Entwicklung eines Konzepts zur robusten Optimierung von neuen Fahrwerksarchitekturen

Von der Fakultät für
Maschinenbau, Elektro- und Energiesysteme der
Brandenburgischen Technischen Universität Cottbus-Senftenberg
zur Erlangung des akademischen Grades eines
Doktors der Ingenieurwissenschaften

genehmigte Dissertation

vorgelegt von

Dipl.-Ing.
Paul Tobe Ubben

geboren am 27. April 1988 in Aurich

Vorsitzender: Prof. Dr.-Ing. Klaus Höschler
Gutachter: Prof. Dr.-Ing. habil. Prof. (NUST) Dieter Bestle
Gutachter: Univ.-Prof. Dipl.-Ing. Dr. techn. Christian Bucher

Tag der mündlichen Prüfung: 28. April 2017

Danksagung

Diese Arbeit entstand in intensiven Jahren bei der Fahrwerkvorentwicklung der Daimler AG und dem Lehrstuhl für Technische Mechanik und Fahrzeugdynamik der BTU Cottbus-Senftenberg, wobei mich viele Personen unterstützt haben bei denen ich mich Bedanken möchte.

In erster Linie bedanke ich mich bei meinem Doktorvater Prof. Dr.-Ing. habil. Prof. (NUST) Dieter Bestle. Du hast dir Zeit genommen, warst mir stets ein verlässlicher Ansprechpartner und hast mich in intensiven Diskussionen vorangetrieben. Neben dem Fachlichen habe ich mich durch unsere Zusammenarbeit auch persönlich stark weiterentwickelt. Weiterer Dank gilt Prof. Dr.-Ing. Klaus Höschler, der kurzfristig den Vorsitz des Prüfungsausschusses übernommen hat sowie Univ.-Prof. Dipl.-Ing. Dr. techn. Christian Bucher, der das zweite Gutachten erstellt und die weite Anreise aus Wien nicht gescheut hat.

Weiterhin möchte ich mich bei Dr.-Ing. Michael Kleczka und Dr.-Ing. Jürgen Haug bedanken. Ihr gabt mir die Möglichkeit diese Arbeit in eurer Abteilung und eurem Team aufzunehmen und zu bearbeiten. Hierbei habt Ihr mich von Beginn an mit großem Vertrauen unterstützt und stets Interesse an meiner Arbeit gezeigt. Besonders dir, Jürgen, danke ich für die offenen Gespräche und die Freiheit, die du mir zur Bearbeitung meiner Dissertation gelassen hast. Bedanken möchte ich mich zusätzlich bei der gesamten Abteilung der Fahrwerkvorentwicklung und dem Team Simulation. Besonders Erwähnen möchte ich Timo Gaugele und Michael Panagiotidis. Euch bin ich mit meinen Fragen auf den Geist gegangen und dennoch habt ihr Zeit für mich gefunden.

Nicht zu vergessen sind die Mitarbeiter des Lehrstuhls für Technische Mechanik und Fahrzeugdynamik. Es war immer interessant und aufschlussreich mit euch zu reden und zu diskutieren. Der Austausch mit dir, Michael Lockan, war besonders fruchtbar und hat seine „Fußabdrücke“ hinterlassen. Ein offenes Ohr hatten Lutz Anklam und Peggy Fobo, die meine Aufenthalte am Lehrstuhl so einfach wie möglich machten und mich bei sämtlichen organisatorischen Aufgaben bestmöglich unterstützten. Vielen Dank euch.

Ein wichtiges Erfolgsrezept war die tolle Unterstützung meiner Studenten Yuchen Fu, Lie He, Chandramouli Gnanasambandham, Konstantin Belasik, Oliver Clemens und Fabian Fitzer.

Zu guter Letzt danke ich meinen Eltern, Schwiegereltern und Geschwistern für das Korrekturlesen und die dafür aufgeopferten Weihnachtsfeiertage. Ganz besonderer Dank gilt meiner Frau Lisa, die Weihnachten, Sylvester und vieