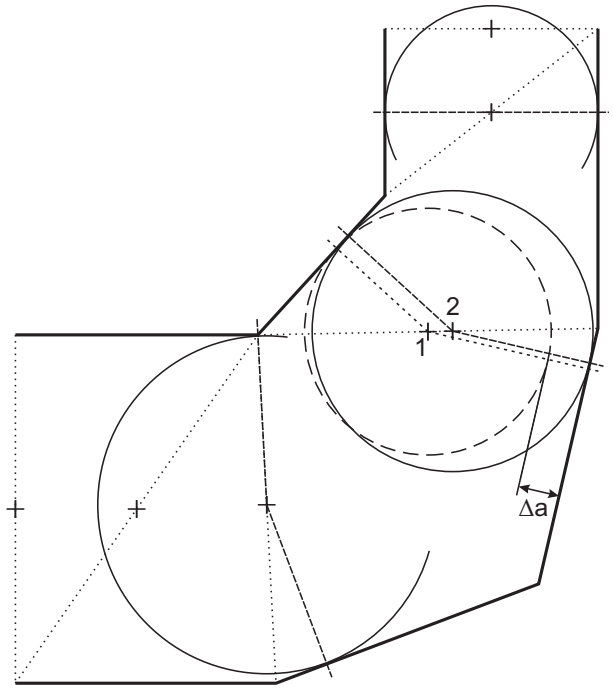


Bild 4.22: Iterative Bestimmung der einbeschriebenen Kreise

Im KNN-Designsystem wird sowohl die Mittelpunktslinie der Auskugelungskreise in der Meridianebene visualisiert als auch die Lage der Schaufelein- und -austrittskante zur Orientierung im normierten Flächenverlauf angedeutet, s. Bild 4.23.

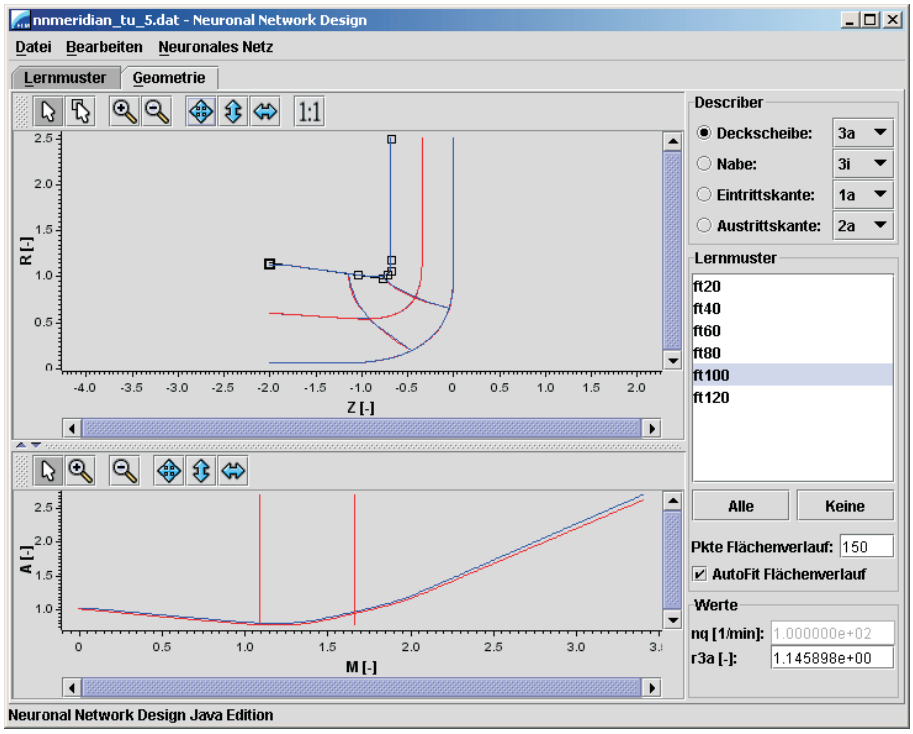


Bild 4.23: Flächenverlauf und Mittelpunktslinie im KNN-Designsystem

4.7.2 Schaufelwinkelverlauf

Zur Darstellung des Schaufelwinkelverlaufs im KNN-Designsystem müssen zunächst die Schaufelwinkel β_S entlang einer Schaufelskelettlinie berechnet werden. Gemäß Gl. 4.19 ist der Schaufelwinkel in der konformen Abbildung wie folgt definiert:

$$\beta_S = \arctan\left(-\frac{dL}{d\varphi}\right) \ . \tag{4.21}$$

Die Berechnung der Schaufelwinkel für eine diskrete Skelettlinie der Originalgeometrie erfolgt dabei mit einer Genauigkeit zweiter Ordnung:

$$\beta_{S,i} = \arctan\left(-\frac{L_{i+1} - L_{i-1}}{\varphi_{i+1} - \varphi_{i-1}}\right) \ . \tag{4.22}$$

Um den Verlauf für eine parametrisierte Skelettlinie zu bestimmen, wird der Schaufelwinkel über die Ableitung der beschreibenden B-Spline Kurve berechnet, s. Bild 4.24.

Für einen Punkt $\vec{P}(t)$ der B-Spline Kurve mit t als Kurvenparameter ergibt sich die Ableitung in diesem Punkt nach Gl. 2.25 zu:

$$\vec{P}'(t) = \sum_{i=1}^{n} \vec{D}_i \cdot N'_{i,k}(t) = \begin{pmatrix} \frac{dL(t)}{dt} \\ \frac{d\varphi(t)}{dt} \end{pmatrix} , \tag{4.23}$$

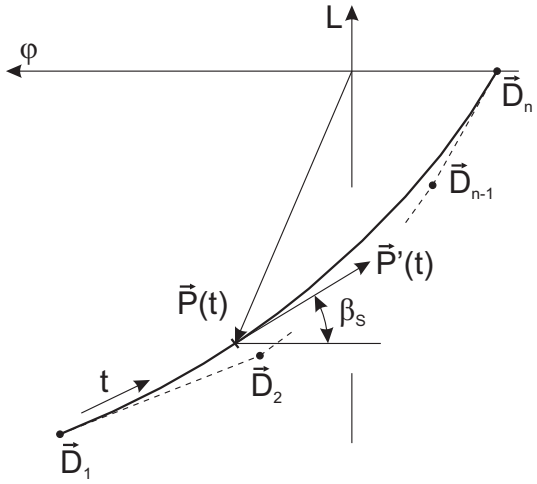


Bild 4.24: Bestimmung des Schaufelwinkels β_S entlang einer B-Spline Kurve

wobei $N'_{i,k}(t)$ die in Gl. 2.27 definierten Ableitungen der Basisfunktionen des B-Splines sind. Der Schaufelwinkel β_S im Punkt $\vec{P}(t)$ lässt sich dann durch Einsetzen der einzelnen Koordinatenableitungen aus Gl. 4.23 bestimmen:

$$\beta_S = \arctan\left(-\frac{\frac{dL(t)}{dt}}{\frac{d\varphi(t)}{dt}}\right) . \tag{4.24}$$

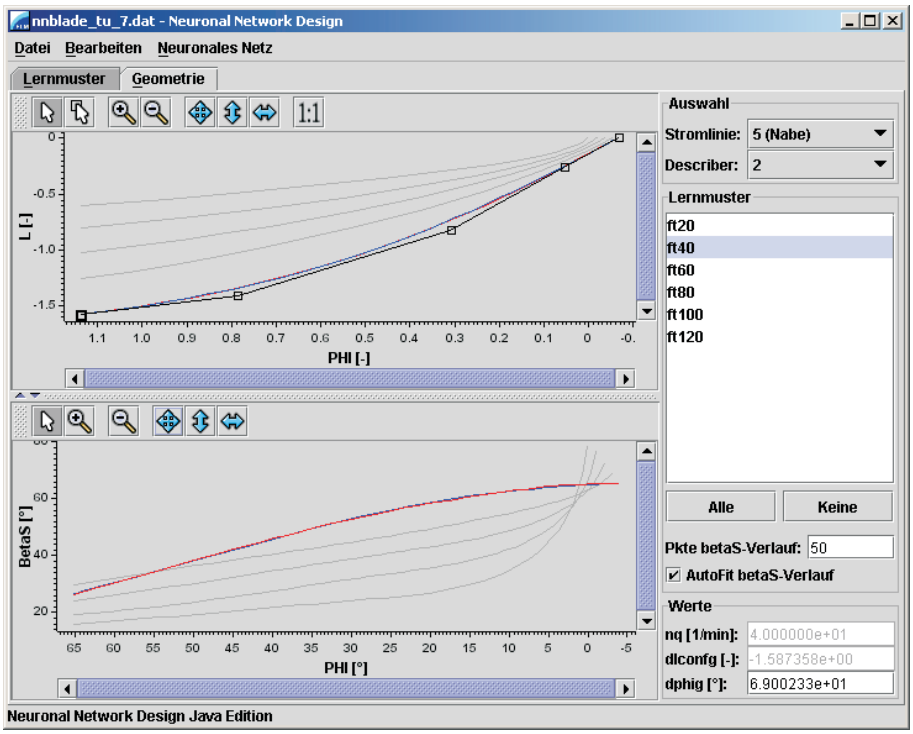


Bild 4.25: Schaufelwinkelverläufe im KNN-Designsystem

Bild 4.25 zeigt die Darstellung der einzelnen Schaufelwinkelverläufe im KNN-Designsystem.

4.7.3 Visualisierung der Lernergebnisse

Damit die Approximationsgenauigkeit und Generalisierungsfähigkeit des trainierten neuronalen Netzes überprüft werden können, lässt sich der Verlauf der gelernten Abbildungsfunktion des Netzes im Lernmustermodul des KNN-Designsystems visualisieren, s. Bild 4.26.

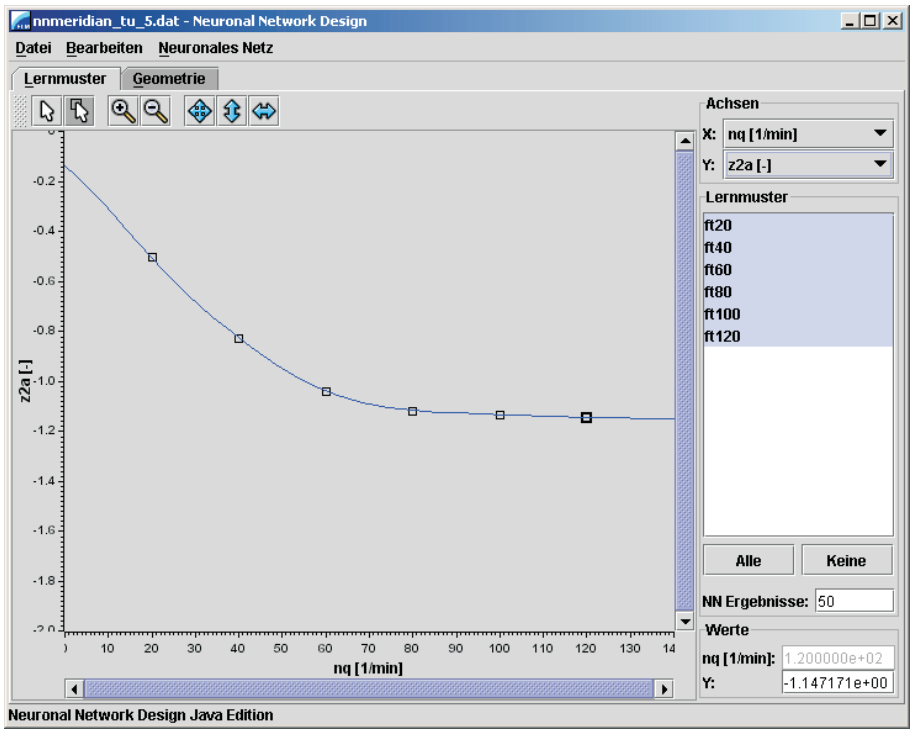


Bild 4.26: Lernergebnisse im KNN-Designsystem

Ausgangsbasis für die Darstellung des Kurvenverlaufs ist die Selektion eines Lernmusters aus der Datenbank. Abhängig von der Wahl des Eingabeparameters i zur Darstellung auf der x-Achse wird ein Satz von p Eingabevektoren generiert, dessen Eingabewerte x_{pi} das Begrenzungsintervall $[x_{i,min}, x_{i,max}]$ in $(p-1)$ gleich große Teilintervalle aufteilen. Die restlichen Eingabewerte $x_{pi'}$ der Eingabevektoren für $i' \neq i$ werden unverändert aus dem selektierten Lernmuster übernommen.

Die generierten Eingabevektoren werden dem trainierten neuronalen Netz zur Abbildung präsentiert und die entsprechenden Ausgabewerte y_{pj} des zur Darstellung auf der y-Achse ausgewählten Ausgabeparameters j anschließend in Abhängigkeit des zugehörigen Eingabewertes x_{pi} angezeigt.

Die Selektion mehrerer Lernmuster zur Darstellung der Kurvenverläufe in unterschiedlichen Ebenen der vom neuronalen Netz aufgespannten Hyperfläche ist ebenfalls möglich.

Kapitel 5

Validierung des Backpropagationverfahrens

5.1 Verifikation des Lernalgorithmus

Eine einfache Verifikation des für diese Arbeit programmierten Backpropagation Lernalgorithmus ist das Erlernen der logischen Verknüpfung XOR, s. Tab. 5.1.

Tabelle 5.1: Die Lernmuster der logische Verknüpfung XOR

Id	Eingabe		Ziel
p	x_1	x_2	y
1	0	0	0
2	0	1	1
3	1	0	1
4	1	1	0

Da die korrekte Abbildung aller binären Lernmuster der logischen Verknüpfung XOR nur mit einem mehrstufigen Netzwerk möglich ist, wird für das zu trainierende Netz eine 2-2-1 Topologie gewählt, s. Bild 5.1.

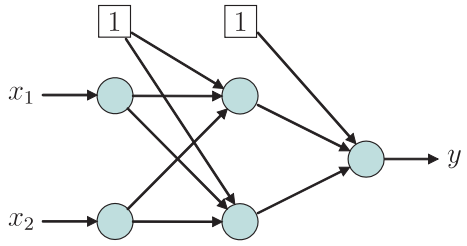


Bild 5.1: Neuronales Netz mit 2-2-1 Topologie für das Trainieren des XOR-Problems

Bei der Lösung des XOR-Problems geht es in erster Linie darum, die Konvergenz- und Approximierungseigenschaften des Backpropagation Verfahrens zu überprüfen.

Hierfür wird bei den verwendeten Lernparametern in Tab. 5.2 ein extrem niedriger quadratischer Gesamtfehler E_{min} des Netzes als Konvergenzkriterium vorgegeben.

Tabelle 5.2: Lernparameter für den Backpropagation Algorithmus

η	α	d	E_{min}
0.3	0.9	0.0	10^{-20}

Zum Vergleich werden die Lernmuster jeweils zweimal mit exakt den gleichen Lernparametern sowohl im Batch- als auch im Online-Modus trainiert. Bild 5.2 zeigt den Fehlerverlauf während des Trainings des XOR-Problems der vier Testdurchläufe.

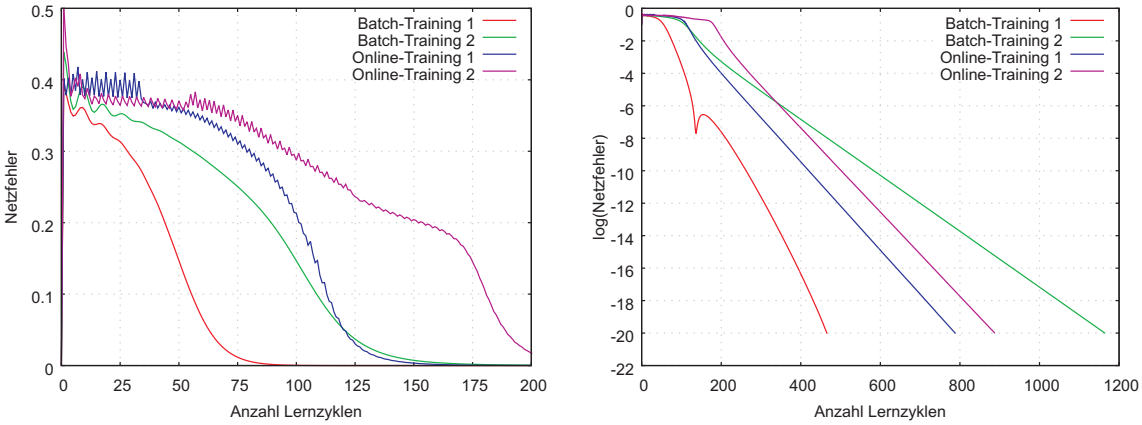


Bild 5.2: Normal (links) und logarithmisch (rechts) dargestellter Konvergenzverlauf des Netzfehlers der einzelnen Lerntests

Die unterschiedlichen Konvergenzverläufe der jeweils äquivalenten Lerntests *Batch-Training 1* und *2* bzw. *Online-Training 1* und *2* rühren von den zufällig initialisierten Startgewichten des Netzes her, wodurch das Lernen immer an einer anderen Stelle der von den Verbindungsgewichten aufgespannten Fehlerfläche beginnt.

Man erkennt zudem deutliche Unterschiede im Konvergenzverhalten zwischen den unterschiedlichen Lernverfahren. Während die Oszillation des Netzfehlers beim Batch-Training bereits in den ersten Lernzyklen vollständig abnimmt, sind die Schwankungen während des Online-Trainings wesentlich intensiver und können im Laufe des Lernens sprungartig zu- oder abnehmen. Diese intensiven Schwankungen lassen sich dadurch erklären, dass die Gewichte des Netzes nach jedem zugeführten Lernmuster verändert werden und die Lernmuster zudem nach jeder gelernten Epoche neu vermischt werden. Dies führt zwangsweise zu unterschiedlich starken Sprüngen auf der Fehlerfläche, die dem Netzwerk eine durchaus gewollte Eigendynamik verleihen, um der Möglichkeit, in einem suboptimalen lokalen Minimum der Fehlerfläche zu landen, entgegenzuwirken.

Das Training dauerte auf einem handelsüblichen Rechner mit 2.2 GHz für alle durchgeführten Lerntests weniger als eine Sekunde.

Zur Bewertung der Approximationseigenschaften des Netzes ist in Bild 5.3 der Approximationsfehler für die Ausgabewerte y der trainierten neuronalen Netze dargestellt. Man erkennt, dass der Fehler zu den Trainingsmustern sehr gering ist und die logische Verknüpfung XOR somit in allen Testfällen richtig gelernt wurde.

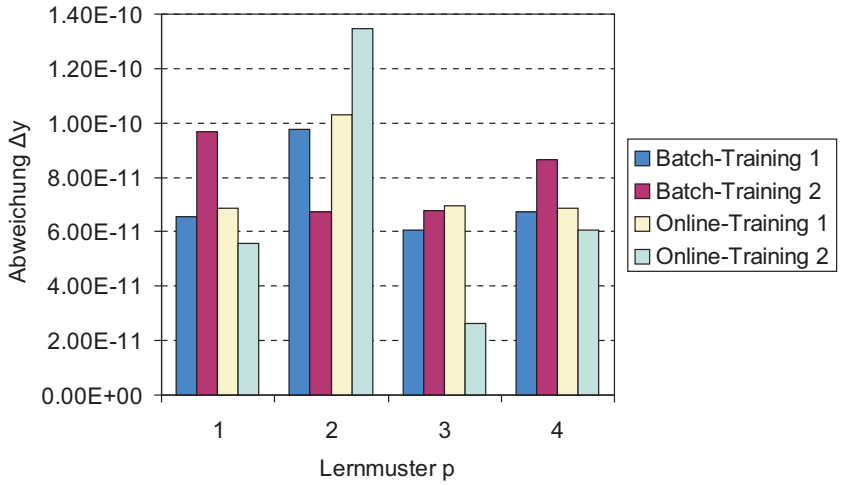


Bild 5.3: Abweichung der Netzwerkausgabe zu den vier Lernmustern des XOR Problems für jeweils zwei äquivalente Lerntests im Batch- bzw. Online Modus

5.2 Generalisierungseigenschaften des Netzes

Neben der Fähigkeit des neuronalen Netzes, die gelernten Trainingsmuster zu approximieren, muss beim Trainieren von Strömungsmaschinengeometrien vor allem auch

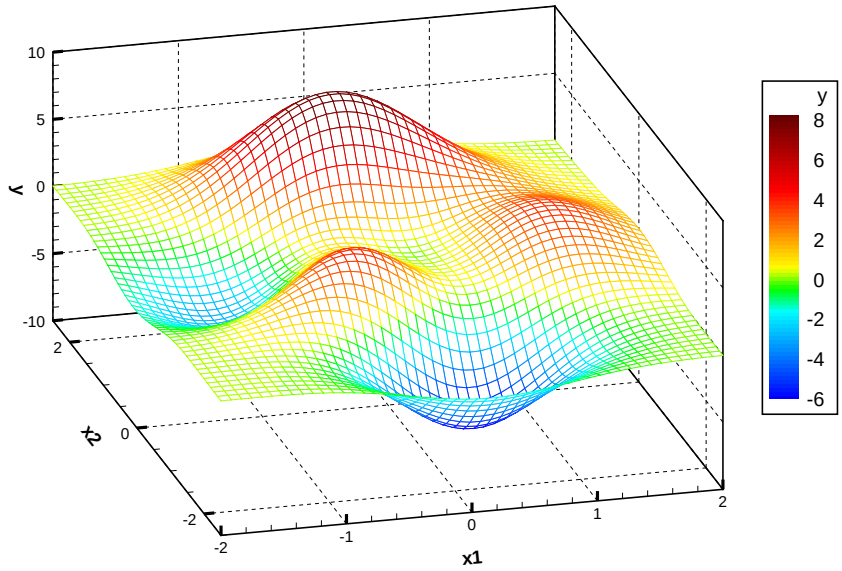


Bild 5.4: Darstellung der analytischen Testfunktion im Bereich $[-2, 2] \times [-2.5, 2.5]$

auf die Generalisierungseigenschaften des Netzes geachtet werden. Um das Verhalten eines neuronalen Netzes zu untersuchen, auf unbekannte Testdaten zu reagieren, wird eine von den Lernmustern unabhängige Validierungsmenge an Testmustern benötigt. Zur besseren Anschauung werden die Generalisierungseigenschaften des Netzes an der folgenden analytischen Funktion demonstriert:

$$f(x_1, x_2) = 3(1-x_1)^2 \cdot e^{-x_1^2-(x_2+1)^2} - 10(\frac{1}{5}x_1 - x_1^3 - x_2^5) \cdot e^{-x_1^2-x_2^2} - \frac{1}{3}e^{-(x_1+1)^2-x_2^2} \ . \quad (5.1)$$

Diese Funktion spannt für $x_1 \in [-2, 2]$ und $x_2 \in [-2.5, 2.5]$ eine Fläche mit drei Maxima und drei Minima auf, s. Bild 5.4.

Zum Trainieren des Netzes wurde ein Lernmustersatz von 200 zufällig ausgewählten Funktionswerten verwendet. Eine 2-14-7-1 Topologie des Netzes erwies sich mit den in Tab. 5.3 angegebenen Lernparametern für ein schnelles Konvergieren des Netzfehlers als optimal. Die Trainingsdauer bis zur Konvergenz betrug dabei 12 Minuten, in denen rund 380 000 Lernzyklen durchlaufen wurden. Der anteilbezogene Approximationsfehler des Netzes bezüglich der Lernmuster ist in Bild 5.5 dargestellt.

Tabelle 5.3: Lernparameter für das Training der analytischen Funktion

η	α	d	E_{min}	Modus
0.3	0.9	0.0	10^{-4}	Online

Man erkennt, dass die betragsmäßige Abweichung bei mehr als 90% der approximierten Lernmuster unter $\Delta y = 3 \cdot 10^{-2}$ liegt. Der maximale Approximationsfehler von $\Delta y = 6.82 \cdot 10^{-2}$ beträgt dabei weniger als 0.5% des Wertebereichs $[-6.5, 8.1]$ der analytischen Testfunktion in den betrachteten Intervallgrenzen.

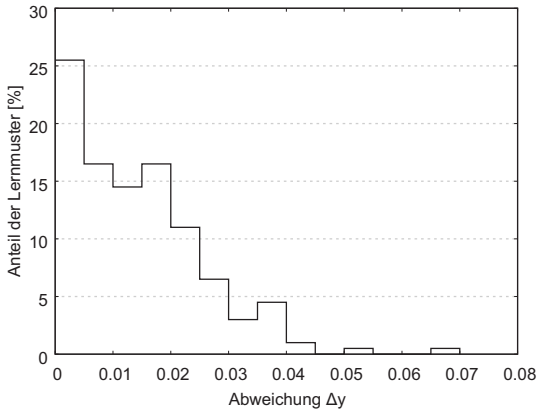


Bild 5.5: Anteilbezogener Approximationsfehler des neuronalen Netzes bezüglich der Lernmuster

In einem zweiten Schritt ist eine Validierungsmenge von 50×50 gleichmäßig verteilten Testmustern erstellt worden. Bild 5.6 zeigt die dreidimensionale Darstellung

der Netzwerkausgabe für die Testmuster (Gitternetz) sowie die Lage der trainierten Lernmuster (Punkte). Im Vergleich zu Bild 5.4 sieht man deutlich, dass das Netz alle drei Minima und Maxima der Funktion genau abbildet.

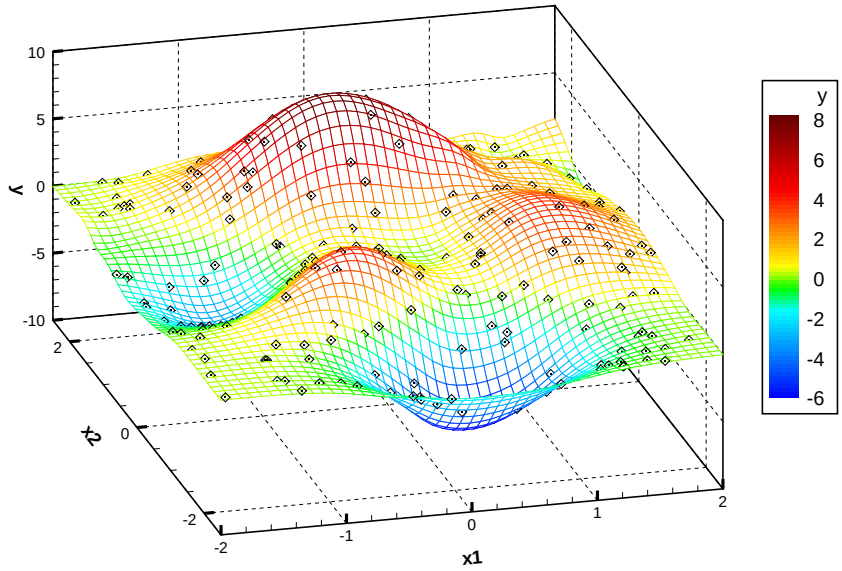
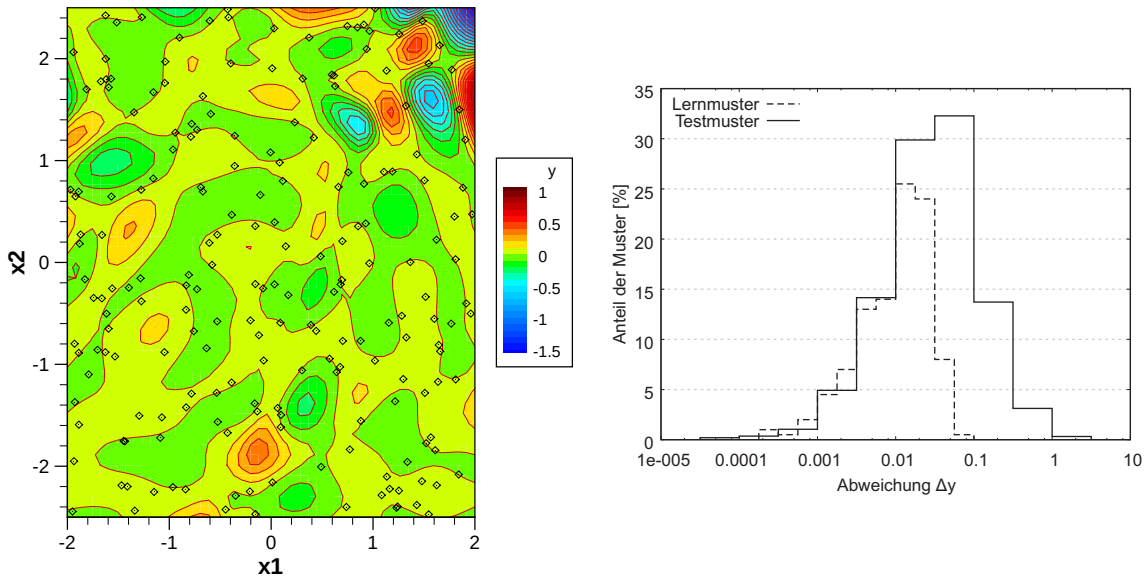


Bild 5.6: Abbildung der analytischen Testfunktion durch das trainierte 2-14-7-1 Netz, wobei die schwarzen Punkte die trainierten Lernmuster darstellen

Eine genauere Beurteilung der Unterschiede ergibt sich durch die Berechnung des Approximationsfehler des Netzes bezüglich der Testmuster, s. Bild 5.7. Damit der Approximationsfehler besser bewertet werden kann, ist im Gegensatz zu Bild 5.5 eine logarithmische Darstellung der Abweichung Δy im Histogramm gewählt worden.

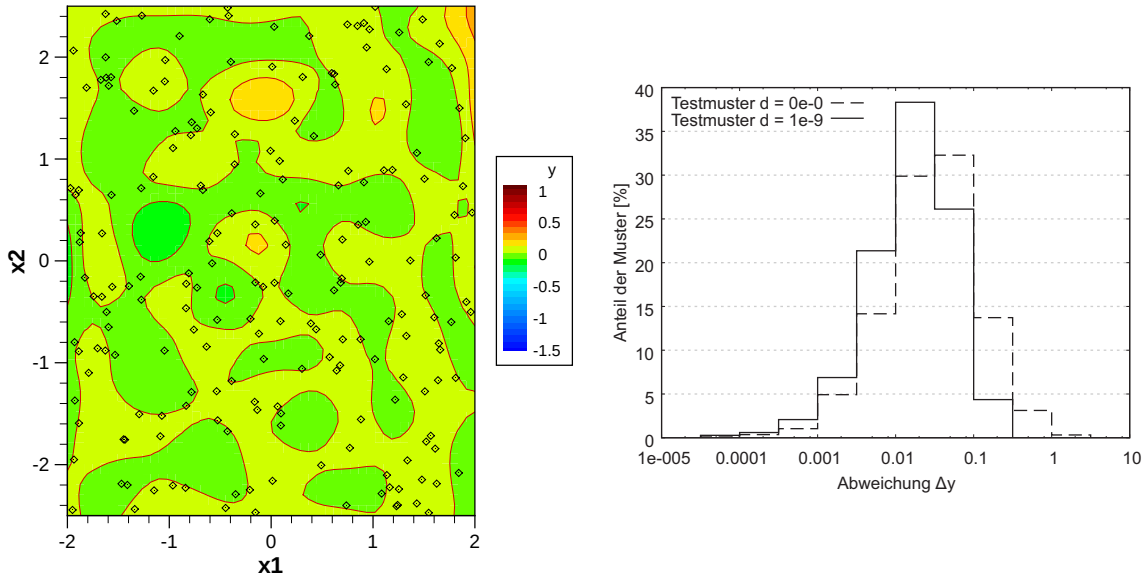
Wie nicht anders zu erwarten, ist der Approximationsfehler der Testmuster größer als bei den Lernmustern. Dennoch ist die Abweichung zu den analytischen Werten für mehr als 80% der Testmuster kleiner als $\Delta y = 0.1$ bzw. 0.7% des Wertebereichs der Funktion. Das neuronale Netz weist somit für ein Großteil der in Bild 5.7a dargestellten Fläche sehr gute Generalisierungseigenschaften auf. Lediglich in der rechten oberen Ecke treten verstärkt Schwankungen des Approximationsfehlers auf, die zum Rand hin stärker werden und im Extrapolationsbereich des Netzes bei $(x_1, x_2) = (2.0, 2.5)$ betragsmäßig ihr Maximum von $\Delta y = 1.53$ erreichen, was einer prozentualen Abweichung von fast 10.5% entspricht. Die vergleichsweise hohen Gradienten der Fehlerfläche an dieser Stelle weisen auf starke Verbindungsgewichte zwischen den für diesen Bereich zuständigen Neuronen hin. Abhilfe verspricht die Erhöhung des Weight Decay Lernparameters d, der das Netz zu betragsmäßig kleineren Verbindungsgewichten zwingt und somit die Abbildungsfunktion geglättet wird, vgl. Kap. 3.5.6. In Bild 5.8 ist die Abweichung zu den Testmustern eines neu trainierten Netzes mit $d = 10^{-9}$ dargestellt.

Im Vergleich zur ebenfalls im Histogramm dargestellten vorherigen Variante mit $d = 0$ werden die Testmuster deutlich besser approximiert. Eine Abweichung unter $\Delta y = 0.1$ erreichen in diesem Fall mehr als 95% der Testmuster. Die betragsmäßig höchste Abweichung liegt ebenfalls im Extrapolationsbereich des Netzes bei



(a) (b)

Bild 5.7: Abweichung der Netzwerkausgabe zu den Testmustern: (a) Absolute Abweichung im Höhenliniendiagramm, (b) anteilbezogene Abweichung der Testmuster im Vergleich zu den Lernmustern



(a) (b)

Bild 5.8: Abweichung der Netzwerkausgabe zu den Testmustern nach der Erhöhung des Weight Decay Parameters auf $d = 10^{-9}$: (a) Absolute Abweichung im Höhenliniendiagramm, (b) Vergleich der anteilbezogenen Abweichung der Testmuster mit und ohne Weight Decay

$\Delta y = 2.45 \cdot 10^{-1}$ und somit unter 1.7% des Wertebereichs der Funktion. Die Trainingsdauer lag allerdings mit 14.5 Minuten und rund 445 000 Lernzyklen etwas höher als beim vorherigen Testfall.

Bei Werten von $d > 10^{-9}$ werden die Verbindungsgewichte des Netzes derart stark beeinflusst, dass der festgelegte Netzfehler $E_{min} = 10^{-4}$ im Zusammenhang mit den eingestellten Lernparametern sowie der verwendeten Topologie während des Trainings nicht mehr erreicht wird.

Kapitel 6

Numerische Strömungssimulation

Zur einfacheren Klassifizierung der im KNN-Designsystem verarbeiteten Strömungsmaschinengeometrien werden für die weiteren Betrachtungen folgende Geometrietypen definiert:

- **Originalgeometrie**: hierbei handelt es sich um die unveränderte Geometrie, die als Eingabe im KNN-Designsystem aufbereitet und parametrisiert werden soll.
- **Lerngeometrie**: diese Geometrie ist die ausgehend von der Originalgeometrie vom KNN-Designsystem aufbereitete, parametrisierte und interaktiv systematisierte Geometrie, die in Form eines Lernmusters dem neuronalen Netz zum Trainieren angeboten wird.
- **Approximierte Geometrie**: diese Geometrie entsteht durch das Abrufen des neuronalen Netzes eines bereits in der Lerndatenbank vorhandenen Eingabemusters der jeweiligen Lerngeometrie. Durch einen Vergleich der approximierten Geometrie mit der Lerngeometrie wird die Approximationsgenauigkeit des neuronalen Netzes beurteilt.
- **Interpolierte Geometrie**: hierbei handelt es sich ebenfalls um eine aus dem neuronalen Netz abgerufenen Geometrie, deren Eingabemuster allerdings nicht in der Lerndatenbank vorhanden ist.

Die Bestimmung der Approximationsgenauigkeit des neuronalen Netzes durch einen Vergleich der approximierten Geometrie mit der zugehörigen Lerngeometrie ist zwar durchaus sinnvoll, allerdings werden dadurch lediglich Aussagen über die Qualität der Geometrieparameter gewonnen und nicht über die Strömungseigenschaften der zugrunde liegenden Maschine. Außerdem ergeben sich trotz guter Approximationseigenschaften des neuronalen Netzes bereits bei der Datenaufbereitung, der Parametrisierung und der anschließenden Systematisierung Unterschiede zwischen den Lerngeometrien und den für einen Vergleich besser geeigneten Originalgeometrien. Aus diesem Grunde werden lediglich die approximierten Geometrien einer Strömungssimulation mit anschließender Auswertung unterzogen. Sowohl die Geometrie als auch

die Strömungsergebnisse können dann mit dem jeweiligen Original qualitativ und quantitativ verglichen werden.

Bei der Beurteilung der Generalisierungseigenschaften erweist sich die in Kap. 5.2 beschriebene Methode ebenfalls als ungeeignet, zumal die Anzahl der zur Verfügung stehenden Lerngeometrien zu gering und somit eine Aufteilung in Lern- und Testmuster ungeeignet ist. Zudem lässt sich die Generalisierungsfähigkeit des Netzes auch hierbei besser durch eine Strömungssimulation der interpolierten Geometrien beurteilen.

Die Berechnung der Strömung erfolgt für den Auslegungspunkt des jeweiligen Laufrades unter Verwendung des am Lehrstuhl entwickelten 3D Navier-Stokes-Codes, s. Skoda [54]. Als Ausgangsbasis der Untersuchungen wird das Standard k-ϵ Turbulenzmodell mit Wandfunktionen angewendet.

Oberstes Ziel der numerischen Berechnung ist die Vorhersage der integralen Energieumsetzung, wie z.B. die Totaldruckänderung und der Wirkungsgrad des Laufrades. Meist erhält man hierfür quantitativ richtige Aussagen, zumindest ist jedoch eine zuverlässige relative Vorhersage im Sinne einer Verbesserung oder Verschlechterung der Energieumsetzung der approximierten Geometrie im Vergleich zur Originalgeometrie möglich.

Ein weiteres wichtiges Ergebnis ist die Berechnung umfangsgemittelter Stromfeldgrößen als Funktion der Schaufelhöhe. Diese Aussagen sind für die Ausrichtung der Schaufel in die Strömung und für die Beurteilung, wie gleichmäßig die Energiewandlung in den einzelnen Flutbahnschnitten der Beschaufelung umgesetzt wird, von Bedeutung.

6.1 Berechnung geeigneter Mittelwerte

Um einen qualitativen und quantitativen Eindruck der Strömung durch eine Turbomaschine zu bekommen, wird die dreidimensionale Lösung der Simulationsrechnung durch geeignete Mittelungsvorschriften auf flächenhaft gemittelte bzw. umfangsgemittelte Werte reduziert. Hierbei kommen vor allem das flächengewichtete Mittel und das massenstromgewichtete Mittel in Betracht, die am Beispiel einer Umfangsmittelung kurz vorgestellt werden.

Die Flächenmittelung wird beispielsweise zur Berechnung des gemittelten statischen Drucks $\overline{p}$ und der Meridiangeschwindigkeit $\overline{c}_m$ verwendet:

$$\overline{p} = \frac{1}{\sum_j A_j} \cdot \sum_j p_j \cdot A_j \ , \tag{6.1}$$

$$\overline{c}_m = \frac{1}{\sum_j A_{m,j}} \cdot \sum_j c_{m,j} \cdot A_{m,j} \ , \tag{6.2}$$

wobei die Teilfläche $A_{m,j}$ der projizierte Anteil der Fläche A_j in Richtung $\vec{c}_{m,j}$ ist.

Diese Mittelungsvorschrift ist zum einen konsistent für den Druck, da sie einer Kraftmittelung entspricht, zum anderen erfüllt sie im Falle einer Geschwindigkeit die Massenerhaltung. Allerdings ist die flächengewichtete Mittelung in Umfangsrichtung für die Umfangsgeschwindigkeitskomponente c_u ungeeignet, da der projizierte Flächenanteil stets Null ist.

Unter der Voraussetzung eines inkompressiblen Fluids berechnet sich das massenstromgewichtete Mittel allgemein für eine Strömungsgröße ϕ zu:

$$\overline{\phi} = \frac{1}{\sum_j c_{n,j} \cdot A_j} \sum_j \phi \cdot c_{n,j} \cdot A_j \ , \tag{6.3}$$

wobei $c_{n,j}$ die Geschwindigkeitskomponente senkrecht zur Fläche A_j darstellt. Diese Mittelungsvorschrift ist zwar physikalisch sinnvoll, da sie im Falle einer Geschwindigkeitsmittelung ebenfalls einer Kraftmittelung entspricht und somit eine äquivalente Aussage zum flächengemittelten statischen Druck bildet, sie erfüllt allerdings nicht die Massenerhaltung und eignet sich somit nur bedingt für die Mittelung von Durchflussgeschwindigkeiten. Bei der Berechnung der repräsentativen Umfangsgeschwindigkeit $\overline{c}_u$ wird aufgrund der oben beschriebenen Probleme allerdings immer das massenstromgewichtete Mittel verwendet.

6.2 Integrale Energieumsetzung

Ausgehend von einer vorgegebenen Fall- bzw. Förderhöhe H und einem vorgegebenen Volumenstrom Q, lässt sich der Betriebspunkt einer hydraulischen Strömungsmaschine zusammen mit der Drehzahl n des Laufrades durch die spezifische Drehzahl n_q beschreiben:

$$n_q = n \cdot \frac{Q^{1/2}}{H^{3/4}} \ . \tag{6.4}$$

Aufgrund der besseren Vergleichsmöglichkeiten der Integralwerte von unterschiedlich dimensionierten Strömungsmaschinen mit dem Laufraddurchmesser D, empfiehlt sich die Verwendung dimensionsloser Kenngrößen zur Beschreibung des Betriebspunktes:

$$n_q = 157.79 \cdot \frac{\varphi^{1/2}}{\Psi_t^{3/4}} \ . \tag{6.5}$$

Die Volumenzahl φ und die Druckzahl Ψ_t sind dabei wie folgt definiert:

$$\varphi = \frac{8Q}{\pi\omega D^3} \ , \tag{6.6}$$

$$\Psi_t = \frac{2gH}{u_{ref}^2} \qquad \text{mit} \qquad u_{ref} = \frac{\omega D}{2} \ . \tag{6.7}$$

In der Regel wird der Volumenstrom Q für die numerische Strömungssimulation als Randbedingung vorgegeben. Die Druckzahl Ψ_t wird anschließend aus der Strömungslösung als Differenz des mittleren Totaldrucks $\overline{p}_t$ zweier Bilanzierungsebenen vor dem ersten und nach dem letzten Stufenelement berechnet. Am Beispiel einer Turbinenstufe, s. Bild 6.1, ergibt sich die Druckzahl Ψ_t somit zu:

$$\Psi_t = \frac{\overline{p}_{t,La_3} - \overline{p}_{t,Le_0}}{p_{ref}} \qquad \text{mit} \qquad p_{ref} = \frac{\rho u_{ref}^2}{2} \; . \tag{6.8}$$

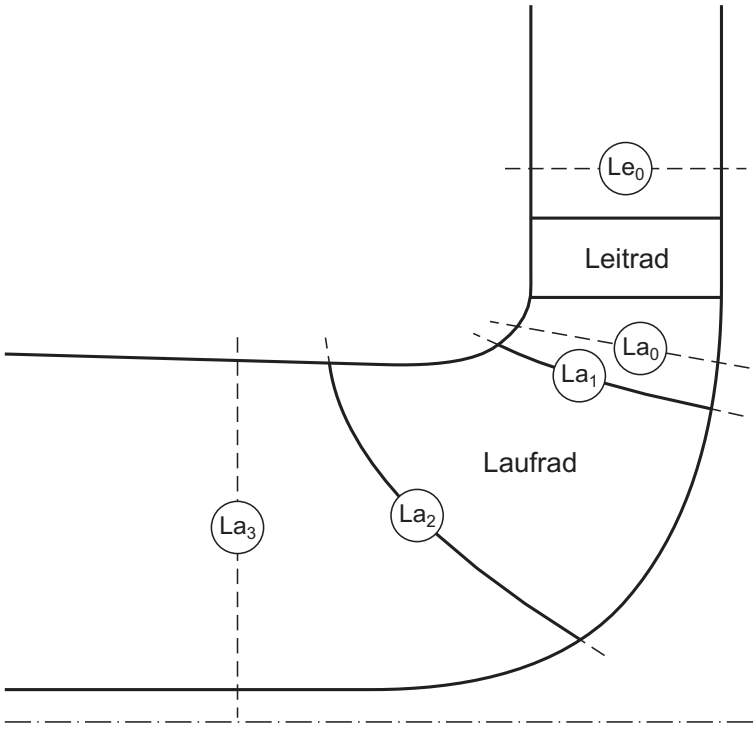


Bild 6.1: Lage der Bilanzierungsebenen zur Berechnung der integralen Werte Ψ_t, $\Psi_{t,La}$ und $\Psi_{t,th}$ sowie der umfangsgemittelten Geschwindigkeit $\overline{c}_m$ und $\overline{c}_u$ entlang der Schaufelein- und -austrittskante des Laufrades am Beispiel einer Francis Turbinenstufe

Die theoretische Druckzahl $\Psi_{t,th}$ ergibt sich aus der Drehimpulsänderung im Laufrad zwischen Bilanzierungsebene La_0 und La_3:

$$\Psi_{t,th} = \frac{8(\overline{rc}_{u,La_3} - \overline{rc}_{u,La_0})}{\omega D^2} \; . \tag{6.9}$$

Aus den beiden Integralwerten lässt sich der hydraulische Wirkungsgrad η_h der Turbinenstufe als Verhältnis von Nutzen durch Aufwand wie folgt berechnen:

$$\eta_{h,Tu} = \frac{\Psi_{t,th}}{\Psi_t} \; . \tag{6.10}$$

Zur Bestimmung des Laufradwirkungsgrades $\eta_{h,La}$ wird die Druckzahl Ψ_t der Stufe in Gl. 6.10 durch die Druckzahl $\Psi_{t,La}$ ersetzt, die sich aus der Totaldruckbilanz für das Laufrad zwischen Ebene La_0 und La_3 ergibt:

$$\Psi_{t,La} = \frac{\overline{p}_{t,La_3} - \overline{p}_{t,La_0}}{p_{ref}} \ . \tag{6.11}$$

Der hydraulische Wirkungsgrad einer Kreiselpumpe verhält sich reziprok zum Turbinenwirkungsgrad und ist wie folgt definiert:

$$\eta_{h,Pu} = \frac{\Psi_t}{\Psi_{t,th}} \ . \tag{6.12}$$

Kapitel 7

Ergebnisse

Das in dieser Arbeit beschriebene KNN-Designsystem soll zur Generierung einer Laufradbaureihe für Francis Turbinen einerseits und Radialpumpen andererseits angewendet werden. Ausgangsbasis hierfür ist jeweils ein Satz vorhandener Strömungsmaschinengeometrien, die im KNN-Designsystem aufbereitet und trainiert worden sind. Dieses Kapitel befasst sich mit der Beurteilung der approximierten und der interpolierten Geometrien der trainierten neuronalen Netze sowie mit den Ergebnissen der durchgeführten Strömungsberechnungen. Die Strömungsergebnisse der Originalgeometrien werden hierfür als Vergleich hinzugezogen.

7.1 Francis Turbinen

In Abhängigkeit von der spezifischen Drehzahl n_q hat Lepach [26] in seiner Arbeit sechs Stufen von Francis Turbinen entworfen und optimiert. Dabei beschränkte sich die Optimierung auf die Laufradgeometrie. Die Leitradgeometrie wurde stets auf die gleiche Weise generiert und lediglich zum Erzeugen des am Laufradeintritt erforderlichen Drehimpulses entsprechend eingestellt. Dadurch konnte die Aufbereitung und Parametrisierung der Geometrien im KNN-Designsystem auf die Meridiankontur sowie die Laufschaufel beschränkt werden.

Bei den entworfenen Turbinen handelt es sich um eine Modellbaureihe, für die die in Tab. 7.1 dargestellten Laufraddaten identisch sind.

Tabelle 7.1: Identische Laufraddaten der einzelnen Turbinen der Modellbaureihe

Drehzahl	n	$[\frac{1}{\text{min}}]$	1000
Außendurchmesser	D_{1a}	[mm]	340
Profildicke Hinterkante	d_{HK}	[mm]	2
maximale Profildicke	d_{max}	[mm]	7
relative Dickenrücklage	$x_{d_{max}}/l$	[%]	20

Eine weitere Reduzierung der Anzahl der zu lernenden Geometrieparameter ergibt sich hierbei ebenfalls durch die konstant über die spezifische Drehzahl gehaltenen Profilierungsparameter d_{HK}, d_{max} und $x_{d_{max}}/l$ der Laufschaufel.

Tab. 7.2 fasst die Auslegungsdaten der untersuchten Francis Turbinen zusammen. Die Volumenzahl φ und die Druckzahl Ψ_t sind in Abhängigkeit von der spezifischen Drehzahl n_q in Bild 7.1 qualitativ dargestellt, s. auch LEPACH [26].

Tabelle 7.2: Auslegungsdaten der zu trainierenden Francis Turbinen

Spez. Drehzahl	n_q	$[\frac{1}{\text{min}}]$	20	40	60	80	100	120
Volumenzahl	φ	[-]	0.040	0.132	0.233	0.312	0.343	0.312
Druckzahl	Ψ_t	[-]	1.850	1.6125	1.375	1.1375	0.900	0.6625
Volumenstrom	Q	$[\frac{\text{m}^3}{\text{s}}]$	0.0653	0.2127	0.3768	0.5041	0.5543	0.5041
Fallhöhe	H	[m]	29.883	26.047	22.211	18.374	14.538	10.701
Schaufelzahl La.	z_{La}	[-]	17	15	15	13	11	9
Schaufelzahl Le.	z_{Le}	[-]	24	20	16	16	16	16

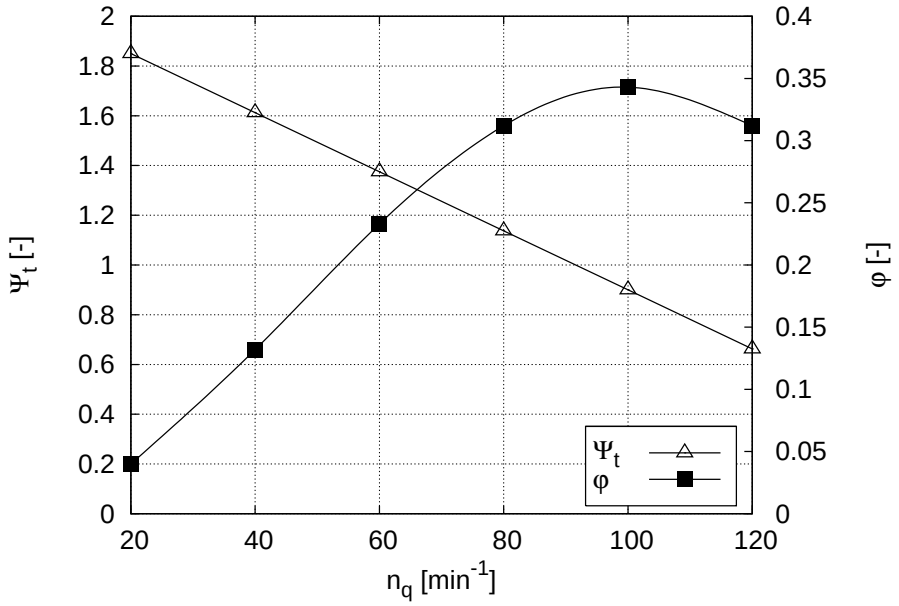


Bild 7.1: Qualitativer Verlauf der Volumenzahl φ und der Druckzahl Ψ_t im Auslegungspunkt in Abhängigkeit von der spezifischen Drehzahl n_q der zu lernenden Francis Turbinen

7.1.1 Geometrievergleich Approximation – Original

Die im KNN-Designsystem parametrisierten und systematisierten Geometrien wurden in zwei unterschiedlichen neuronalen Netzen trainiert. In einem Netz wurden

die Deckscheiben- und Nabenkontur sowie die Ein- und Austrittskanten der Laufschaufel trainiert. Tab. 7.3 zeigt die gewählte Netztopologie und die für das Training verwendeten Lernparameter.

Tabelle 7.3: Netztopologie und Lernparameter für das Training der Meridiankonturen der Turbinen

Netztopologie		1-5-25
Lernmodus		Batch
Lernrate	η	0.3
Momentum	α	0.9
Weight Decay	d	0.0
Netzfehler	E_{min}	10^{-8}

Das zweite neuronale Netz, dessen Einstellungen in Tab. 7.4 zusammengefasst sind, wurde zum Trainieren der konformen Abbildung der Laufradbeschaufelung eingesetzt.

Tabelle 7.4: Netztopologie und Lernparameter für das Training der konformen Abbildung der Laufradbeschaufelung

Netztopologie		1-7-45
Lernmodus		Batch
Lernrate	η	0.3
Momentum	α	0.9
Weight Decay	d	$5.0 \cdot 10^{-8}$
Netzfehler	E_{min}	10^{-8}

Um einen Eindruck der Approximationsgenauigkeit der beiden neuronalen Netze zu gewinnen, werden die durch das trainierte neuronale Netz approximierten Geometrien mit den Lerngeometrien verglichen. Bild 7.2 zeigt die maximalen Abweichung der unterschiedlichen Geometrieparameter der beiden Netze. Zur besseren Darstellung wurden die Geometrieparameter entsprechend klassifiziert.

Man erkennt, dass die Abweichungen der R- und Z-Koordinaten, der Streckenverhältnisse μ sowie die konformen Längen ΔL der approximierten Geometrieparameter deutlich kleiner sind als die Winkelgrößen $\Delta\varphi$ bzw. β_S. Grund hierfür ist die Vorgabe eines im Vergleich zu den anderen Geometrieparametern sehr großen Transformationsintervalls für das neuronale Netz, um die in den Lernmustern auftretenden Winkelgrößen abzudecken. Auf dieses Problem wurde bereits in Kap. 4.3 näher eingegangen. Insgesamt jedoch sind die Abweichungen der approximierten Geometrien von den Lerngeometrien vernachlässigbar klein, so dass sich die weiteren Untersuchungen auf einen Vergleich der approximierten Geometrien mit den Originalgeometrien beschränken lassen.

Die größten Geometrieabweichungen entstehen durch die Parametrisierung der Originalgeometrie sowie durch die anschließende manuelle Systematisierung der Geome-

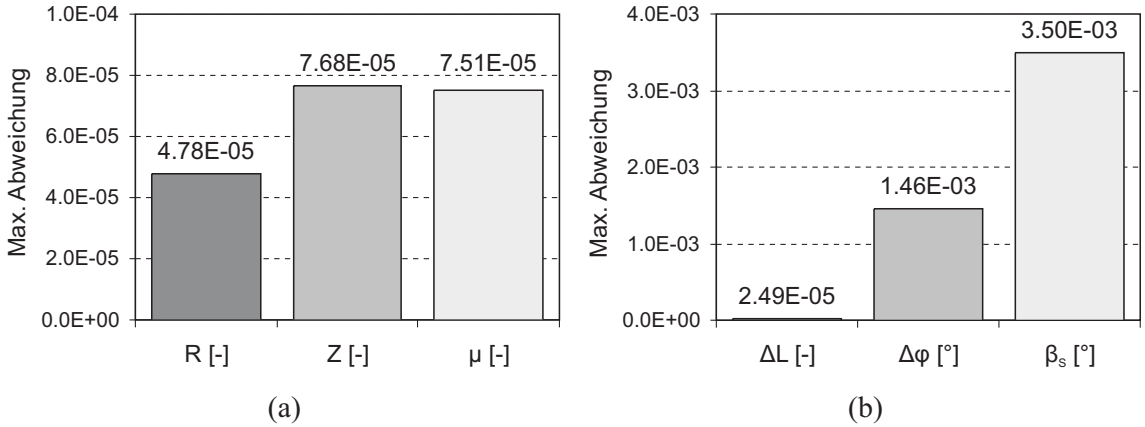


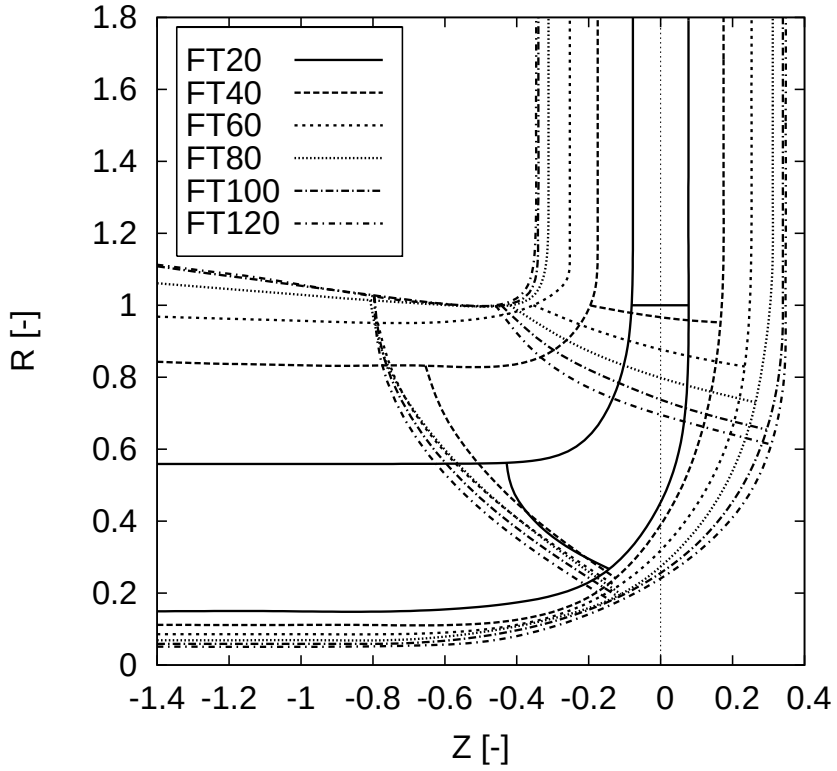
Bild 7.2: Maximale Abweichungen der von den beiden neuronalen Netzen approximierten Geometrieparameter zu den in den Lernmustern enthaltenen Geometrieparametern: (a) Meridiangeometrie, (b) konforme Abbildung

trieparameter. Bild 7.3 zeigt die Meridiankonturen der Originalgeometrien und der approximierten Geometrien im Vergleich. Vor allem am Beispiel der Turbine vom Typ FT60 erkennt man deutlich den Einfluss der Systematisierung auf die approximierte Geometrie. Während die Deckscheibenkontur im Schaufeleintrittsbereich der Originalgeometrie leicht von einer gewollten Systematik abweicht, fällt sie bei der approximierten Geometrie wesentlich besser aus. Ob sich die Systematisierung allerdings auch positiv auf die zu berechnende Strömung auswirkt, ist zu diesem Zeitpunkt noch offen.

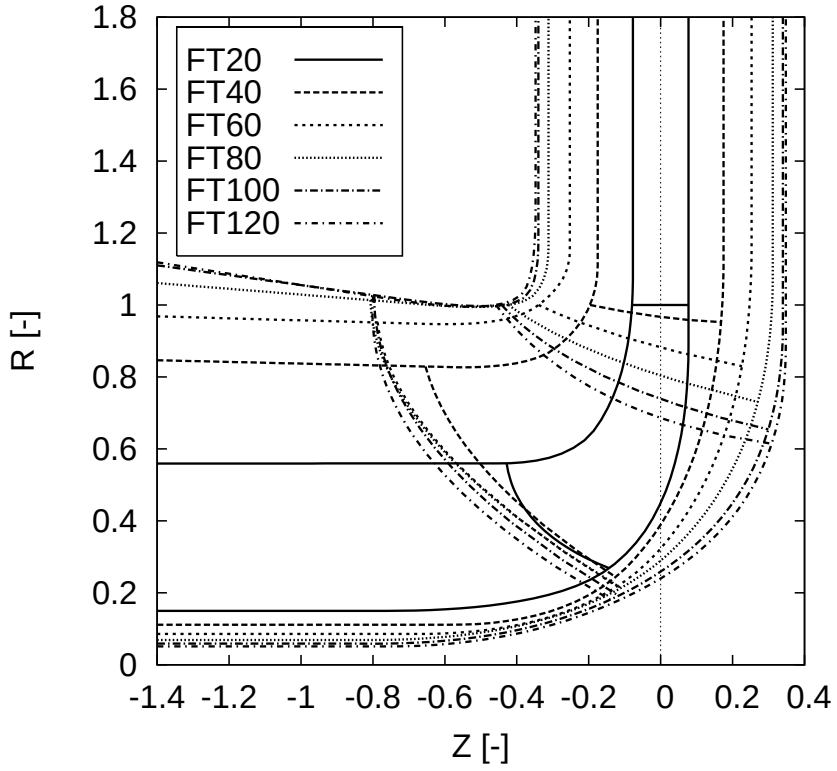
Eine genauere Beurteilung der Unterschiede ergibt sich aus einem Vergleich des meridionalen Flächenverlaufs. Dieser ist für drei repräsentative Turbinen mit den spezifischen Drehzahlen $n_q = 20$ min^{-1}, $n_q = 60$ min^{-1} und $n_q = 100$ min^{-1} in Bild 7.4 dargestellt. Während die Flächenverläufe der approximierten Geometrien im Vergleich zu den Originalgeometrie bei den Turbinen FT20 und FT100 trotz Systematisierung sehr gut übereinstimmen, ergeben sich, wie bereits oben festgestellt, bei der Turbine FT60 im Laufradbereich signifikante Unterschiede.

Neben den Geometrieabweichungen durch die Parametrisierung und Systematisierung entstehen weitere Ungenauigkeiten bei der Erstellung der Schaufelgeometrie. Zum einen muss die Schaufel aus den Ergebnissen der beiden neuronalen Netze zur Approximation der Meridiankontur sowie der konformen Abbildung in eine dreidimensionale Geometrie zurück berechnet werden. Zum anderen ist es erforderlich, die Schaufelgeometrie mit einer höheren Anzahl gleichmäßig über den Querschnitt verteilter Flutbahnen im Real-Time-Designsystem des Lehrstuhls (RTD) neu zu verschneiden, um einerseits eine problemlose Aufbereitung der Schaufelgeometrie für die Strömungsberechnung zu gewährleisten und andererseits die Geometrie sinnvoll mit der Originalgeometrie vergleichen zu können. Bild 7.5 zeigt die konforme Abbildung sowie den Schaufelwinkelverlauf der drei Turbinen im Bereich der Deckscheibe, der Schaufelmitte und der Nabe.

Man erkennt, dass die konforme Abbildung sowie der Schaufelwinkelverlauf der approximierten Geometrien und der Originalgeometrien bei den Turbinen FT20 und

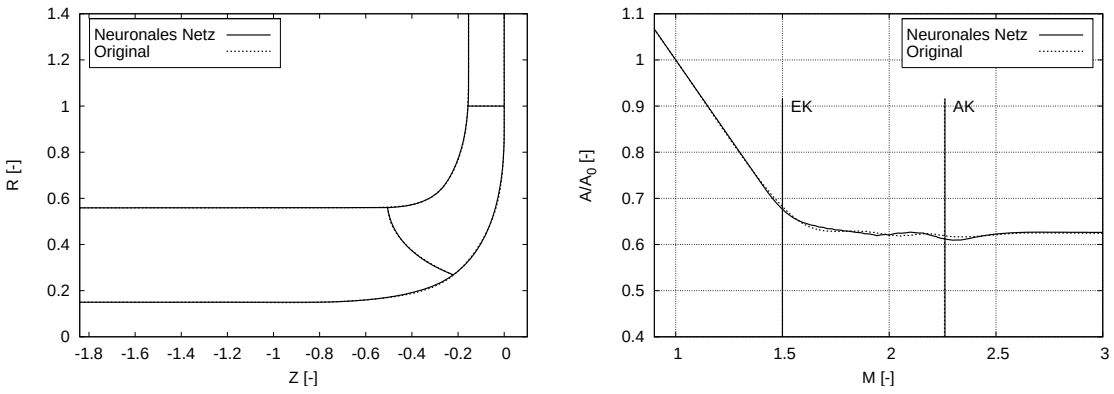


(a)

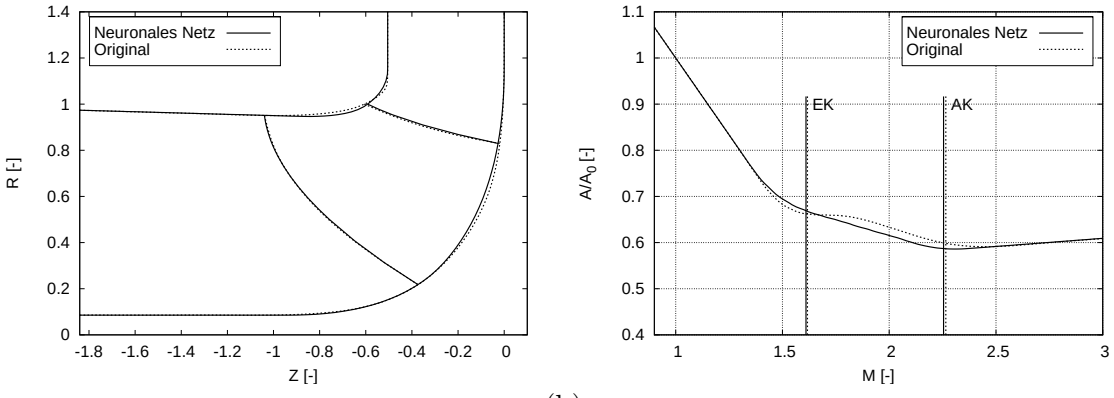


(b)

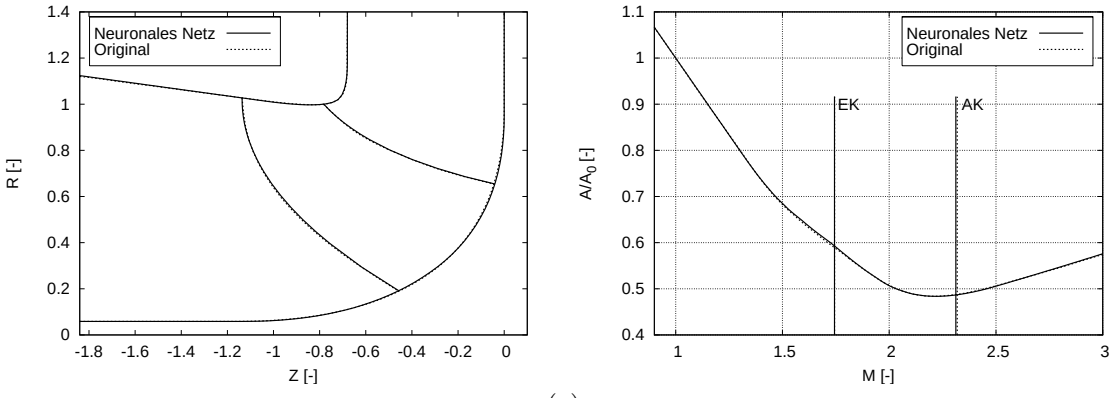
Bild 7.3: Meridiangeometrien der untersuchten Turbinen: (a) Original, (b) Approximation



(a)



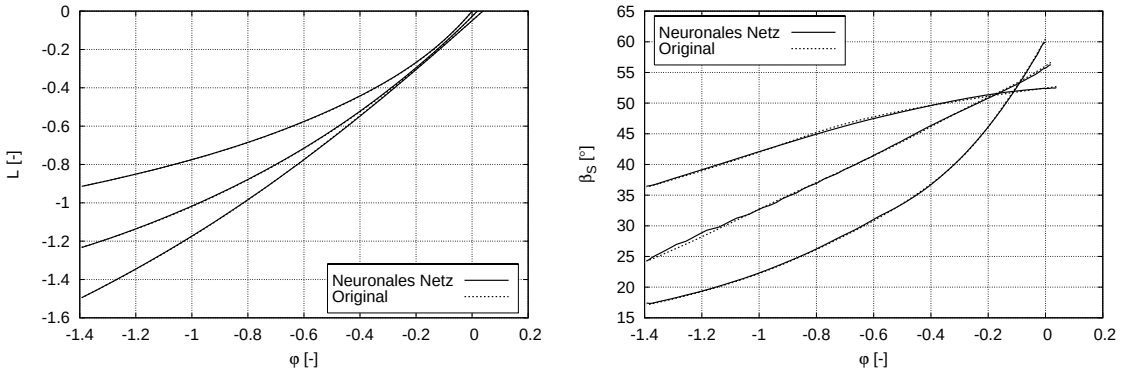
(b)



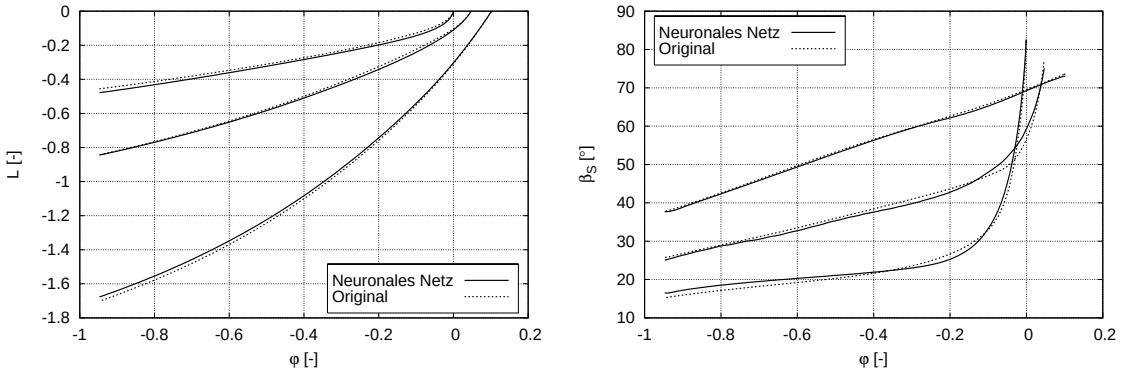
(c)

Bild 7.4: Vergleich der Meridiankontur (links) sowie des meridionalen Flächenverlaufs (rechts) zwischen approximierter Geometrie und Originalgeometrie: (a) FT20, (b) FT60, (c) FT100

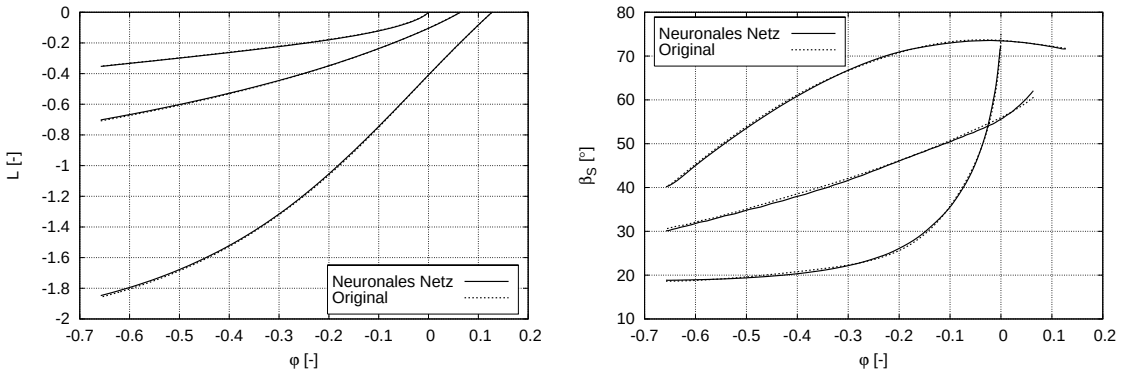
FT100 trotz der zahlreichen Fehlerquellen nahezu deckungsgleich sind. Beeinflusst durch die vergleichsweise große Veränderung der Deckscheibenkontur bei der Systematisierung, ist es bei der Turbine FT60 nicht weiter verwunderlich, dass sich auch die Schaufelwinkelverläufe zur Deckscheibe hin zunehmend verändern.



(a)

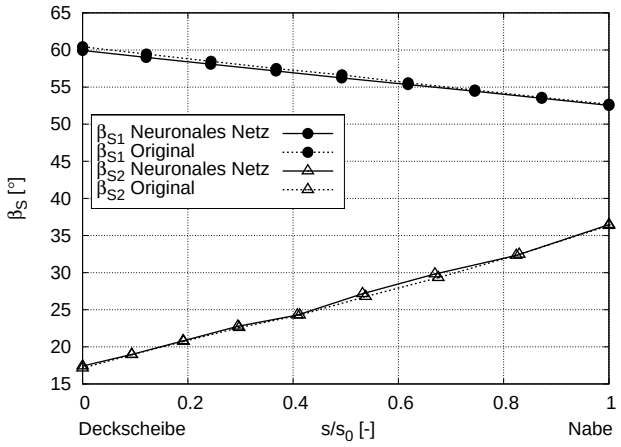


(b)

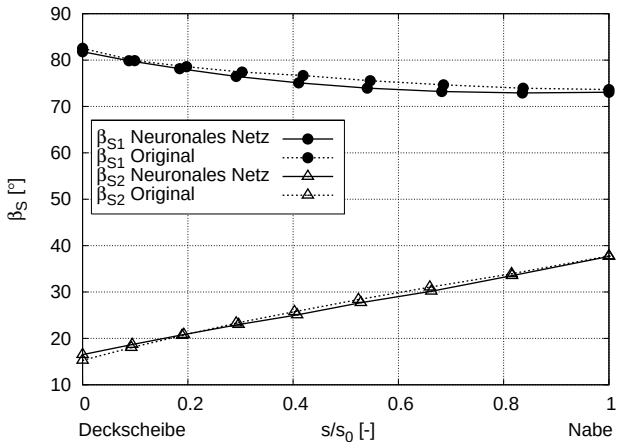


(c)

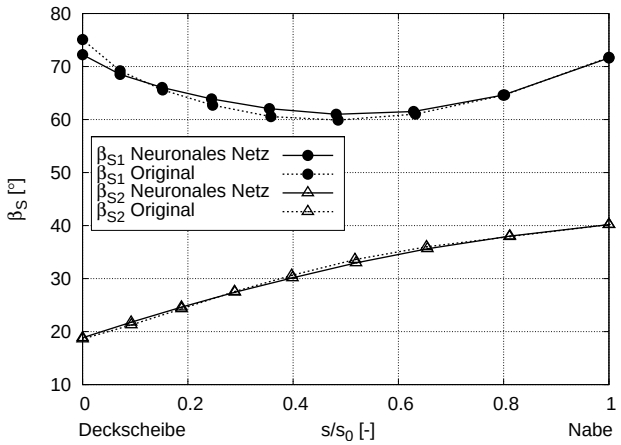
Bild 7.5: Vergleich der konformen Abbildung (links) sowie des Schaufelwinkelverlaufs (rechts) zwischen approximierter Geometrie und Originalgeometrie an Deckscheibe, Schaufelmitte und Nabe: (a) FT20, (b) FT60, (c) FT100



(a)



(b)



(c)

Bild 7.6: Vergleich der Ein- und -austrittswinkelverteilung der Laufradbeschaufelung zwischen approximierter Geometrie und Originalgeometrie: (a) FT20, (b) FT60, (c) FT100

Ein weiteres interessantes Ergebnis betrifft die Ein- und Austrittswinkelverteilung der Laufschaufel, s. Bild 7.6.

Hierbei hat sich eine zunehmende Verschlechterung der Eintrittswinkelverteilung für eine spezifische Drehzahl $n_q > 80$ min^{-1} im Bereich der Deckscheibe ergeben. Dies lässt sich dadurch begründen, dass der auf der Deckscheibenkontur liegende Eintrittskantenpunkt (R_{1a}, Z_{1a}) durch $R_{1a} = 1$ fest vorgegeben ist. Der zugehörige Wert Z_{1a} ergibt sich dann aus dem Schnitt der Geraden $R = 1 = \text{const.}$ mit der Deckscheibenkontur. Da die Tangente der Deckscheibenkontur bei $R = 1$ für größer werdende spezifische Drehzahlen ebenfalls gegen $R = \text{const.}$ geht, bedeutet bereits eine kleine Abweichung des Verlaufs der Deckscheibenkontur vom Original eine große Verschiebung der Eintrittskante in Z-Richtung, was sich erheblich auf den Schaufelwinkelverlauf auswirkt.

7.1.2 Vergleich der Strömungsergebnisse Approximation – Original

Für die Beurteilung der strömungsmechanischen Eigenschaften wird die Strömung durch die approximierten Turbinen im jeweiligen Auslegungspunkt berechnet und mit den Strömungsergebnissen der Originalgeometrien aus der Arbeit von Lepach [26] verglichen. Damit die Berechnungsergebnisse sinnvoll miteinander verglichen werden können, ist es wichtig, die gleichen Randbedingungen wie in der Simulation der Originalgeometrien zu verwenden. Die Leitradgeometrie wurde dabei unverändert übernommen, ebenso wie die Kontrollraumgrenzen, die Randbedingungen am Ein- bzw. Austritt, die Einstellungen für die Netzgenerierung und den Strömungslöser sowie die Lage der Bilanzierungsebenen.

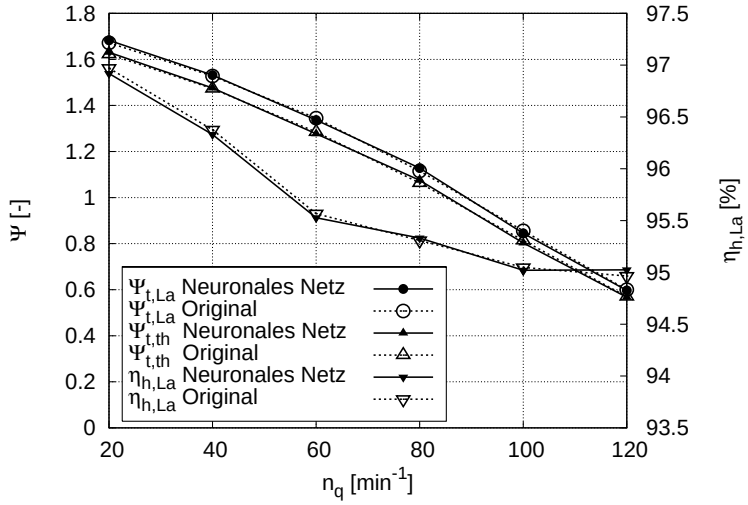
Die Rechnungen wurden auf Netzen mit insgesamt 75 600 Punkten durchgeführt. Für die Turbulenzmodellierung wurde das Standard k-ϵ Modell mit Wandfunktionen verwendet. Zur Diskretisierung der konvektiven Flüsse wurde ein hochauflösendes Verfahren 2. Ordnung eingesetzt, bei dem es sich um das Diskretisierungsschema OSHER handelt, s. Skoda [54]. In Tab. 7.5 sind die Netzdimensionen sowie die wesentlichen Einstellparameter des 3D Navier-Stokes-Codes zur Berechnung der Strömung zusammengefasst.

Tabelle 7.5: Netzdimensionen und Einstellparameter des 3D Navier-Stokes-Codes je Stufenelement

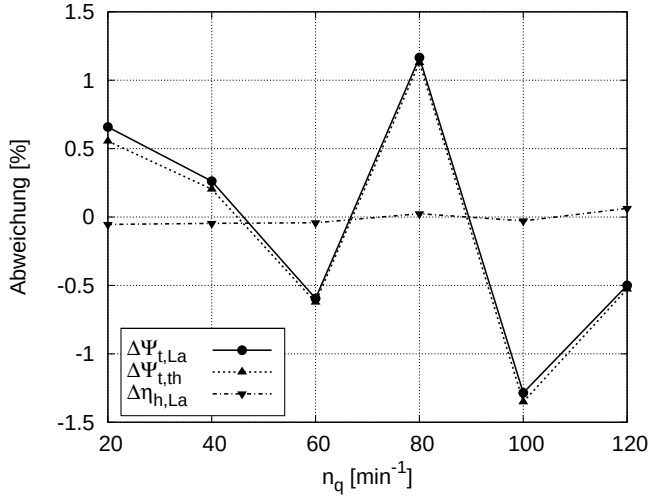
N_i	N_j	N_k	N_{ges}	Diskretisierungsschema	Konvergenzkriterium
75	24	21	37 800	OSHER 2. Ordnung	10^{-5}

Da das Leitrad für alle untersuchten Turbinen vorgegeben und somit nicht Bestandteil der gelernten Geometrien ist, werden bei der Beurteilung der Strömungsergebnisse lediglich die integralen Größen für die Energieumsetzung im Laufrad miteinander verglichen. Bild 7.7a zeigt den Verlauf der Druckzahl $\Psi_{t,La}$, der theoretischen Totaldruckzahl $\Psi_{t,th}$ sowie des hydraulischen Wirkungsgrades des Laufrades $\eta_{h,La}$

der Approximation im Vergleich zum Original in Abhängigkeit von der spezifischen Drehzahl. Die prozentuale Abweichungen dieser Größen wird in Bild 7.7b verdeutlicht.



(a)



(b)

Bild 7.7: Vergleich der Druckzahlen $\Psi_{t,La}$ und $\Psi_{t,th}$ sowie des Wirkungsgrades $\eta_{h,La}$ der berechneten Turbinenstufen zwischen Approximation und Original: (a) integrale Kenngrößen, (b) prozentuale Abweichung der Kenngrößen

Man erkennt, dass die Ergebnisse der Druckzahlen $\Psi_{t,La}$ und $\Psi_{t,th}$ der approximierten Geometrien mit einer betragsmäßigen Abweichung von unter 1.5% sehr gut mit den Ergebnissen der Originalgeometrien übereinstimmen. Bei vier der sechs untersuchten Turbinen ist der Laufradwirkungsgrad $\eta_{h,La}$ geringfügig schlechter ausgefallen als beim Original, allerdings ist die maximale Abweichung von 0.06% des Wirkungsgrades verschwindend klein.

Ein weiteres Ergebnis zur Beurteilung der Abweichungen der in Bild 7.6 gezeigten

Schaufelein- und -austrittswinkelverteilung ist die Darstellung der umfangsgemittelten Meridian- und Umfangsgeschwindigkeiten der Strömung entlang der Schaufelein- und -austrittskante, die für die drei repräsentativen Turbinen FT20, FT60 und FT100 in den Bildern 7.8 und 7.9 aufgetragen sind.

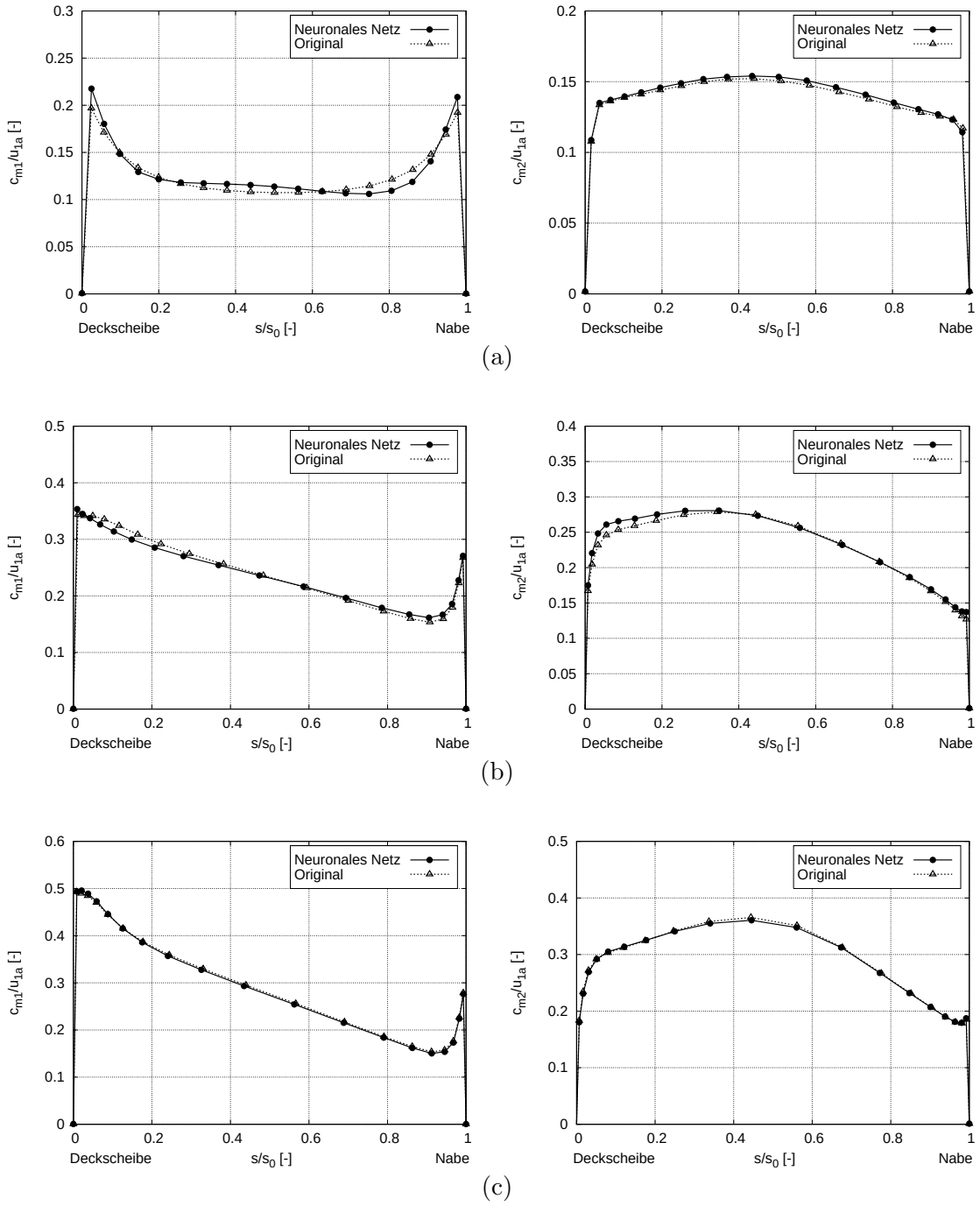


Bild 7.8: Vergleich der gemittelten Meridiangeschwindigkeiten entlang der Schaufelein- (links) und -austrittskante (rechts) zwischen Approximation und Original: (a) FT20, (b) FT60, (c) FT100

Auch hier zeigt sich eine gute Übereinstimmung sowohl bei den Meridian- als auch der Umfangsgeschwindigkeitsverteilungen. Es zeigt sich jedoch, dass der Einfluss der

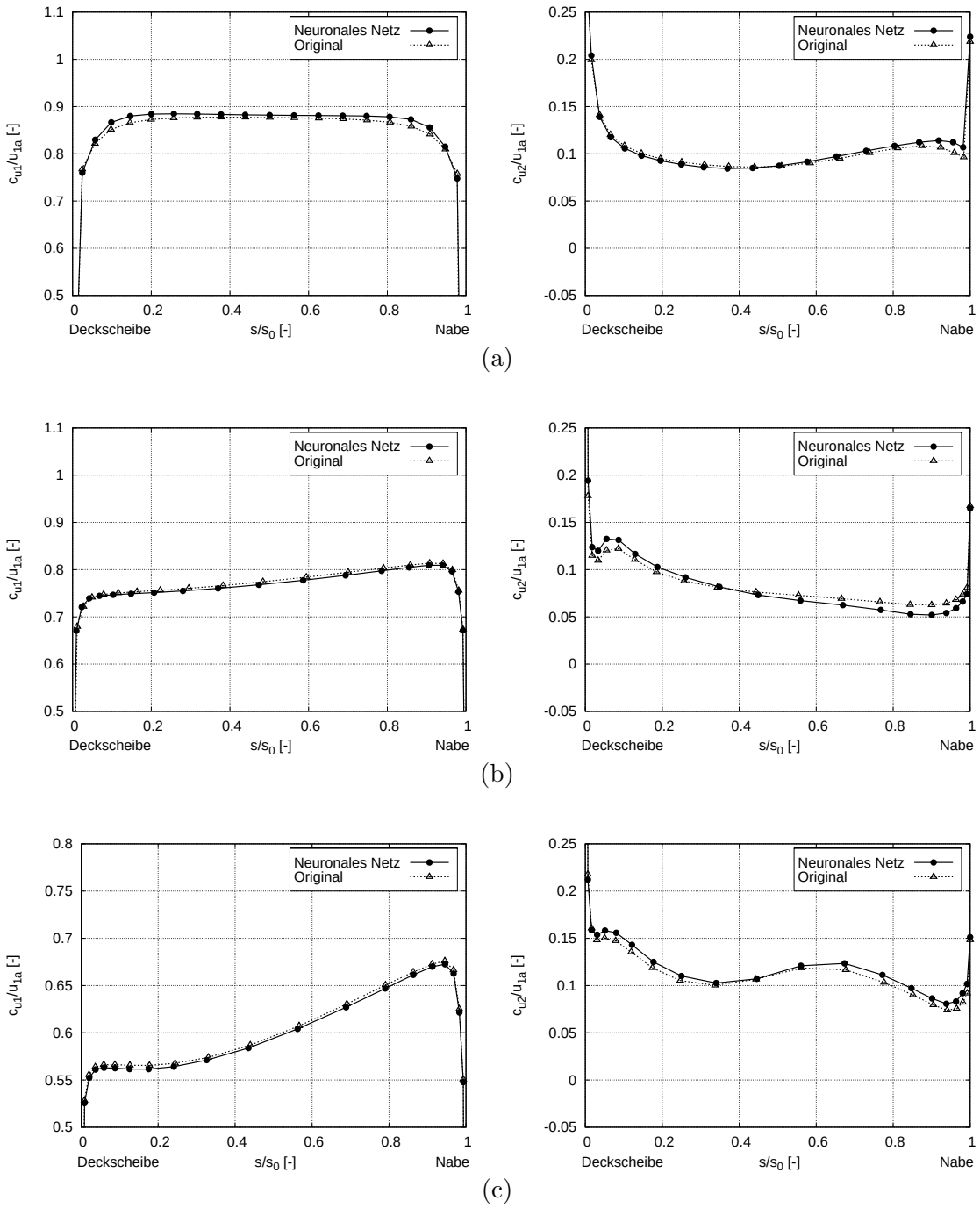
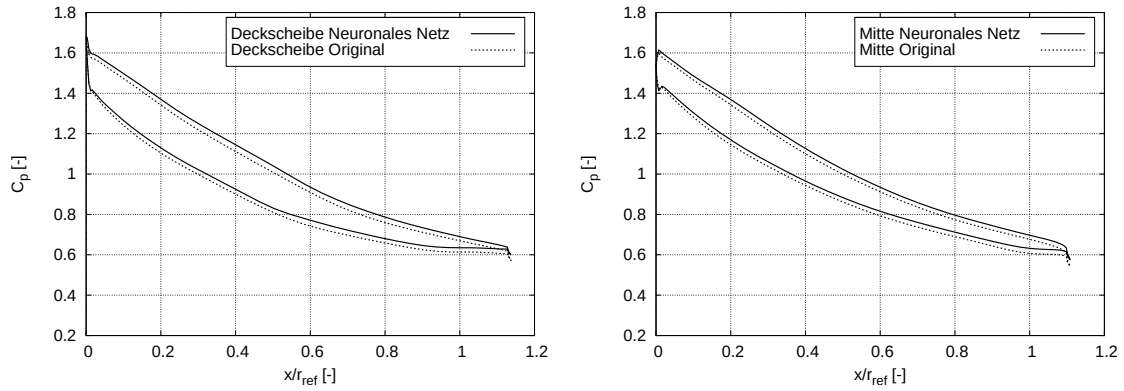


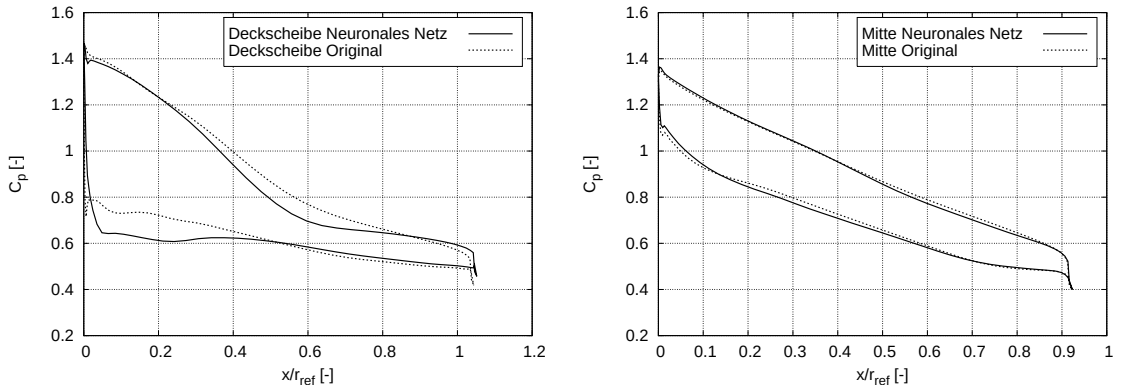
Bild 7.9: Vergleich der gemittelten Umfangsgeschwindigkeiten entlang der Schaufelein- (links) und -austrittskante (rechts) zwischen Approximation und Original: (a) FT20, (b) FT60, (c) FT100

Schaufelwinkelabweichungen auf die Geschwindigkeiten am Eintritt des Laufrades mit kleiner werdender spezifischer Drehzahl deutlich zunimmt. Trotz der kleineren Differenz der Schaufelwinkel, s. Bild 7.6, weichen die Geschwindigkeiten der Turbine FT20 am Eintritt stärker ab als bei den anderen Turbinen. An der Meridiangeschwindigkeitsverteilung der Turbine FT60 erkennt man zudem die Auswirkungen der durch die Systematisierung veränderten Deckscheibenkontur. Die Unterschiede

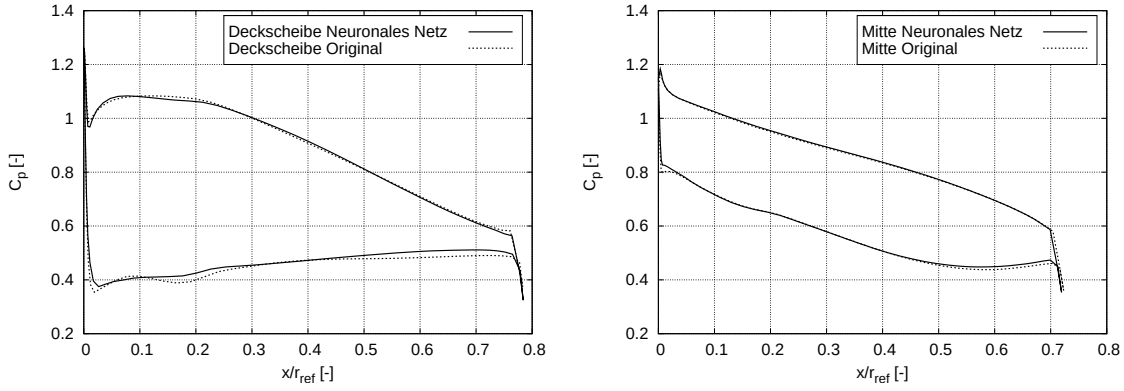
in den Umfangsgeschwindigkeiten am Austritt sind vor allem auf die Abweichungen des Schaufelwinkelverlaufs in Strömungsrichtung sowie der Schaufellänge zurückzuführen, die sich aus der Lage der Ein- und Austrittskanten in der Meridiankontur ergibt.



(a)



(b)



(c)

Bild 7.10: Vergleich der dimensionslosen Druckverteilung auf der Laufschaufel an der Deckscheibe (links) und in der Schaufelmitte (rechts) zwischen Approximation und Original: (a) FT20, (b) FT60, (c) FT100

In Bild 7.10 werden die dimensionslosen, auf das Kavitationsniveau bezogenen Druckverteilungen auf der Laufschaufel entlang der Deckscheibe und in der Schaufel-

mitte verglichen. Die Druckverteilung stellt hierbei ein Maß für die Schaufelbelastung entlang eines gedachten Stromfadens dar. Auf die Darstellung der Druckverteilungen an der Nabe wurde verzichtet, da bei diesen nahezu keine Unterschiede zwischen Approximation und Original zu erkennen sind. Wie nicht anders zu erwarten, weichen auch die Druckverteilungen der Turbinen FT20 und FT100 sowohl an der Deckscheibe als auch in der Schaufelmitte nur unwesentlich von den Druckverteilungen des Originals ab. Die Ausnahme bildet wie bereits erwähnt die Turbine FT60. Während sich die Druckverteilung im mittleren Bereich bei $x/r_{ref} = 0.6$ verschlechtert hat, ist sie im Eintrittsbereich der Schaufel durch die Systematisierung deutlich besser geworden, zumal die Originalschaufel zudem eine leichte Saugspitze aufweist.

7.1.3 Beurteilung interpolierter Geometrien

In diesem Kapitel sollen die Generalisierungseigenschaften der beiden beteiligten neuronalen Netze zur Generierung von Turbinengeometrien beurteilt werden. Hierfür wurden in Abhängigkeit von der spezifischen Drehzahl die in Tab. 7.6 zusammengefassten, nicht in den Lernmustern enthaltenen Geometrien generiert und nachgerechnet.

Tabelle 7.6: Auslegungsdaten der interpolierten Francis Turbinen

Spez. Drehzahl	n_q	$[\frac{1}{\text{min}}]$	30	50	70	90	110
Volumenzahl	φ	[-]	0.082	0.183	0.277	0.335	0.336
Druckzahl	Ψ_t	[-]	1.7312	1.4937	1.2562	1.0187	0.7812
Volumenstrom	Q	$[\frac{\text{m}^3}{\text{s}}]$	0.1331	0.2963	0.4479	0.5407	0.5424
Fallhöhe	H	[m]	27.965	24.129	20.292	16.456	12.620
Schaufelzahl La.	z_{La}	[-]	15	15	13	13	9
Schaufelzahl Le.	z_{Le}	[-]	20	16	16	16	16

Bild 7.11 zeigt die Meridiangeometrien der approximierten und interpolierten Geometrien. Man sieht deutlich, dass die Meridiankonturen gleichmäßig und ohne größere Abweichungen systematisch auseinander hervor gehen.

Wie schon bei den approximierten Geometrien werden die Ergebnisse für drei repräsentative Turbinen vom Typ FT30, FT70 und FT110 genauer analysiert.

In Bild 7.12 ist die konforme Abbildung sowie der Schaufelwinkelverlauf der drei Turbinen abgebildet.

Entscheidend für ein optimales Betriebsverhalten ist die geeignete Wahl der Schaufelwinkel β_{S1} und β_{S2} sowie ein gleichmäßiger Verlauf des Schaufelwinkels über die Schaufellänge. Um bei den interpolierten Geometrien größere Schwankungen in der konformen Abbildung und somit im Schaufelwinkelverlauf zu unterdrücken, wurde beim Trainieren der Schaufelgeometrien auf den Weight Decay Lernparameter d zurückgegriffen, s. Tab. 7.4.

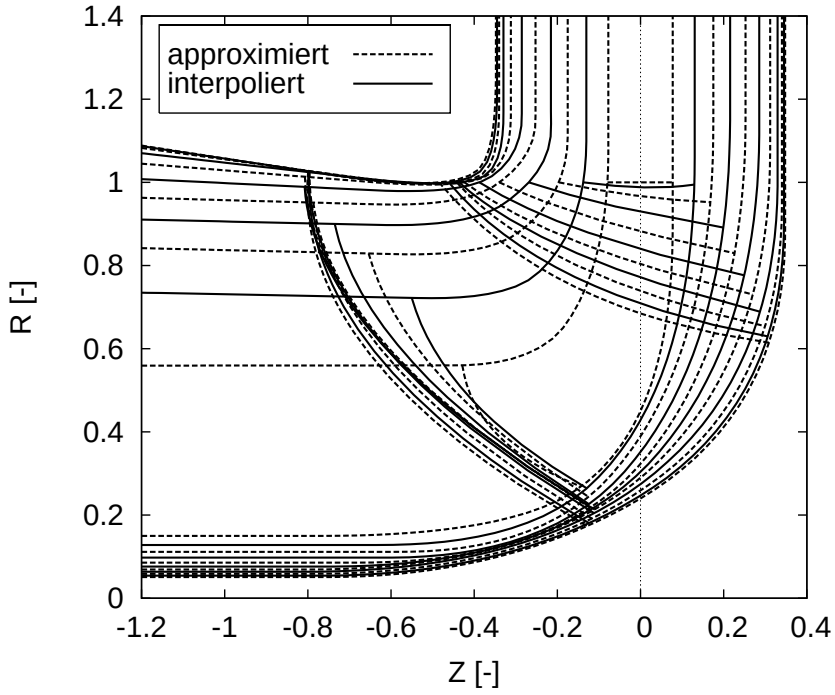


Bild 7.11: Approximierte und interpolierte Meridiangeometrien in Abhängigkeit von der spezifischen Drehzahl im Bereich $n_q = 20$ – 120 min^{-1}

Die integralen Werte der Druckzahl $\Psi_{t,La}$, der theoretischen Druckzahl $\Psi_{t,th}$ und des hydraulischen Wirkungsgrades $\eta_{h,La}$ sind in Bild 7.13 für alle aus dem neuronalen Netz generierten Turbinengeometrien dargestellt.

Über den gesamten spezifischen Drehzahlbereich ist ein gleichmäßiger Verlauf der Druckzahlen zu beobachten. Im Vergleich zum tendenziellen Wirkungsgradverlauf der approximierten Geometrien ist der Wirkungsgrad der interpolierten Turbinen FT30, FT90 und FT110 sogar geringfügig besser.

Die umfangsgemittelten Geschwindigkeitsverteilungen der drei repräsentativen Turbinen entlang der Schaufelein- und -austrittskante sowie die Druckverteilung auf der Schaufel entlang der Deckscheibe, der Schaufelmitte und der Nabe sind in den Bildern 7.14, 7.15 und 7.16 dargestellt.

Vergleicht man die Geschwindigkeitsverteilungen mit den Ergebnissen der approximierten Turbinen FT20, FT60 und FT100, so stellt man ein ähnliches Verhalten der Geschwindigkeitsverläufe fest. Dabei stellt der wellige Verlauf der Umfangsgeschwindigkeit am Austritt der Turbine FT110 ein eher unbefriedigendes Ergebnis dar. Diese Welligkeit ist zwar bei der approximierten Turbine FT100 ebenfalls zu beobachten, allerdings ist sie dort nicht so stark ausgeprägt.

Die Druckverteilungen entlang der Schaufel könnten für die Turbine FT30 kaum besser sein. Bild 7.16a zeigt eindrucksvoll, dass die Schaufel optimal angeströmt wird. Wie beabsichtigt, stellt sich auch eine höhere Schaufelbelastung im vorderen Deckscheibenbereich der Schaufel ein. Ebenfalls zufriedenstellend ist die Druckverteilung der Turbine FT70. Auch hier erkennt man eine ideale Anströmung der Schaufel,

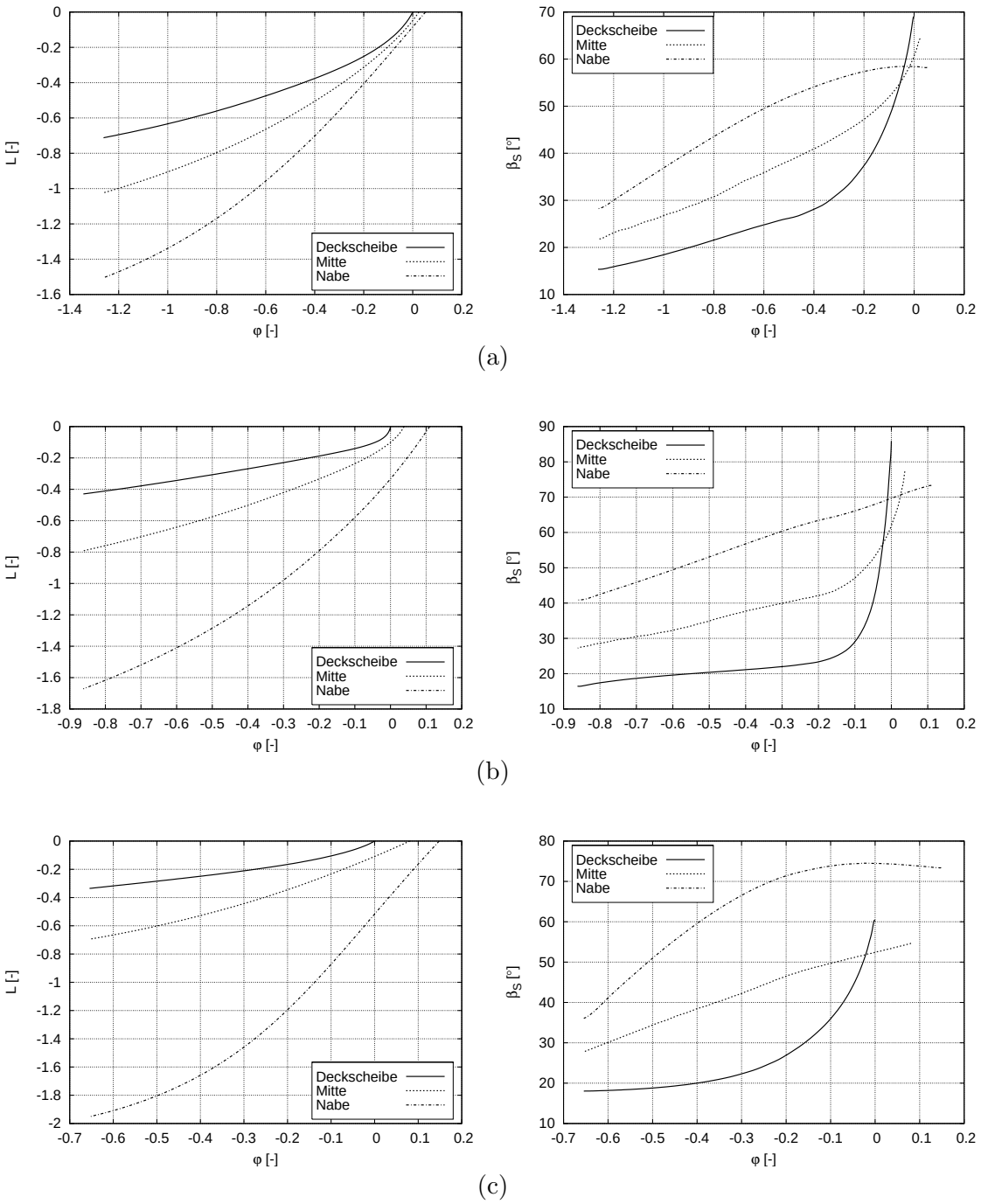


Bild 7.12: Konformen Abbildung (links) sowie Schaufelwinkelverlauf (rechts) der interpolierten Schaufelgeometrien: (a) FT30, (b) FT70, (c) FT110

sowie eine höhere Schaufelbelastung im Eintrittsbereich. Unschön ist allerdings der bauchige Verlauf des Druckes entlang der Deckscheibe auf der Saugseite der Schaufel und die etwas zu geringe Belastung im hinteren Bereich. Der Druckverlauf der Turbine FT110 entlang der Deckscheibenkontur weist eine starke Saugspitze auf, was zum Kavitieren der Strömung in diesem Bereich führen kann. Grund hierfür ist, wie bereits erwähnt, die große Schwankungsbreite bei der Lage der Eintrittskante in Z-Richtung im Bereich der Deckscheibe für große spezifische Drehzahlen, die sich in

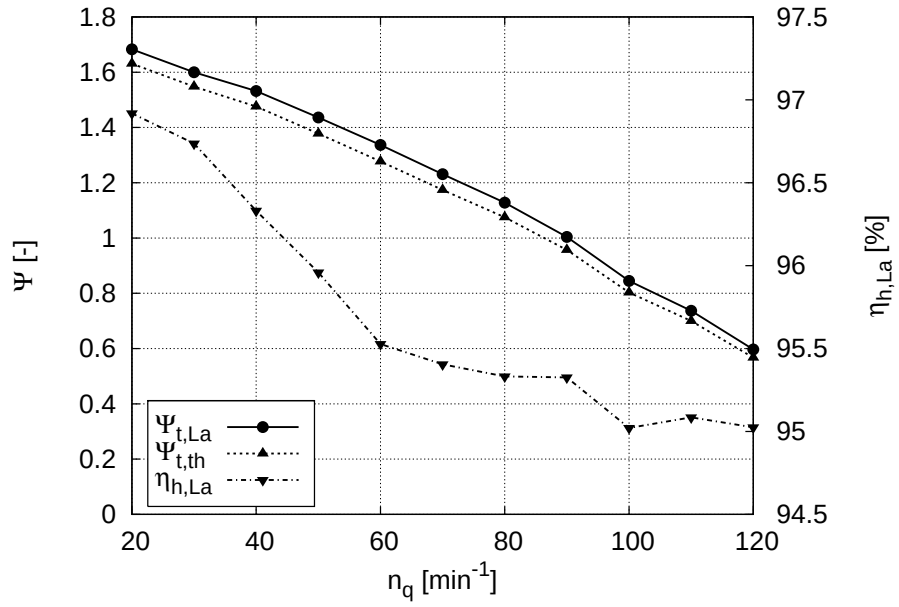


Bild 7.13: Integralwerte der approximierten und interpolierten Turbinen: Totaldruckerhöhung im Laufrad $\Psi_{t,La}$, theoretische Totaldruckerhöhung $\Psi_{t,th}$ und hydraulischer Laufradwirkungsgrad $\eta_{h,La}$

starkem Maße auf den Schaufeleintrittswinkel auswirken kann.

Als Fazit lässt sich allgemein festhalten, dass alle aus dem neuronalen Netz erstellten Turbinen im Mittel qualitativ und quantitativ akzeptable Strömungsergebnisse erzielt haben. Die Verwendung der trainierten neuronalen Netze ermöglicht somit eine deutliche Vereinfachung im Entwurfsprozess neuer Laufräder von Francis Turbinen sowie eine drastische Verkürzung des Zeitaufwandes bei der Optimierung.

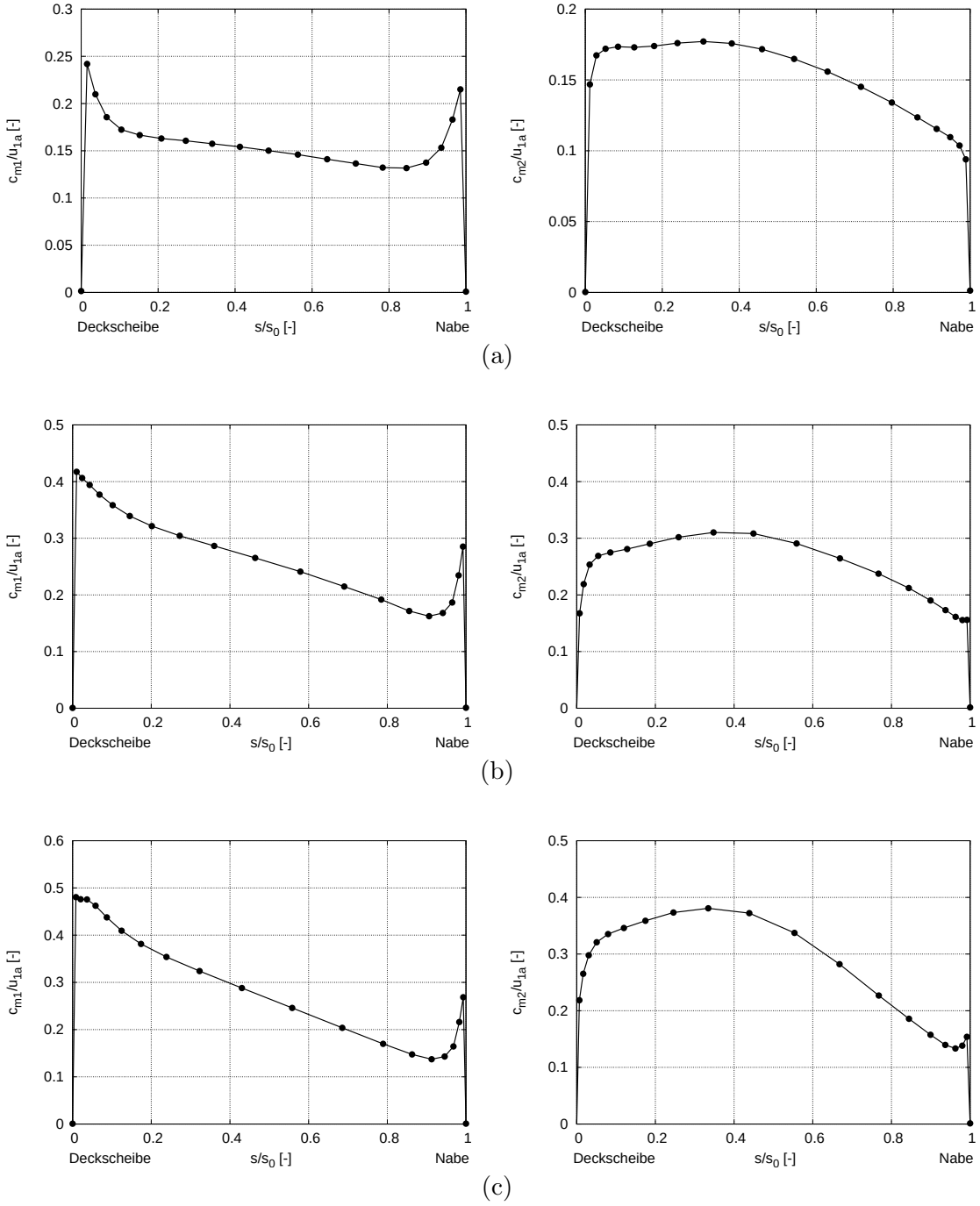


Bild 7.14: Gemittelte Meridiangeschwindigkeiten entlang der Schaufelein- (links) und -austrittskante (rechts): (a) FT30, (b) FT70, (c) FT110

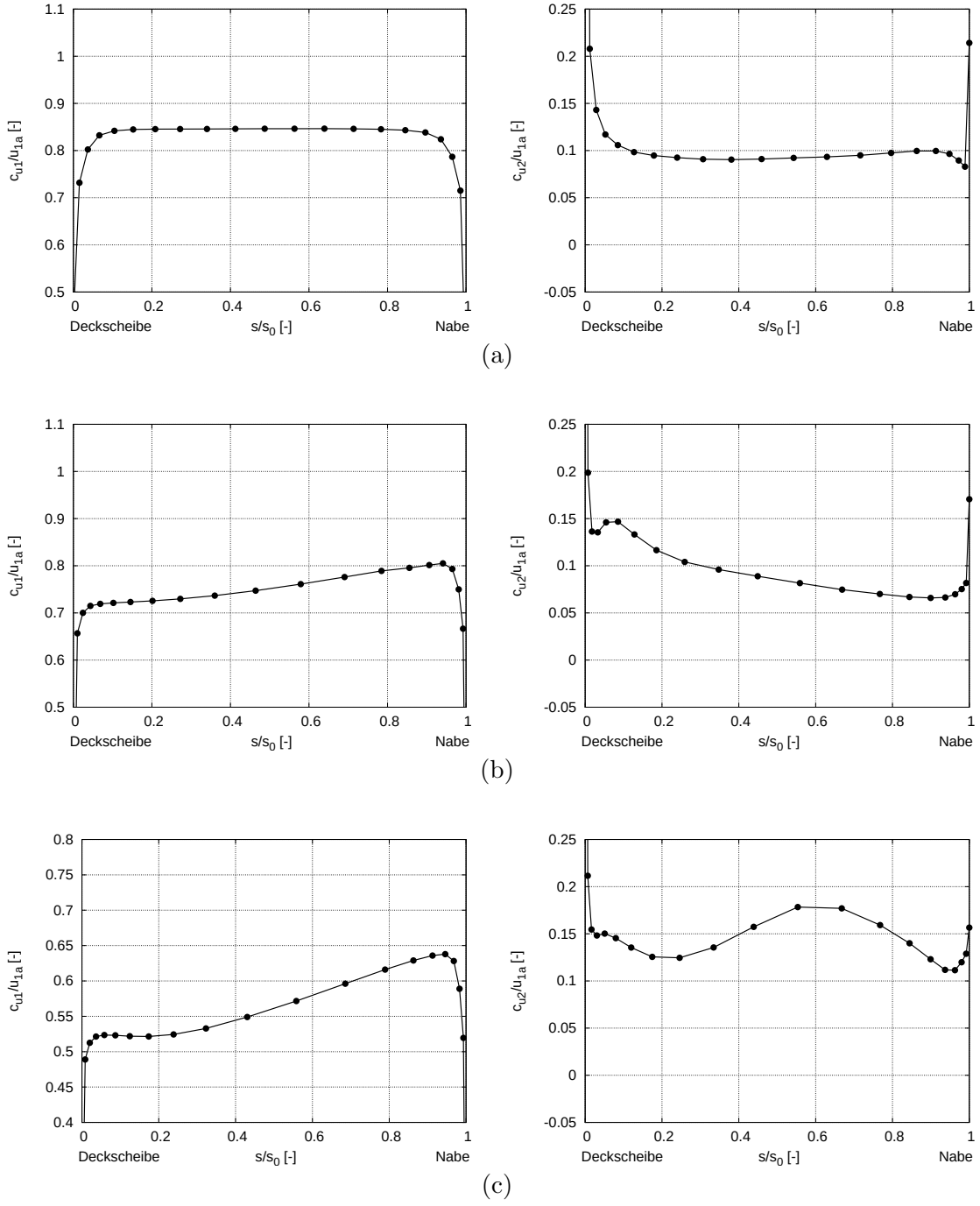
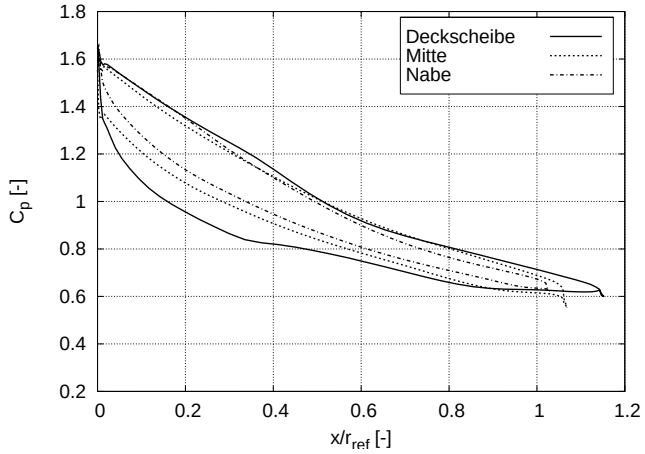
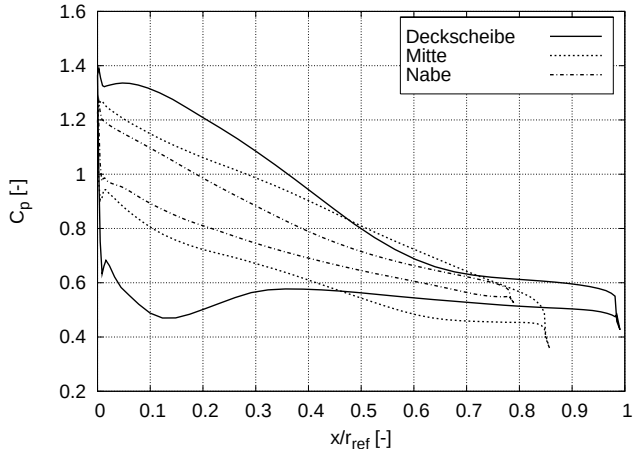


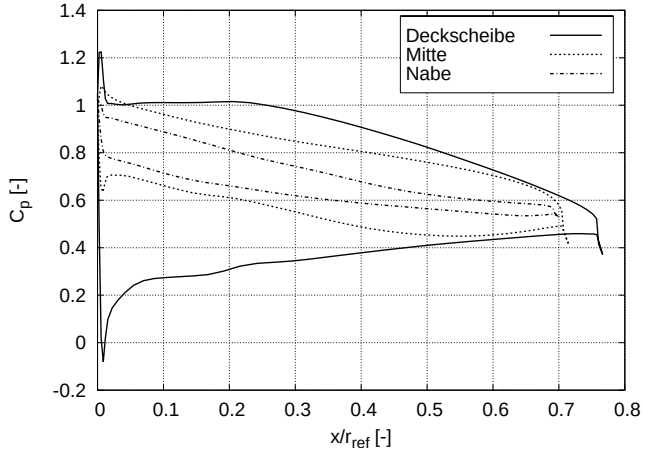
Bild 7.15: Gemittelte Umfangsgeschwindigkeiten entlang der Schaufelein- (links) und -austrittskante (rechts): (a) FT30, (b) FT70, (c) FT110



(a)



(b)



(c)

Bild 7.16: Dimensionslose Druckverteilung auf der Laufschaufel an der Deckscheibe, der Schaufelmitte und der Nabe: (a) FT30, (b) FT70, (c) FT110

7.2 Kreiselpumpen

Ebenfalls in Abhängigkeit von der spezifischen Drehzahl n_q wurden eine Modellbaureihe von sechs Kreiselpumpen entworfen. Die Meridiankonturen der Pumpen wurden dabei aus einer im RTD integrierten Statistik berechnet. Zum Generieren der Laufräder wurde das im RTD nach der Theorie von Pfleiderer [31] arbeitende Erstentwurfsmodul eingesetzt. Eine Optimierung der Geometrien wurde nicht durchgeführt.

Wie auch bei der Turbinenbaureihe ist die Drehzahl, der Außendurchmesser des Laufrades und die Profilierung bei allen Pumpen identisch. Tab. 7.7 zeigt die identischen Laufraddaten.

Tabelle 7.7: Identische Laufraddaten der einzelnen Pumpen der Modellbaureihe

Drehzahl	n	$[\frac{1}{\text{min}}]$	1000
Außendurchmesser	D_{2a}	[mm]	340
Profildicke Hinterkante	d_{HK}	[mm]	5
maximale Profildicke	d_{max}	[mm]	5
relative Dickenrücklage	$x_{d_{max}}/l$	[%]	10

Die Auslegungsdaten der Kreiselpumpen sind in Tab. 7.8 zusammengefasst. Der Verlauf der Volumenzahl φ sowie der Druckzahl Ψ_t in Abhängigkeit von der spezifischen Drehzahl n_q ist grafisch in Bild 7.17 dargestellt.

Tabelle 7.8: Auslegungsdaten der zu trainierenden Kreiselpumpen

Spez. Drehzahl	n_q	$[\frac{1}{\text{min}}]$	10	20	30	50	70	90
Volumenzahl	φ	[-]	0.0051	0.0197	0.0411	0.0857	0.1239	0.1594
Druckzahl	Ψ_t	[-]	1.180	1.1473	1.0889	0.8994	0.7347	0.6214
Volumenstrom	Q	$[\frac{\text{m}^3}{\text{s}}]$	0.0083	0.0319	0.0664	0.1384	0.2003	0.2576
Förderhöhe	H	[m]	19.061	18.532	17.589	14.528	11.867	10.038
Schaufelzahl La.	z_{La}	[-]	6	7	7	9	10	10

7.2.1 Geometrievergleich Approximation – Original

Die gewählte Netztopologie sowie die für das Training verwendeten Lernparameter der beiden beteiligten neuronalen Netze zum Trainieren der Meridiankontur und der konformen Abbildung sind in Tab. 7.9 bzw. 7.10 zusammengestellt.

Die Approximationsgenauigkeit der beiden neuronalen Netze wird durch den Vergleich der approximierten Geometrien mit den Lerngeometrien in Bild 7.18 verdeutlicht.

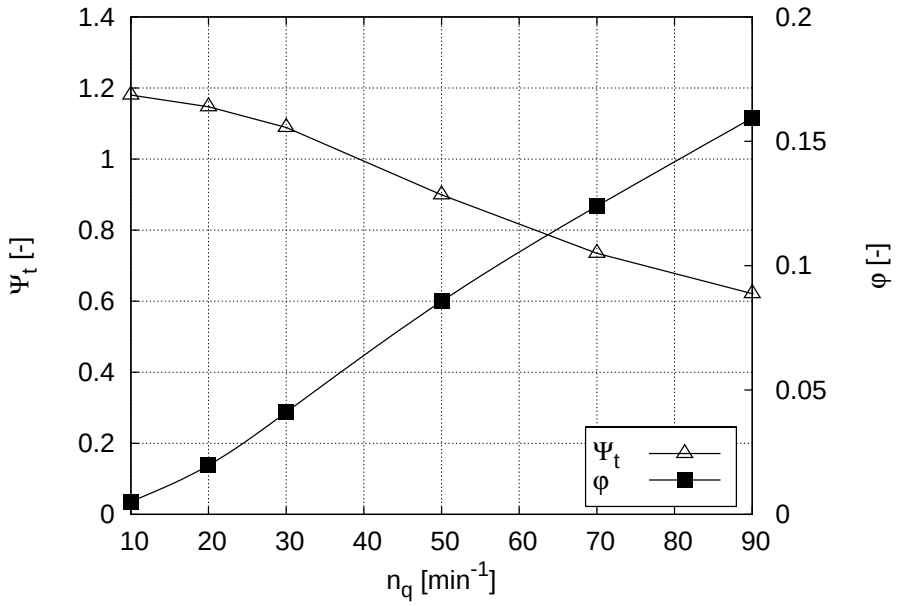


Bild 7.17: Verlauf der Volumenzahl φ und der Druckzahl Ψ_t im Auslegungspunkt in Abhängigkeit von der spezifischen Drehzahl n_q der zu lernenden Kreiselpumpen

Tabelle 7.9: Netztopologie und Lernparameter für das Training der Meridiankonturen der Pumpen

Netztopologie		1-5-20
Lernmodus		Batch
Lernrate	η	0.3
Momentum	α	0.9
Weight Decay	d	0.0
Netzfehler	E_{min}	10^{-8}

Tabelle 7.10: Netztopologie und Lernparameter für das Training der konformen Abbildung der Laufradbeschaufelung

Netztopologie		1-7-45
Lernmodus		Batch
Lernrate	η	0.3
Momentum	α	0.9
Weight Decay	d	$4.5 \cdot 10^{-8}$
Netzfehler	E_{min}	10^{-8}

Man erkennt auch hier, dass die maximalen Abweichungen der R- und Z-Koordinaten, der Streckenverhältnisse μ sowie die konformen Längen ΔL der approximierten Geometrieparameter deutlich kleiner sind als die Winkelgrößen $\Delta\varphi$ bzw. β_S.

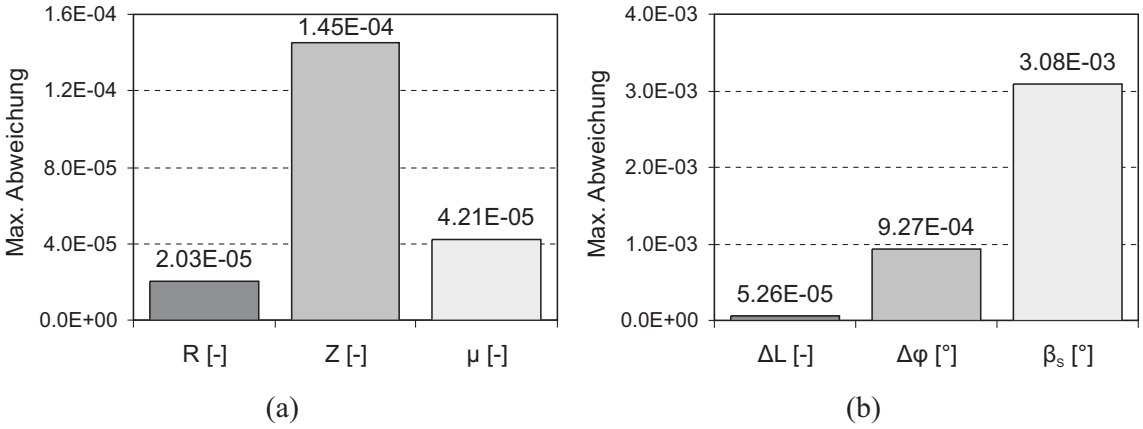


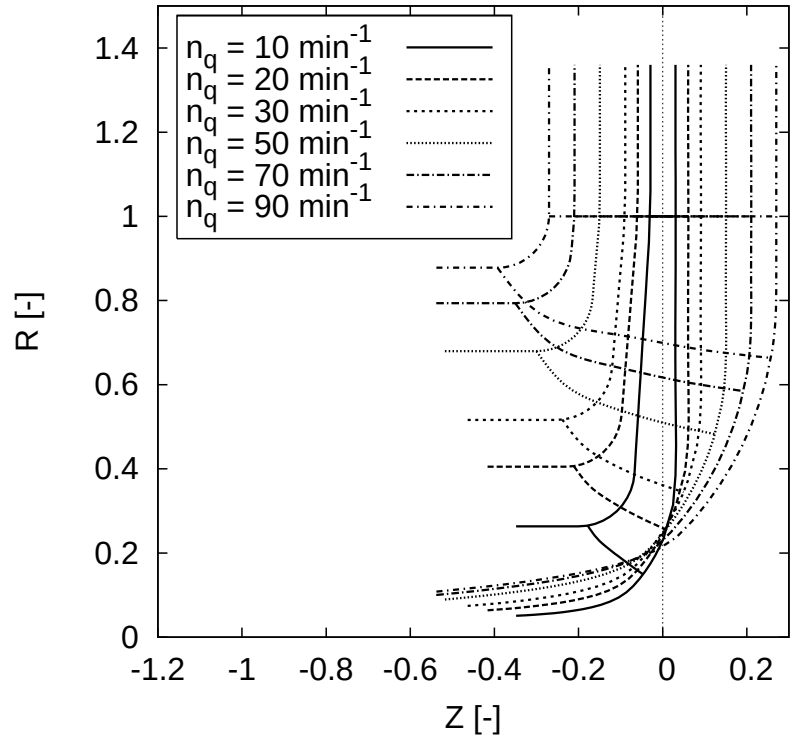
Bild 7.18: Maximale Abweichungen der von den beiden neuronalen Netzen approximierten Geometrieparameter zu den in den Lernmustern enthaltenen Geometrieparametern: (a) Meridiangeometrie, (b) konforme Abbildung

Allerdings fällt die höhere Abweichung der Geometrieparameter zur Beschreibung der Z-Koordinaten auf. Hierbei handelt es sich um den Geometrieparameter Z_{BCi} eines B-Spline Kontrollpunktes zur Beschreibung der Nabenkontur, s. Bild 4.14. Dieser Wert hat für alle trainierten Lerngeometrien den konstanten Wert 0. Die Approximation eines konstanten Wertes stellt für ein neuronales Netz mit nichtlinearer Aktivierungsfunktion ein Problem dar, da sich eine geeignete Kombination der Gewichte zur Abbildung gerader Strecken aus den nichtlinearen Aktivierungsfunktionen als schwierig erweist. Lediglich durch extrem hohe Gewichtswerte lässt sich der exponentielle Anteil der hier verwendeten sigmoiden Aktivierungsfunktion derart unterdrücken, dass die Darstellung einer Geraden möglich wäre. Da aber hohe Gewichtswerte aufgrund der schlechteren Generalisierungseigenschaften des Netzes unerwünscht sind, nimmt man lieber die ohnehin schon vernachlässigbar kleinen Abweichungen zu den Lerngeometrien in Kauf.

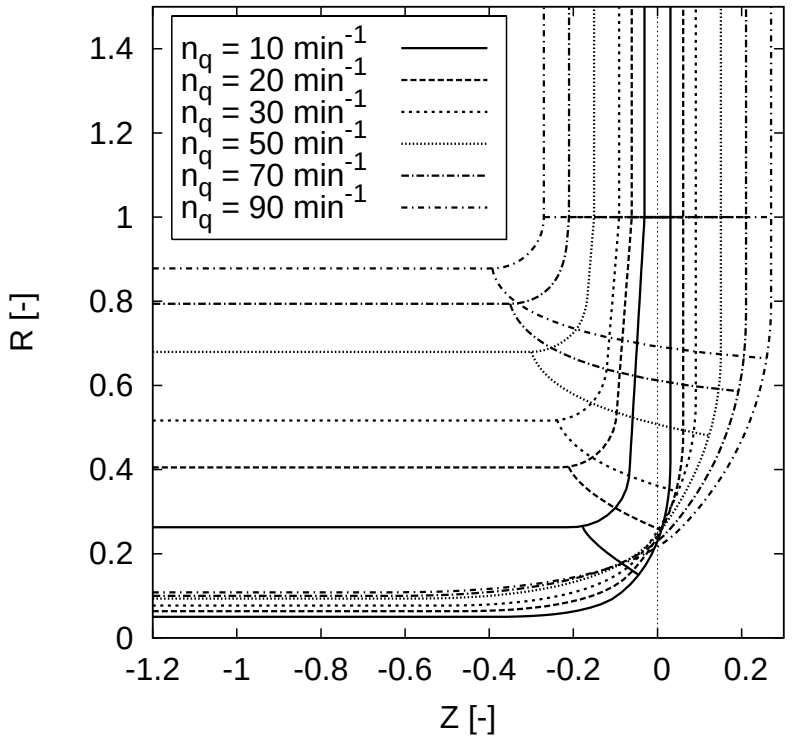
Da die Meridiankonturen der Originalgeometrien bereits systematisch aus der bereits oben erwähnten Pumpenstatistik berechnet wurden, erkennt man in Bild 7.19 kaum Unterschiede zwischen Approximation und Original.

Der meridionale Flächenverlauf ist für drei repräsentative Pumpen mit den spezifischen Drehzahlen $n_q = 20$ min^{-1}, $n_q = 50$ min^{-1} und $n_q = 90$ min^{-1} in Bild 7.20 dargestellt. Beim Vergleich des Flächenverlaufs mit den Originalgeometrien beobachtet man leichte Abweichungen im Austrittsbereich des Laufrades für spezifische Drehzahlen $n_q < 40$ min^{-1}. Dies ist auf die verwendete Parametrisierung der Geometrie zurückzuführen, die den Übergang einer schrägen Deckscheibenkontur im Laufradbereich in einen rein radialen Austrittsbereich erlaubt. Dennoch wird der Übergang im KNN-Designsystem stetig durch einen B-Spline beschrieben. Eine Änderung des Knotenvektors des B-Splines wäre demnach für eine weitere Glättung der Kurve an dieser Stelle die einfachste Variante.

Bild 7.21 zeigt die konforme Abbildung sowie den Schaufelwinkelverlauf der drei Pumpen im Bereich der Deckscheibe, der Schaufelmitte und der Nabe. Man erkennt, dass die konforme Abbildung im Vergleich zu den Originalgeometrien bei

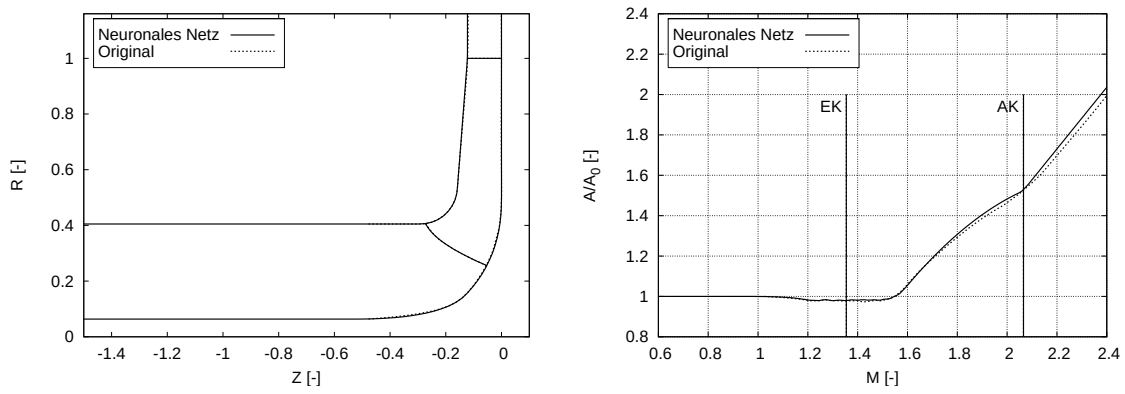


(a)

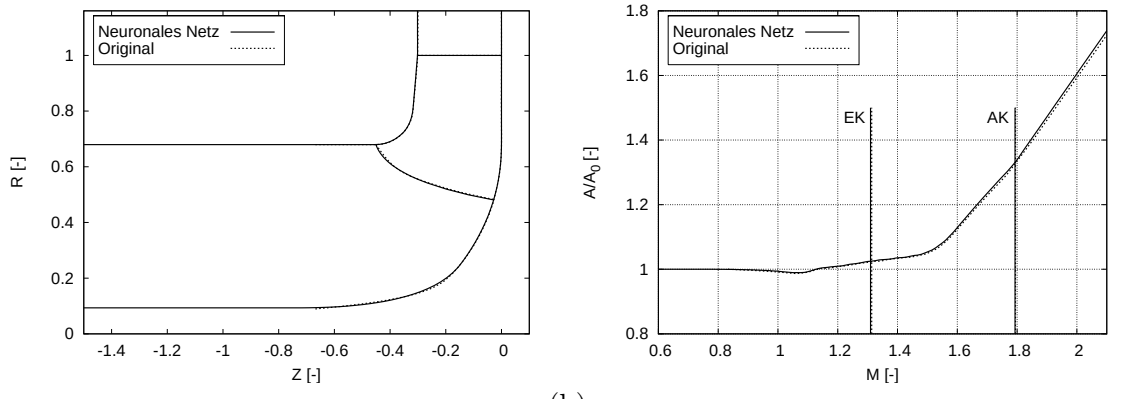


(b)

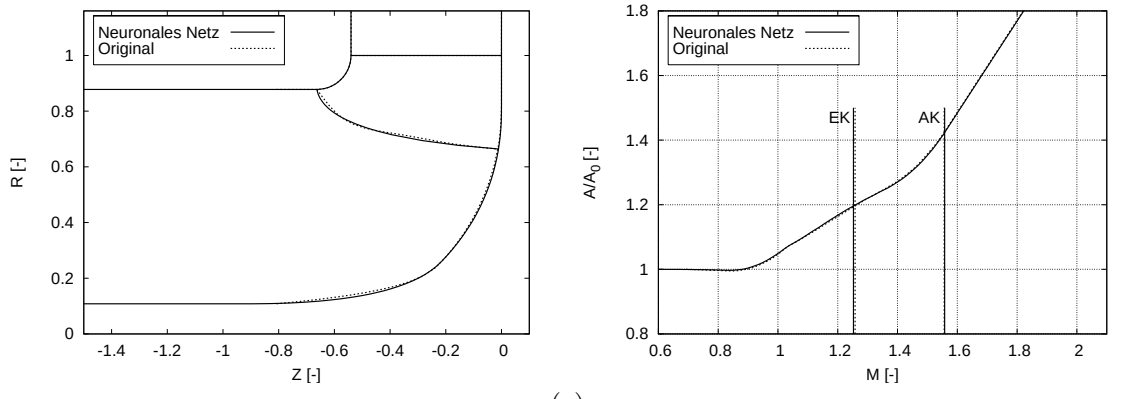
Bild 7.19: Meridiangeometrie der untersuchten Pumpen: (a) Original, (b) Approximation



(a)



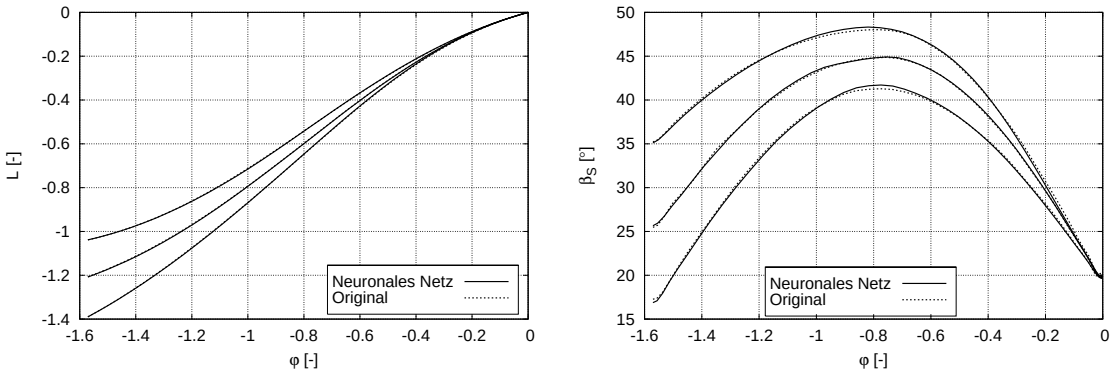
(b)



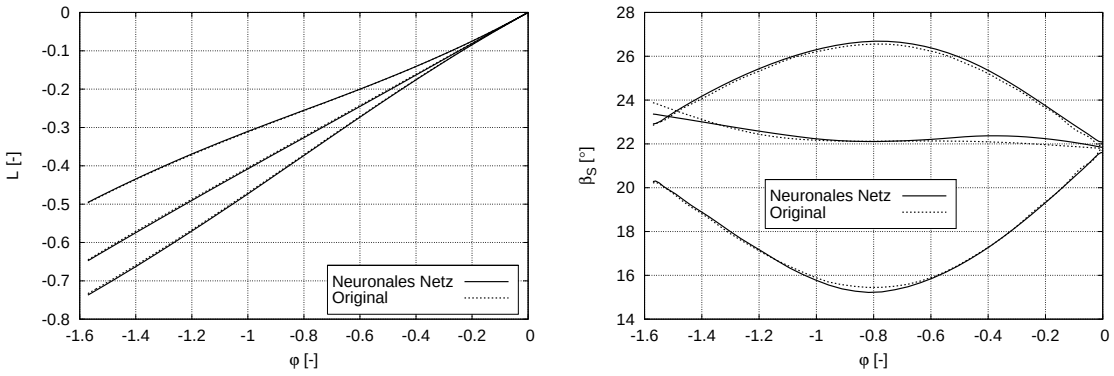
(c)

Bild 7.20: Vergleich der Meridiankontur (links) sowie des meridionalen Flächenverlaufs (rechts) zwischen approximierter Geometrie und Originalgeometrie: (a) $n_q = 20$ min^{-1}, (b) $n_q = 50$ min^{-1}, (c) $n_q = 90$ min^{-1}

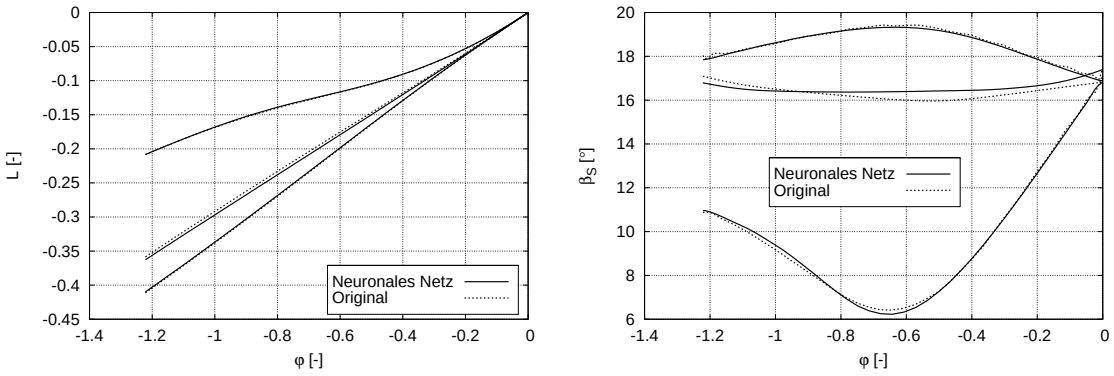
allen Pumpen sehr gut approximiert wird. Die Schaufelwinkelabweichungen von unter einem Grad ergeben sich aus den bei der Parametrisierung und Systematisierung der konformen Abbildung verursachten Abweichungen sowie den bei der Erstellung der dreidimensionalen Geometrie und der Neuverschneidung gemachten Fehlern.



(a)

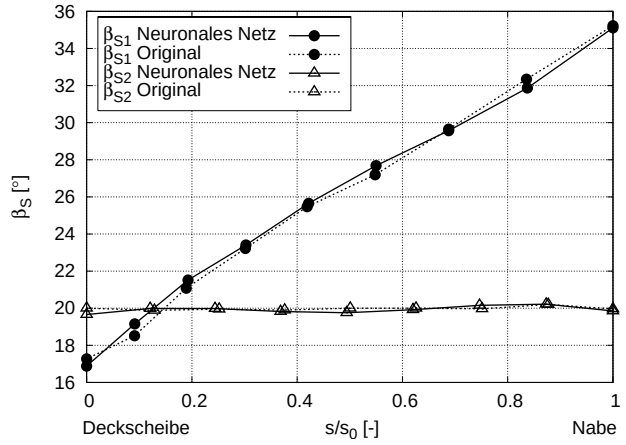


(b)

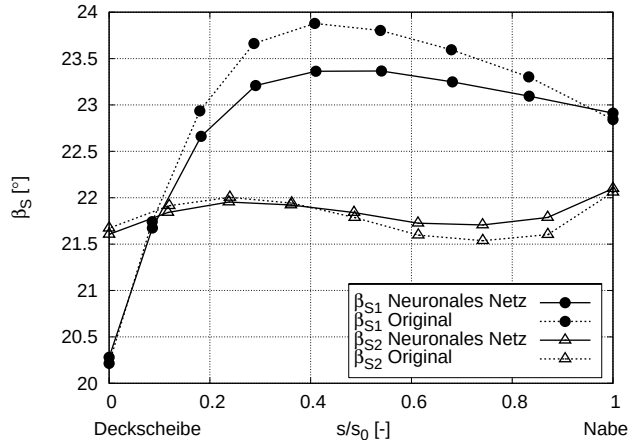


(c)

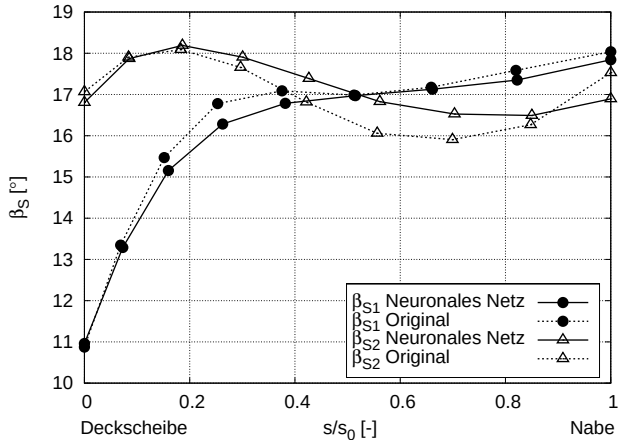
Bild 7.21: Vergleich der konformen Abbildung (links) sowie des Schaufelwinkelverlaufs (rechts) zwischen approximierter Geometrie und Originalgeometrie an Deckscheibe, Schaufelmitte und Nabe: (a) $n_q = 20$ min^{-1}, (b) $n_q = 50$ min^{-1}, (c) $n_q = 90$ min^{-1}



(a)



(b)



(c)

Bild 7.22: Vergleich der Ein- und -austrittswinkelverteilung der Laufradbeschaufelung zwischen approximierter Geometrie und Originalgeometrie: (a) $n_q = 20\ \mathrm{min}^{-1}$, (b) $n_q = 50\ \mathrm{min}^{-1}$, (c) $n_q = 90\ \mathrm{min}^{-1}$

In Bild 7.22 werden die Unterschiede in der Schaufelwinkelverteilung entlang der Ein- und -austrittskante deutlich. Eine Verschlechterung des Schaufelaustrittswinkels an der Deckscheibe kann im Vergleich zum Schaufeleintrittswinkel bei den Turbinen für größer werdende spezifischen Drehzahlen nicht beobachtet werden, da die Austrittskante des Laufrades bei allen Pumpen im radial verlaufenden Austrittsbereich liegt und somit eine übermäßig starke Verschiebung des Punktes (R_{2a}, Z_{2a}) in Z-Richtung ausgeschlossen ist.

7.2.2 Vergleich der Strömungsergebnisse Approximation – Original

Die Berechnung der Strömung durch die untersuchten Pumpen wurden auf Netzen mit insgesamt 22 000 Punkten durchgeführt. Für die Turbulenzmodellierung wurde ebefalls das Standard k-ϵ Modell mit Wandfunktionen verwendet. Aufgrund des schlechten Konvergenzverhalten bei der Verwendung eines Verfahrens 2. Ordnung wurde zur Gewährleistung einer stabilen Lösung eine Linearkombination aus dem 2. Ordnung genauen Diskretisierungsschema MINMOD und dem 1. Ordnung genauen Diskretisierungsschema UDS[1] mit einem konstanten Blendingfaktor $\beta = 0.4$ verwendet, s. Skoda [54]. In Tab. 7.11 sind die Netzdimensionen sowie die wesentlichen Einstellparameter des 3D Navier-Stokes-Codes zur Berechnung der Strömung zusammengefasst.

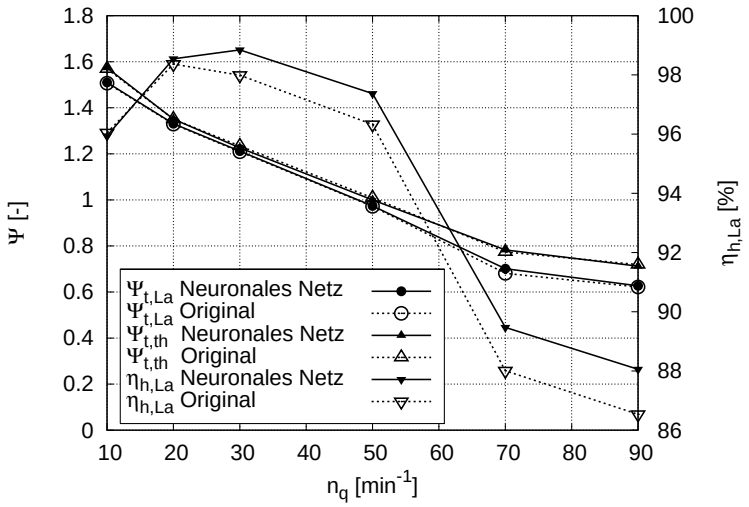
Tabelle 7.11: Netzdimensionen und Einstellparameter des 3D Navier-Stokes-Codes

N_i	N_j	N_k	N_{ges}	Diskretisierungsschema	Konvergenzkriterium
80	25	11	22 000	MINMOD $\beta = 0.4$	10^{-4}

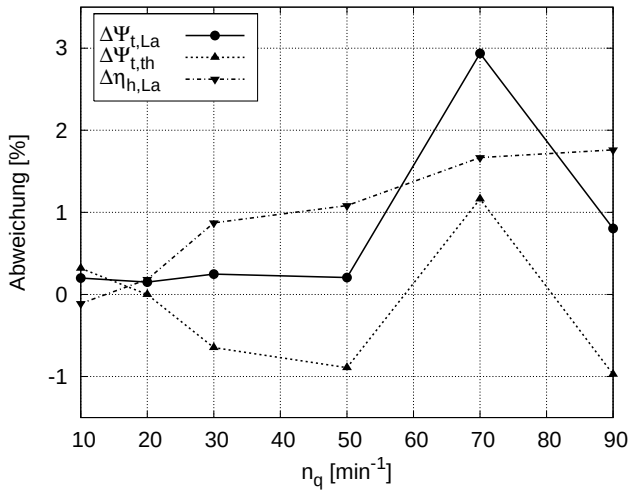
Bild 7.23a zeigt den Verlauf der Druckzahl $\Psi_{t,La}$, der theoretischen Totaldruckzahl $\Psi_{t,th}$ sowie des hydraulischen Wirkungsgrades des Laufrades $\eta_{h,La}$ der Approximation im Vergleich zum Original in Abhängigkeit von der spezifischen Drehzahl. Die prozentuale Abweichungen dieser Größen ist in Bild 7.23b dargestellt.

Man erkennt, dass die Ergebnisse der Druckzahlen $\Psi_{t,La}$ und $\Psi_{t,th}$ der approximierten Geometrien mit einer betragsmäßigen Abweichung von unter 3% sehr gut mit den Ergebnissen der Originalgeometrien übereinstimmen. Bei fast allen approximierten Geometrien stellte sich jedoch ein besserer Laufradwirkungsgrad $\eta_{h,La}$ als beim Original heraus. Lepach [26] hat in seiner Arbeit durch eingehende Untersuchungen jedoch gezeigt, dass sich die berechneten Wirkungsgrade zweier durchgeführter Strömungsberechnungen mit einem Konvergenzkriterium von 10^{-4} bei *gleicher* Geometrie und unterschiedlichen Startlösungen bereits bis zu 0.5 Prozentpunkte unterscheiden können, sodass die maximale Abweichung im Wirkungsgrad von ca. 1.8 Prozentpunkten zu relativieren ist. Ein weiterer Grund für die hohen Wirkungsgradabweichungen liegt in der unterschiedlichen Aufbereitung der Geometrien. Während die

[1] UDS = Upwind Differencing Scheme



(a)



(b)

Bild 7.23: Vergleich der Druckzahlen $\Psi_{t,La}$ und $\Psi_{t,th}$ sowie des Wirkungsgrades $\eta_{h,La}$ der berechneten Pumpen zwischen Approximation und Original: (a) integrale Kenngrößen, (b) prozentuale Abweichung der Kenngrößen

aus der Pumpenstatistik gewonnene Meridiankontur aus zusammengesetzten Kreisbögen und Geraden besteht und zur besseren Lage des Kontrollraumeintritts von Hand verlängert werden musste, wird die Meridiankontur aus dem neuronalen Netz durch stetig aneinander grenzende B-Splines approximiert, die einen glatteren Verlauf der Deckscheiben- und Nabenkontur sowie der Schaufelein- und -austrittskanten ermöglichen. Ebenso kann auch die Systematisierung der Meridiankonturen und der konformen Abbildungen Grund für eine Verbesserung des Wirkungsgrades sein.

Die Bilder 7.24 und 7.25 zeigen die Verteilungen der umfangsgemittelten Meridian- und Umfangsgeschwindigkeiten der Strömung entlang der Schaufelein- und -austrittskante für die drei repräsentativen Pumpenlaufräder. Hier zeigt sich eine gute Übereinstimmung sowohl der Meridian- als auch der Umfangsgeschwindigkeitsver-

teilung. Im Meridiangeschwindigkeitsverlauf entlang der Austrittskante erkennt man bei der Pumpe mit der spezifischen Drehzahl $n_q = 20$ min^{-1} im Deckscheibenbereich eine leichte Rückströmung des Fluids in das Laufrad, was auf ein zu große Austrittsbreite des Laufrades hindeutet.

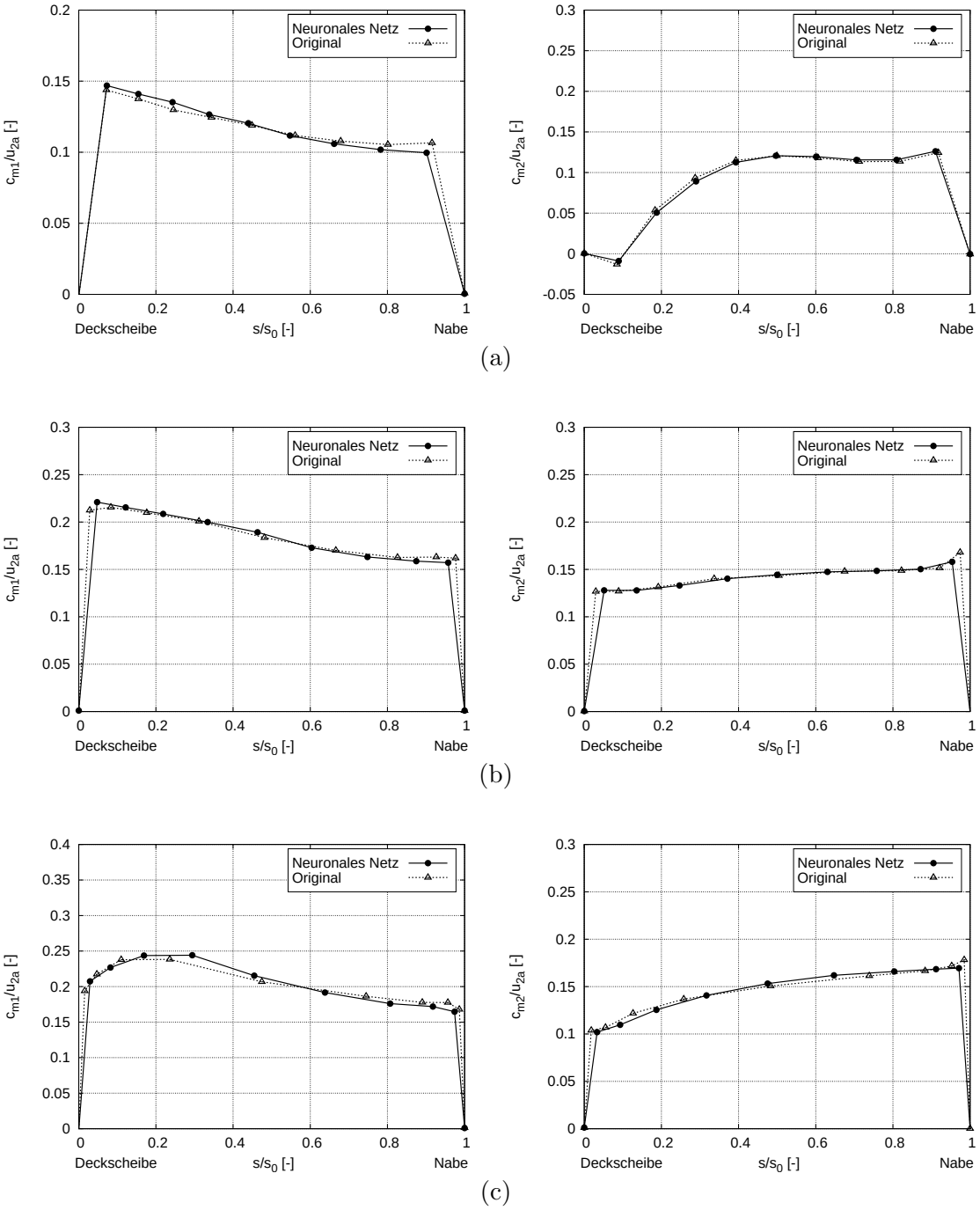
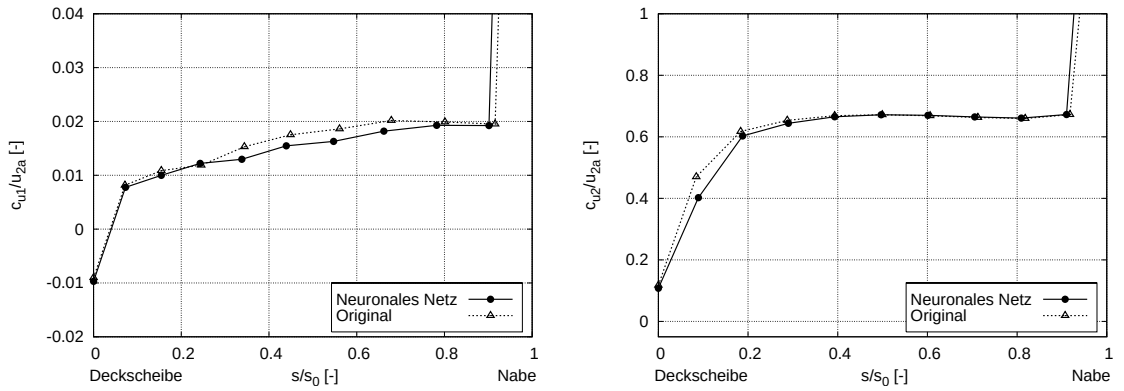
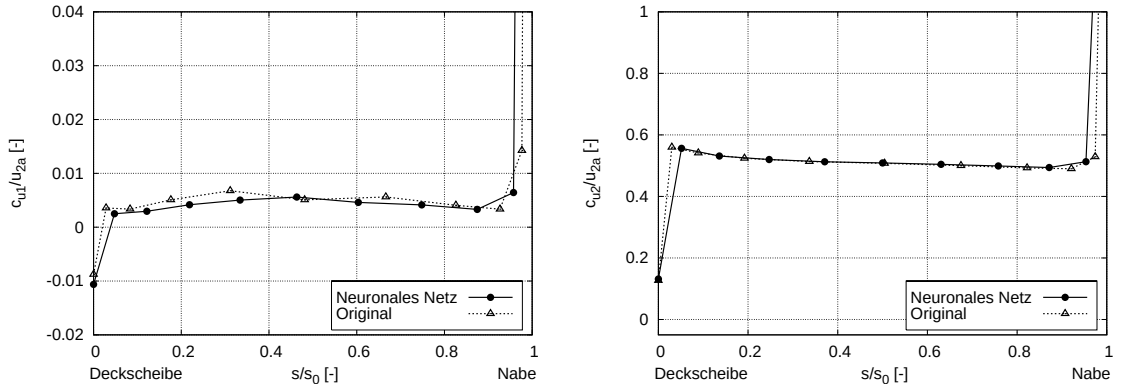


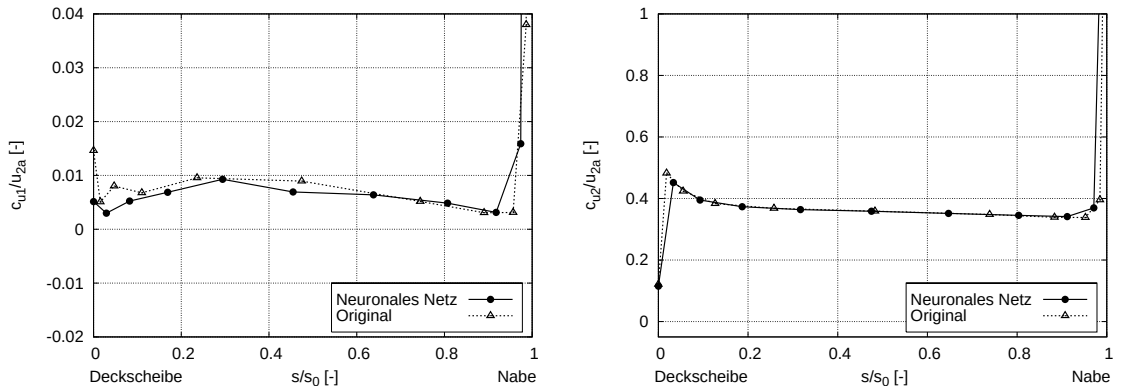
Bild 7.24: Vergleich der gemittelten Meridiangeschwindigkeiten entlang der Schaufelein- (links) und -austrittskante (rechts) zwischen Approximation und Original: (a) $n_q = 20$ min^{-1}, (b) $n_q = 50$ min^{-1}, (c) $n_q = 90$ min^{-1}



(a)



(b)



(c)

Bild 7.25: Vergleich der gemittelten Umfangsgeschwindigkeiten entlang der Schaufelein- (links) und -austrittskante (rechts) zwischen Approximation und Original: (a) $n_q = 20$ min^{-1}, (b) $n_q = 50$ min^{-1}, (c) $n_q = 90$ min^{-1}

In Bild 7.26 werden die dimensionslosen Druckverteilungen auf der Laufschaufel an der Deckscheibe und in der Schaufelmitte verglichen. Auf die Druckverteilungen an der Nabe wurde aus Gründen der kaum bemerkbaren Unterschiede zwischen Approximation und Original ebenfalls verzichtet.

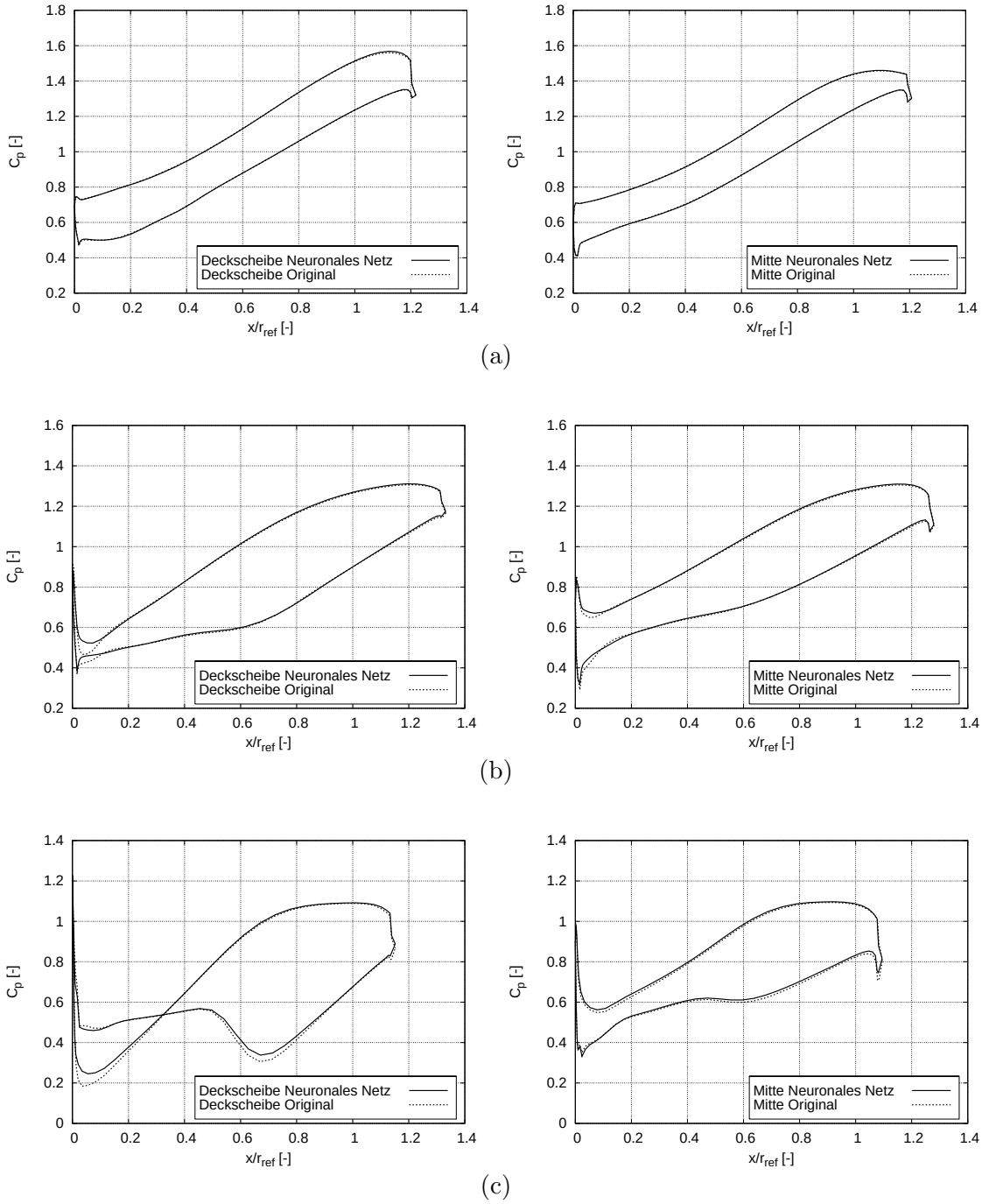


Bild 7.26: Vergleich der dimensionslosen Druckverteilung auf der Laufschaufel an der Deckscheibe (links) und in der Schaufelmitte (rechts) zwischen Approximation und Original: (a) $n_q = 20$ min^{-1}, (b) $n_q = 50$ min^{-1}, (c) $n_q = 90$ min^{-1}

Während die Druckverteilung in der Schaufelmitte bei allen Pumpen nahezu identisch ist, ergeben sich im Bereich der Deckscheibe bei den Pumpen mit den spezifischen Drehzahlen $n_q = 50$ min^{-1} und $n_q = 90$ min^{-1} vor allem im Eintrittsbereich kleinere Unterschiede. Die Approximation der Pumpe mit der spezifischen Drehzahl $n_q = 50$ min^{-1} weist eine gleichmäßigere und stärkere Belastung des vorderen Schaufelbereichs sowie eine kleinere Saugspitze auf als das Original. Dadurch wird

die Strömung besser geführt und eine bessere Energieumsetzung in der Maschine erreicht. Bei der Pumpe mit der spezifischen Drehzahl $n_q = 90$ min^{-1} zeigt sich ein typisches Überlastverhalten. Der Schaufeleintrittswinkel β_{S1} wurde hierbei für den vorgegebenen Volumenstrom im Erstentwurf zu niedrig berechnet, so dass sich der Druckverlauf im Eintrittsbereich des Laufrades überschneidet. Dennoch ist die Überschneidung bei der approximierten Geometrie geringer als beim Original und somit auch die Strömungsverluste.

7.2.3 Beurteilung interpolierter Geometrien

Zur Beurteilung der Generalisierungseigenschaften der neuronalen Netze wurden die Geometrien für die in Tab. 7.12 zusammengefassten Pumpenlaufrädern generiert und nachgerechnet.

Tabelle 7.12: Auslegungsdaten der interpolierten Kreiselpumpen

Spez. Drehzahl	n_q	$[\frac{1}{\text{min}}]$	15	40	60	80	85
Volumenzahl	φ	[-]	0.0114	0.0637	0.1058	0.1417	0.1507
Druckzahl	Ψ_t	[-]	1.1653	0.9946	0.8122	0.6723	0.6460
Volumenstrom	Q	$[\frac{\text{m}^3}{\text{s}}]$	0.0184	0.1030	0.1711	0.2290	0.2435
Förderhöhe	H	[m]	18.823	16.065	13.120	10.859	10.435
Schaufelzahl La.	z_{La}	[-]	6	9	10	10	10

Bild 7.27 zeigt die Meridiankonturen der approximierten und interpolierten Geometrien. Auch hier sieht man deutlich, dass die Meridiankonturen systematisch auseinander hervor gehen.

Auffallend sind kleinere Unstimmigkeiten bei den Pumpen mit den spezifischen Drehzahlen $n_q = 80$ min^{-1} und $n_q = 85$ min^{-1} an der Deckscheibenkontur im Bereich des Laufradaustritts, die auf Schwankungen der Abbildungsfunktion des neuronalen Netzes in diesem Bereich hindeuten. Eine Erhöhung des Weight Decay Lernparameters d könnte hier zu besseren Ergebnissen führen.

Die Ergebnisse für drei repräsentative Pumpen mit den spezifischen Drehzahlen $n_q = 15$ min^{-1}, $n_q = 40$ min^{-1} und $n_q = 80$ min^{-1} sollen genauer analysiert werden.

In Bild 7.28 ist die konforme Abbildung sowie der Schaufelwinkelverlauf der drei Pumpen abgebildet. Bei allen Schaufelgeometrien ist ein gleichmäßiger und im Vergleich zu den approximierten Geometrien sinnvoller Verlauf der konformen Abbildung sowie der Schaufelwinkel zu beobachten.

Die integralen Werte der Druckzahl $\Psi_{t,La}$, der theoretischen Druckzahl $\Psi_{t,th}$ und des hydraulischen Wirkungsgrades $\eta_{h,La}$ sind in Bild 7.29 für alle aus dem neuronalen Netz generierten Pumpengeometrien dargestellt. Über den gesamten spezifischen Drehzahlbereich ist ebenso wie bei den Turbinen ein relativ gleichmäßiger Verlauf

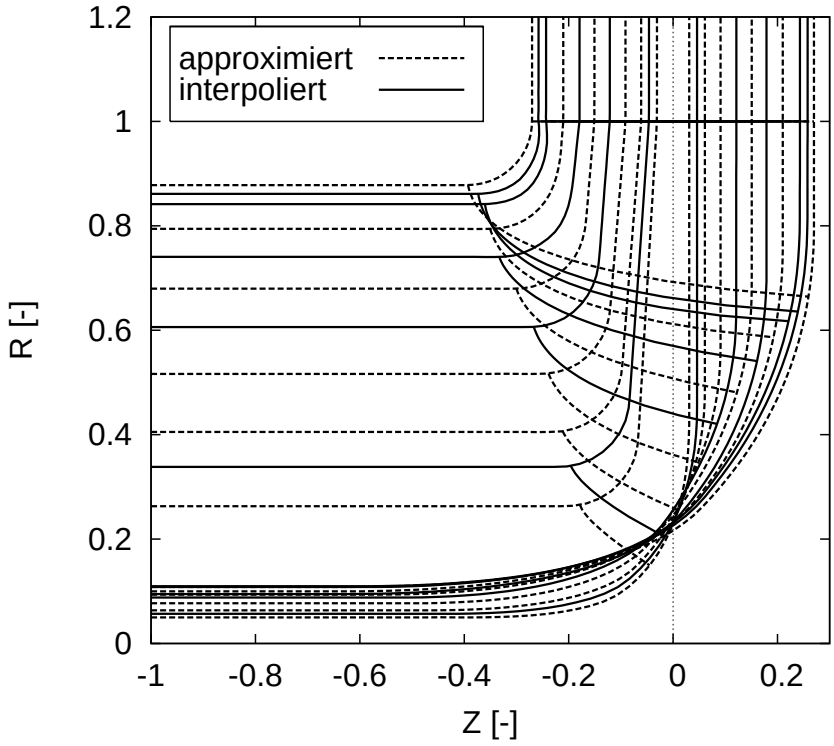


Bild 7.27: Approximierte und interpolierte Meridiangeometrien in Abhängigkeit von der spezifischen Drehzahl im Bereich $n_q = 10$ – 90 min^{-1}

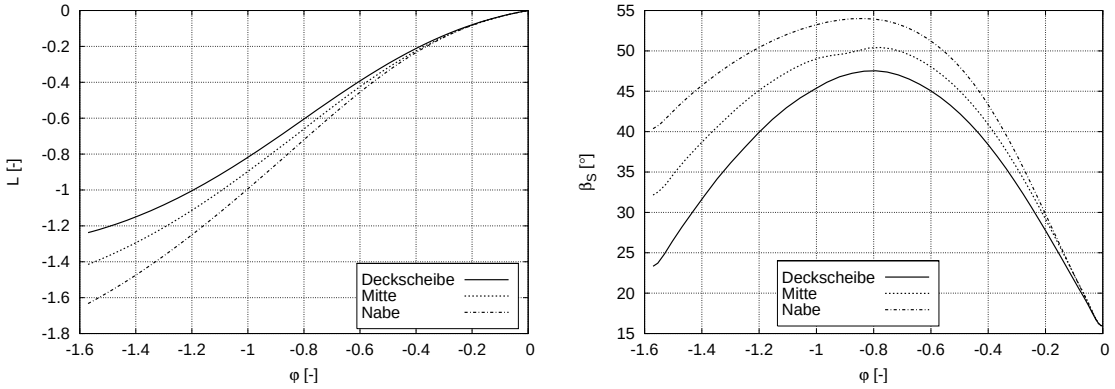
der Druckzahlen zu beobachten. Lediglich die Pumpenlaufräder mit den spezifischen Drehzahlen $n_q = 60$ min^{-1}, $n_q = 70$ min^{-1} und $n_q = 80$ min^{-1} weichen von dem zu erwarteten Wirkungsgradverlauf ab, wobei das Originallaufrad mit der spezifischen Drehzahl $n_q = 70$ min^{-1} ebenfalls nur einen Wirkungsgrad von $\eta_{h,La} = 88\%$ aufweist, vgl. Bild 7.23a, und somit das Lernergebnis des neuronalen Netzes im umliegenden spezifischen Drehzahlbereich stark beeinflusst.

Die umfangsgemittelten Geschwindigkeitsverteilungen der drei repräsentativen Pumpenlaufräder entlang der Schaufelein- und -austrittskante sowie die Druckverteilung auf der Schaufel entlang der Deckscheibe, der Schaufelmitte und der Nabe sind in den Bildern 7.30, 7.31 und 7.32 dargestellt.

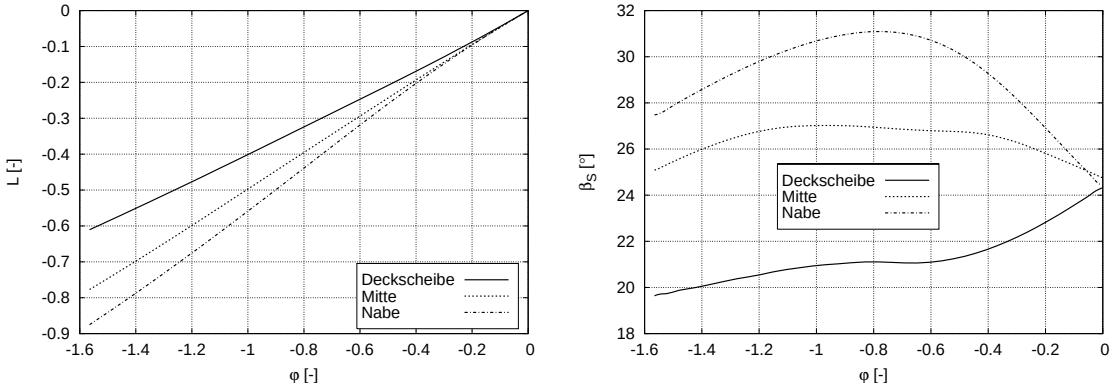
Vergleicht man die Geschwindigkeitsverteilungen mit den Ergebnissen der approximierten Pumpen aus den Bildern 7.24 und 7.25, so stellt man ein ähnliches Verhalten der Geschwindigkeitsverläufe fest. Bei dem Pumpenlaufrad mit der spezifischen Drehzahl $n_q = 15$ min^{-1} erkennt man im Meridiangeschwindigkeitsverlauf, wie auch schon bei dem Laufrad mit der spezifischen Drehzahl $n_q = 20$ min^{-1} in Bild 7.24a, ein deutlich ausgeprägtes Rückströmgebiet im Bereich des Laufradaustritts, bedingt durch die zu große Laufradaustrittsbreite.

Das Teillastverhalten des Pumpenlaufrades mit der spezifischen Drehzahl $n_q = 15$ min^{-1} ist auch in den Druckverteilungen entlang der Schaufel in Bild 7.32a an der Saugspitze im Eintrittsbereich der Schaufel zu sehen. Ansonsten ist die Schaufel bis zum Austritt auf allen Strombahnen gleichmäßig belastet. Bei der Pumpe mit

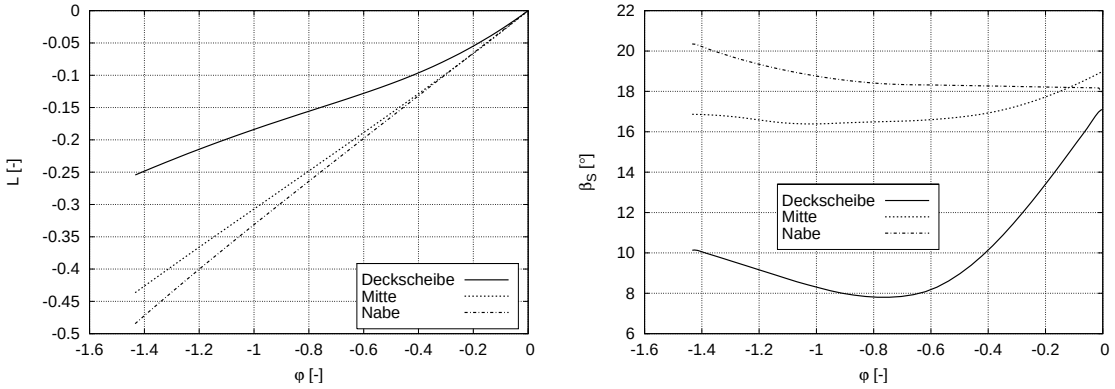
der spezifischen Drehzahl $n_q = 40$ min^{-1} zeichnet sich im Eintrittsbereich an der Deckscheibe bereits eine Druckspitze und eine geringere Belastung der Schaufel ab, was auf einen zu kleinen Schaufeleintrittswinkel schließen lässt. In der Schaufelmitte und an der Nabe wird die Schaufel dagegen nahezu optimal angeströmt.



(a)



(b)



(c)

Bild 7.28: Konformen Abbildung (links) sowie Schaufelwinkelverlauf (rechts) der interpolierten Schaufelgeometrien: (a) $n_q = 15$ min^{-1}, (b) $n_q = 40$ min^{-1}, (c) $n_q = 80$ min^{-1}

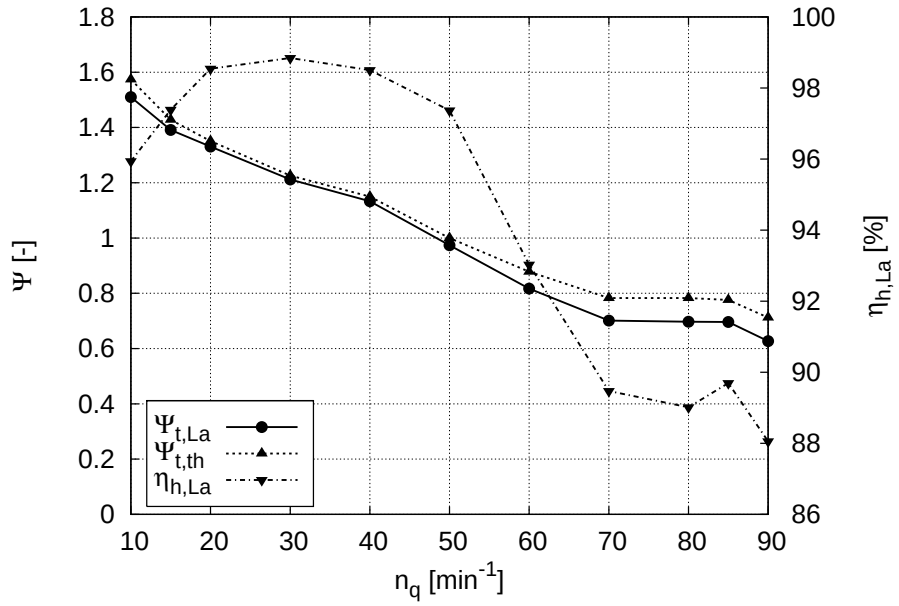


Bild 7.29: Integralwerte der approximierten und interpolierten Pumpen: Totaldruckerhöhung im Laufrad $\Psi_{t,La}$, theoretische Totaldruckerhöhung $\Psi_{t,th}$ und hydraulischer Laufradwirkungsgrad $\eta_{h,La}$

In Bild 7.32c beobachtet man eine Überschneidung des Druckverlaufs auf der Deckscheibe und in der Schaufelmitte, wie sie auch bei dem Pumpenlaufrad mit der spezifischen Drehzahl $n_q = 90$ min^{-1} in Bild 7.26c zu beobachten ist.

Zur Beurteilung des Gesamtergebnisses lässt sich festhalten, dass die Strömungsergebnisse für alle aus dem neuronalen Netz generierten Pumpengeometrien mit den Ergebnissen der Originalgeometrien in akzeptabler Weise übereinstimmen. Durch die Parametrisierung und Systematisierung wurde sogar eine leichte Verbesserung der Strömungsergebnisse erzielt. Eine Optimierung der Geometrien mit anschließendem Nachtrainieren der neuronalen Netze ist allerdings ratsam, um ein besseres Strömungsverhalten in den Pumpen zu erhalten und somit auch eine genauere Strömungsberechnung mit einem Verfahren 2. Ordnung sowie einem niedrigeren Konvergenzkriterium zu ermöglichen.

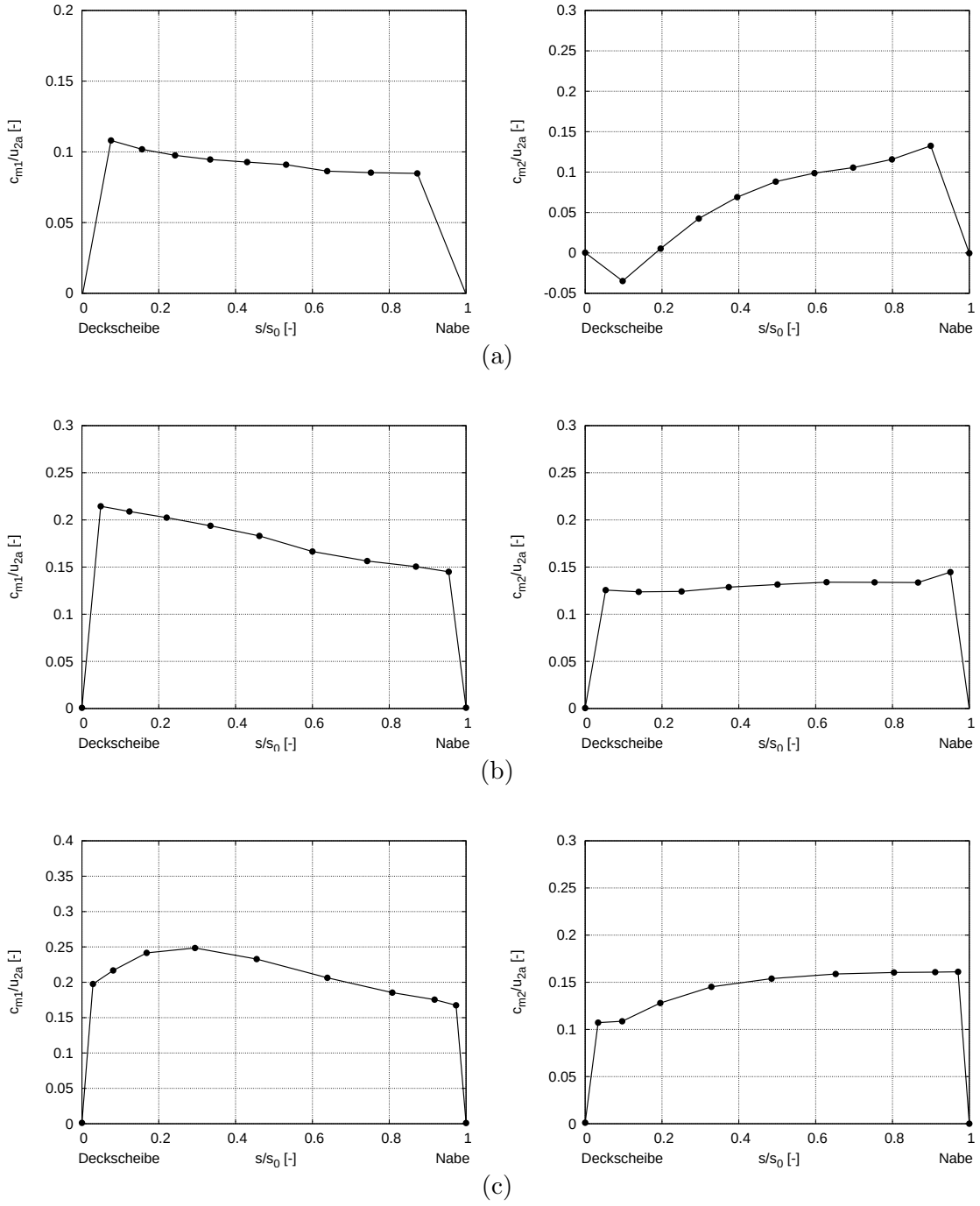


Bild 7.30: Gemittelte Meridiangeschwindigkeiten entlang der Schaufelein- (links) und -austrittskante (rechts): (a) $n_q = 15$ min^{-1}, (b) $n_q = 40$ min^{-1}, (c) $n_q = 80$ min^{-1}

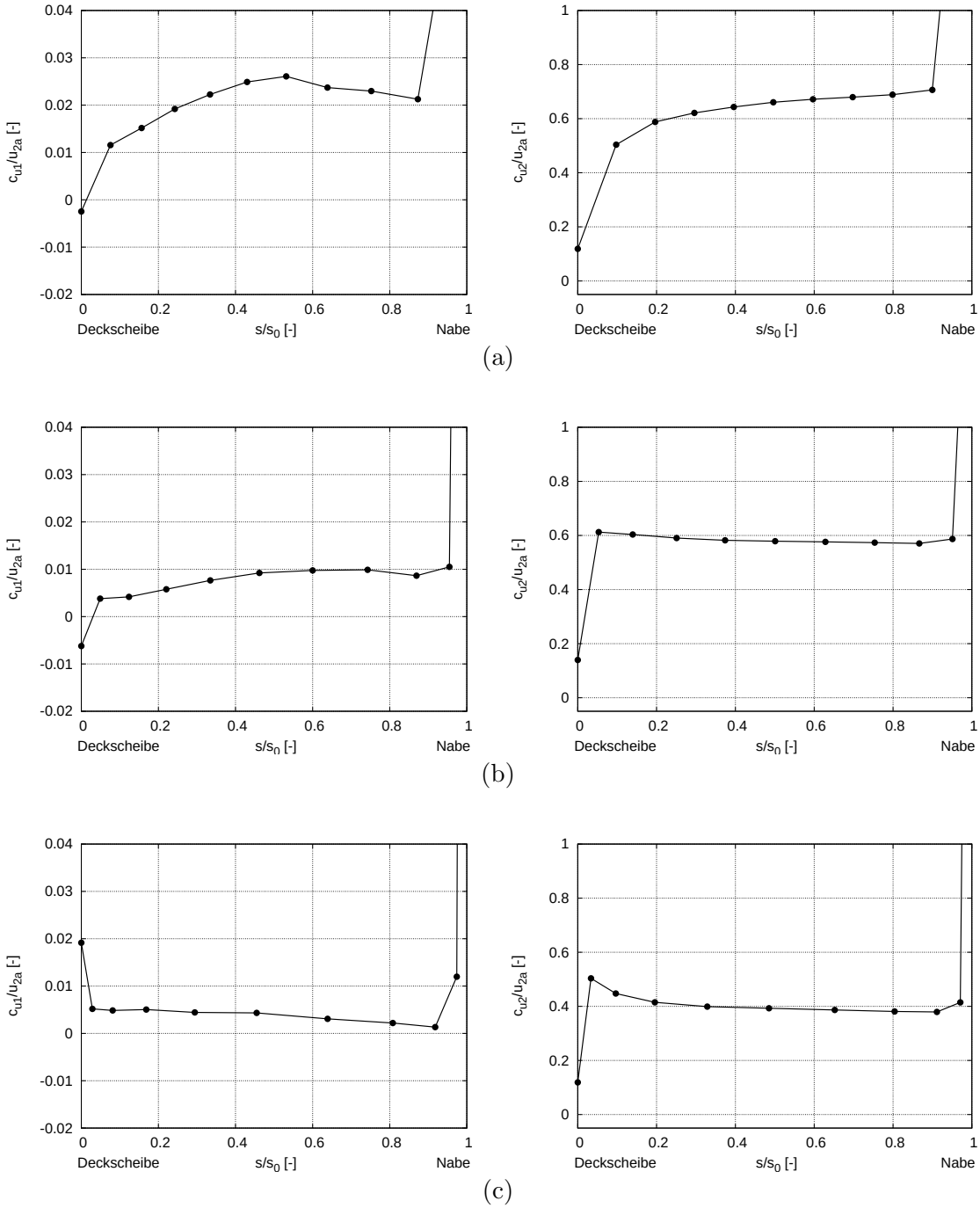
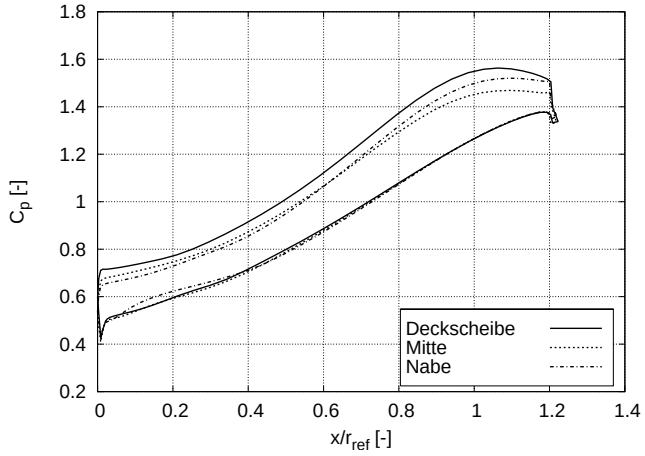
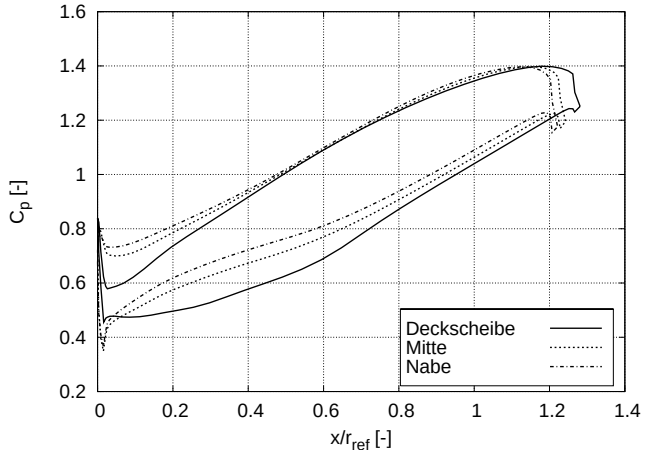


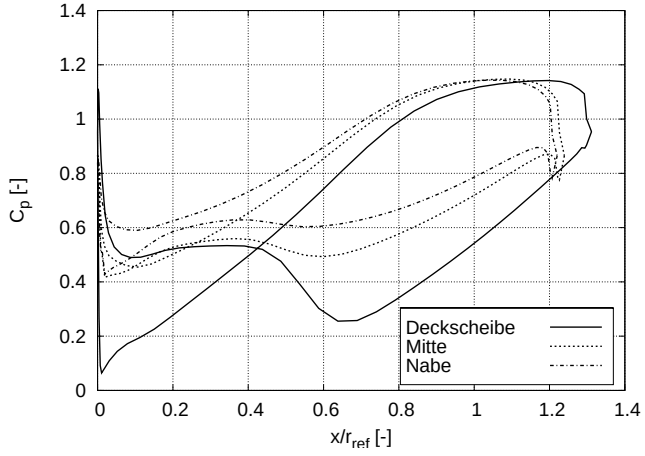
Bild 7.31: Gemittelte Umfangsgeschwindigkeiten entlang der Schaufelein- (links) und -austrittskante (rechts): (a) $n_q = 15$ min^{-1}, (b) $n_q = 40$ min^{-1}, (c) $n_q = 80$ min^{-1}



(a)



(b)



(c)

Bild 7.32: Dimensionslose Druckverteilung auf der Laufschaufel an der Deckscheibe, der Schaufelmitte und der Nabe: (a) $n_q = 15$ min^{-1}, (b) $n_q = 40$ min^{-1}, (c) $n_q = 80$ min^{-1}

Kapitel 8

Bewertung und Ausblick

Das in der vorliegenden Arbeit entwickelte KNN-Designsystem stellt ein mächtiges Werkzeug für das schnelle Entwerfen nahezu optimaler Beschaufelungen hydraulischer Maschinen dar. Das System ermöglicht die Aufbereitung und Parametrisierung vorhandener Geometrien von Francis Turbinen und Kreiselpumpen, die in Abhängigkeit von einer beliebigen Anzahl an Betriebspunktparametern mit dem integrierten Backpropagation Lernverfahren trainiert werden können. Mit Hilfe der trainierten neuronalen Netze können anschließend neue Strömungsmaschinengeometrien unter Angabe einer beliebigen spezifischen Drehzahl generiert werden, die vergleichbare strömungsmechanische Eigenschaften wie die Originalgeometrien aufweisen. Damit ein optimales Lernergebnis erzielt werden kann, stehen dem Anwender zwei interaktive Modifikationsbausteine zur Systematisierung der parametrisierten Geometrien sowie der daraus gewonnenen Lernmuster zur Verfügung. Entsprechend der Entwurfsmethodik des Lehrstuhls erfolgt die Parametrisierung und Modifikation sowie das Training der rotationssymmetrischen Meridiangeometrien und der konformen Abbildung der Schaufelskelettfläche jeweils getrennt voneinander. Zur besseren Beurteilung der Geometrien lassen sich im System auch während der Modifikation sowohl der meridionale Flächenverlauf als auch der Schaufelwinkelverlauf darstellen. Eine Visualisierung der Lernergebnisse ist ebenfalls möglich.

Der modulare Aufbau des Systems ermöglicht es zum einen, unterschiedliche Strömungsmaschinenbauarten und weitere Parametrisierungsstrategien problemlos zu integrieren, ohne in das grafische Grundsystem eingreifen zu müssen. Zum anderen kann das System als grafische Benutzeroberfläche und Analysewerkzeug für den entwickelten Neuronale-Netze-Simulator zum Trainieren beliebiger tabellarisch aufbereiteter Daten verwendet werden.

Am Beispiel einer Baureihe von sechs optimierten Francis Turbinenstufen im spezifischen Drehzahlbereich $n_q = 20 - 120$ min^{-1} wurde gezeigt, dass der verwendete Lernalgorithmus in der Lage ist, einen funktionalen Zusammenhang zwischen spezifischer Drehzahl und Geometrie zu erkennen und zu lernen. Sowohl bei den approximierten als auch bei den interpolierten Geometrien konnten im Vergleich zu den Originalgeometrien hervorragende Strömungseigenschaften festgestellt werden. Der Einsatz des KNN-Designsystems bedeutet somit eine erhebliche Zeitersparnis im Entwurfs- und Optimierungsprozess neuer Strömungsmaschinen, da beliebige Geo-

metrien mit einem optimalen Strömungsverhalten in Sekundenbruchteilen aus dem neuronalen Netz abgerufen werden können.

Ebenso wurde eine Baureihe von sechs Kreiselpumpen im spezifischen Drehzahlbereich $n_q = 10 - 90$ min^{-1} parametrisiert und trainiert, bei denen es sich um Erstentwurfsgeometrien handelt. Hierbei konnte sogar eine Verbesserung des Strömungsverhaltens bei fast allen, aus dem neuronalen Netz generierten Geometrien im Vergleich zu den Originalgeometrien festgestellt werden.

Bei den untersuchten Geometrien wurde lediglich die Meridiankontur sowie der Verlauf der Skelettfläche des Laufrades parametrisiert und gelernt. Zur Vervollständigung der Geometrie sollten in Zukunft auch die Profilierung des Laufrades und die Schaufelzahl sowie weitere eventuell vorhandene Stufenelemente mitgelernt werden.

Weiterhin wurden die Geometrien lediglich in Abhängigkeit der spezifischen Drehzahl trainiert, was die Bandbreite möglicher Geometrieformen sowie die Fähigkeiten des neuronalen Netzes bei weitem nicht ausschöpft. Wünschenswert wäre eine Unterscheidung der Geometrien durch mehrere Kenngrößen, die sowohl die vorgegebenen strömungsmechanischen Eigenschaften als auch die geometrischen Forderungen berücksichtigen. Voraussetzung für die Generierung optimaler Geometrien aus einem neuronalen Netzes mit mehreren Eingabeparametern ist jedoch eine entsprechend große Anzahl an Lerngeometrien. Auch hierfür stellt sich das entwickelte KNN-Designsystem als äußerst effizientes Werkzeug heraus. Sowohl durch die Aufnahme neuer Geometrien in die Lerndatenbank als auch durch das Ersetzen nicht oder unzureichend optimierter Geometrien, lässt sich das System stetig verbessern, so dass in kürzester Zeit eine optimale Lerndatenbasis zur Verfügung steht.

Literaturverzeichnis

[1] Arnold, K. ; Gosling, J. ; Holmes, D.: *The Java Programming Language*. 3rd Edition. Addison-Wesley, 2000

[2] Aschenbrenner, T.: *Entwurf strömungsoptimaler Beschaufelungen*. Technische Universität München, Lehrstuhl und Laboratorium für Hydraulische Maschinen und Anlagen, Dissertation, 1998

[3] Bader, R.: *Simulation kompressibler und inkompressibler Strömungen in Turbomachinen*. Technische Universität München, Lehrstuhl und Laboratorium für Hydraulische Maschinen und Anlagen, Dissertation, 2000

[4] Bader, R. ; Böhm, C. ; Mochkaai, Y. ; Knapp, W. ; Schilling, R.: Loss Analysis in a Full Spiral Kaplan Turbine. In: *Proceedings of the ERCOFTAC Workshop on Draft Tube Flow*. Porjus Hydropower Center, Schweden, Juni 1999

[5] Bézier, P.: Mathematical and practical possibilities of UNISURF. In: Barnhill, R.-E. (Hrsg.) ; Riesenfeld, R. (Hrsg.): *Computer Aided Geometric Design*. Academic Press, 1974, S. 127–152

[6] Boussinesq, J.: Essai sur La Théorie Des Eaux Courantes. In: *Mem. Présenté Acad. Sci.* 23 (1877), S. 46. – Paris

[7] Chakravarthy, S. R. ; Osher, S.: High resolution applications of the OSHER upwind scheme for the Euler equations. In: *AIAA paper 83-1943* (1983)

[8] Cichocki, A. ; Unbehauen, R.: *Neural Networks for Optimization and Signal Processing*. New York : John Wiley & Sons, 1993

[9] Denton, J. D.: Designing in Three Dimensions. In: *Turbomachinery Design Using CFD*. AGARD, 1994 (Lecture Series 195), Kapitel 3

[10] Eschenauer, H. ; Koski, J. ; Osyczka, A.: *Multicriteria Design Optimization, Procedures and Applications*. Berlin Heidelberg : Springer Verlag, 1990

[11] Fahlman, S. E.: An empirical study of learning speed in back-propagation networks. In: Touretzky, D. (Hrsg.) ; Hinton, G. (Hrsg.) ; Sejnowski, T. (Hrsg.): *Proc. of the 1988 Connectionist Models Summer School*. Carnegie Mellon University, USA : Morgan Kaufmann, 1989

[12] Fernández, A.: *Strömungstechnische Optimierung von Beschaufelungen hydraulischer Maschinen*. Technische Universität München, Lehrstuhl und Laboratorium für Hydraulische Maschinen und Anlagen, Dissertation, 1997

[13] Ferziger, J. H. ; Perić, M.: *Computational Methods for Fluid Dynamics*. Berlin : Springer Verlag, 1996

[14] Fritz, J.: *Strömungswechselwirkungen in hydraulischen Maschinen*. Technische Universität München, Lehrstuhl und Laboratorium für Hydraulische Maschinen und Anlagen, Dissertation, 1999

[15] Geist, A. ; Beguelin, A. ; Dongarra, J. ; Jiang, W. ; Manchek, R. ; Sunderam, V.: *PVM: Parallel Virtual Machine. A Users' Guide and Tutorial for Networked Parallel Computing*. Cambrigde: Massachusetts Institute of Technology, The MIT press, 1994

[16] Goldberg, D. E.: *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA : Addison-Wesley, 1989

[17] Graubard, S. R. (Hrsg.): *Probleme der Künstlichen Intelligenz*. Wien : Springer Verlag, 1996

[18] Harten, A.: High resolution schemes for hyperbolic conservation laws. In: *Journal of Computational Physics* 49 (1983), Nr. 3, S. 357–393

[19] Hooke, R. ; Jeeves, T. A.: Direct Search Solution of Numerical and Statistical Problems. In: *Association for Computing Machinery* 8 (1961), S. 212–229

[20] Hoschek, J. ; Lasser, D.: *Grundlagen der geometrischen Datenverarbeitung*. 2. Stuttgart : B. G. Teubner Verlag, 1992

[21] Koch, M. (Hrsg.) ; Kuhn, Th. (Hrsg.) ; Wernstedt, J. (Hrsg.): *Fuzzy Control: Optimale Nachbildung und Entwurf optimaler Entscheidungen*. R. Oldenbourg Verlag, 1996

[22] Kronschnabl, F.: *Erstellung eines plattformunabhängigen Postprocessingwerkzeuges in Java zur Analyse von Strömungsrechenergebnissen*. Technische Universität München, Lehrstuhl für Fluidmechanik, Diplomarbeit, 2001

[23] Kronschnabl, F.: Algebraische Generierung von Strombahnen für Strömungsmaschinen / Technische Universität München. 2003. – Interner Bericht

[24] Kruse, H. ; Mangold, R. ; Mechler, B. ; Penger, O.: *Programmierung Neuronaler Netze: eine Turbo Pascal Toolbox*. 1. Auflage. Addison-Wesley, 1991

[25] Launder, B. P. ; Spalding, D. B.: The numerical computation of turbulent flows. In: *Computer Methods in applied mechanics and engineering* 3 (1974), S. 269–289

[26] Lepach, T.: *Entwurf und Optimierung von Francis Turbinen*. Technische Universität München, Lehrstuhl für Fluidmechanik, Dissertation, 2005

[27] Minsky, M. L. ; Papert, S. A.: *Perceptrons*. Cambridge, MA : MIT Press, 1969

[28] Müller, N.: *Multi-Level CFD-Technik zur Optimierung von Beschaufelungen Hydraulischer Maschinen*. Technische Universität München, Lehrstuhl für Fluidmechanik, Dissertation, 2006

[29] Pacheco, P.: *Parallel Programming with MPI*. San Francisco: Morgan Kaufmann Publishers Inc., 1997

[30] Penrose, R.: *Computerdenken: Die Debatte um Künstliche Intelligenz, Bewusstsein und die Gesetze der Physik*. Spektrum Akademischer Verlag, 2002

[31] Pfleiderer, C. ; Petermann, H.: *Strömungsmaschinen*. 6. Berlin Heidelberg New York : Springer Verlag, 1991

[32] Piegl, L.: On NURBS: A Survey. In: *IEEE Computer Graphics & Applications* (1991), Jan., S. 55–71

[33] Reynolds, O.: On the Dynamical Theory of Incompressible Viscous Fluids and the Determination of the Criterion. In: *Philosophical Transactions of the Royal Society of London, Series A* 186 (1895), S. 123

[34] Richter, R.: *3D-Echtzeit-Entwurf von Beschaufelungen hydraulischer Strömungsmaschinen auf Multiprozessorsystemen*. Technische Universität München, Lehrstuhl und Laboratorium für Hydraulische Maschinen und Anlagen, Dissertation, 1999

[35] Riedel, N.: *Rotor-Stator Wechselwirkung in hydraulischen Maschinen*. Technische Universität München, Lehrstuhl und Laboratorium für Hydraulische Maschinen und Anlagen, Dissertation, 1997

[36] Riedel, N. ; Schilling, R.: Entwicklung eines 3D Euler Codes zur Berechnung der stationären inkompressiblen Strömung in Lauf- und Leiträdern von Strömungsmaschinen / Lehrstuhl und Laboratorium für Hydraulische Maschinen und Anlagen, TU München. 1996. – Interner Bericht

[37] Rodi, W.: Large-Eddy Simulation of the Flow past Bluff Bodies: State of the Art. In: *Proc. of International Conference on Fluid Enigeering ICFE'97* Bd. 1. Tokyo, Japan, 1997, S. 3 – 13

[38] Rodi, W. ; Bergeles, G.: Engineering Turbulence Modelling and Experiments. In: *Proc. of 3rd Int. Symposium on Engineering Turbulence Modelling and Measurements* Bd. 3. Heraklion, Greece, 1996

[39] Rogers, D. F. ; Fog, N. G.: Constrained B-spline curve and surface fitting. In: *Computer Aided Design* 21 (1989), Dec., S. 641–648

[40] Rumelhart, D. E. ; McClelland, J. L.: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Bd. 1: Foundations. Cambridge, MA : MIT Press, 1986

[41] Sauer, J.: *Neuronale Netze, Fuzzy Control-Systeme und Genetische Algorithmen*. Fachbereich Informatik/Mathematik der Fachhochschule Regensburg, 2003. – Skriptum zur Vorlesung

[42] Schachenmann, A. ; Muggli, F. ; Güelich, J. F.: Comparison of Three Navier-Stokes Codes With LDA-Measurements on an Industrial Radial Pump Impeller. In: *Pumping Machinery* Bd. 154. ASME FED, 1993

[43] Schilling, R.: Numerical Calculation of the Quasi-3D Incompressible Inviscid Flow in Turbomachinery. In: *Proceedings of the* 11^{th} *IAHR Symposium*. Amsterdam, 1982

[44] Schilling, R.: CFD Aided Design of Hydraulic Machinery Bladings. In: Velensek, B. (Hrsg.): *CFD'91 Intensive Course on Computational Fluid Dynamics*. Ljubljana, 1991

[45] Schilling, R.: CFD Aided Design - ein effizientes Werkzeug zur strömungsmechanischen Optimierung von Beschaufelungen hydraulischer Maschinen. In: *VDI-Berichte* (1994), Nr. 1127, S. 209–219

[46] Schilling, R.: Application of CFD-Techniques in Fluid Machinery. In: *The* 12^{th} *International Conference on Fluid Flow Technologies, Conference on Modelling Fluid Flow (CFMM'03)*. Budapest, Hungary, September, 3-6 2003

[47] Schilling, R. ; Knapp, W. ; Bader, R. ; Fritz, J. ; Mochkaai, Y.: Experimental and theoretical loss analysis in a Kaplan turbine. In: *Modelling, Testing & Monitoring for Hydro Powerplants - III*. Aix-en-Provence, France : Int. J. on Hydropower & Dams, 1998, S. 563–570

[48] Schilling, R. ; Riedel, N. ; Ritzinger, S.: A Critical Review of Numerical Models Predicting the Flow through Hydraulic Machinery Bladings. In: *17th IAHR Symposium, Section on Hydraulic Machinery and Cavitation*. Beijing, 1994

[49] Schilling, R. ; Ritzinger, S. ; Stoffel, B. ; Hambrecht, J.: Numerical and Experimental Flow Analysis in Centrifugal Pump Impellers. In: *Proceedings of International Conference on Fluid Enigeering ICFE'97* Bd. 1. Tokyo, Japan, 1997, S. 207–215

[50] Schilling, R. ; Thum, S. ; Müller, N. ; Krämer, S. ; Riedel, N. ; Moser, W.: Design optimisation of hydraulic machinery blading by multi level CFD technique. In: *Proceedings of the* 21^{th} *IAHR Symposium*. Lausanne, 2002

[51] Schilling, R. ; Watzelt, Ch. ; Haas, H. ; Sporer, L. ; Fernández, A.: AIF-Forschungsvorhaben Nr. 8897 Echtzeit-Entwurfssystem / FKM. 1993 (620075). – Abschlußbericht

[52] Schlichting, H.: *Grenzschicht-Theorie*. Verlag G. Braun, Karlsruhe, 1982

[53] Schlichting, H. ; Gersten, K.: *Grenzschicht-Theorie*. Springer, Berlin, 1997

[54] Skoda, R.: *Numerische Simulation abgelöster und transitionaler Strömungen in Turbomaschinen*. Technische Universität München, Lehrstuhl für Fluidmechanik, Dissertation, 2003

[55] Sporer, L.: *Fluid-Struktur Wechselwirkung in axialen Beschaufelungen*, Lehrstuhl und Laboratorium für Hydraulische Maschinen und Anlagen, TU München, Dissertation, 1995

[56] Sporer, L. ; Watzelt, C. ; Haas, H. ; Schilling, R.: Application of a Real-Time Design System to Bulb Turbine Runners. In: *17th IAHR Symposium, Section on Hydraulic Machinery and Cavitation*. Beijing, 1994

[57] Stroustrup, B.: *Die C++-Programmiersprache*. 3. Auflage. Addison-Wesley-Longman, 1998

[58] Thum, S. ; Schilling, R.: Optimization of hydraulic machinery bladings by multilevel CFD techniques. In: *Proceedings of the* 9^{th} *International Symposium on Transport Phenomena and Dynamics of Rotating Machinery, Honolulu, Hawaii, February 10-14*, 2002

[59] Truckenbrodt, E.: *Fluidmechanik I, Grundlagen und elementare Strömungsvorgänge dichtebeständiger Fluide*. 3. Berlin Heidelberg New York : Springer Verlag, 1989

[60] Vu, T. C. ; Heon, K. ; Shyy, W.: A CFD-Based Computer Aided Engineering System for Hydraulic Turbines. In: *Proceedings of the XVII IAHR Symposium, IRCHM(B)*. Beijing, China, 1994

[61] Watzelt, Ch.: *Echtzeit-Entwurf radialer Beschaufelungen auf einem Multiprozessorsystem*. Technische Universität München, Lehrstuhl und Laboratorium für Hydraulische Maschinen und Anlagen, Dissertation, 1995

[62] Watzelt, Ch. ; Haas, H. ; Sporer, L. ; Schilling, R.: Development of a Real-Time Design System for Hydraulic Machinery Bladings. In: *Proceedings of the XVII IAHR Symposium, IRCHM(B)*. Beijing, China, 1994

[63] Werbos, P. J.: *Beyond regression: new tools for prediction and analysis in the behavioral sciences*, Harvard University, Cambridge, MA, Ph. D. thesis, 1974

[64] Werbos, P. J.: Backpropagation: Past and future. In: *Proc. of the Int. Conf. on Neural Networks* Bd. I. New York : IEEE Press, 1988, S. 343–353

[65] Wilcox, D. C.: *Turbulence Modeling for CFD*. 2. La Cañada, California : DCW Industries, Inc., 1998

[66] Wöhler, M.: *Hierarchische Optimierung von Beschaufelungen hydraulischer Maschinen*. Technische Universität München, Lehrstuhl und Laboratorium für Hydraulische Maschinen und Anlagen, Dissertation, 1999

[67] Wu, C. H.: A General Theory of the 3D Flow in Subsonic and Supersonic Turbomachines of Axial, Radial and Mixed Flow Type / NACA. 1952 (TN 2604). – Forschungsbericht

[68] Zell, A.: *Simulation Neuronaler Netze*. R. Oldenbourg Verlag, 2003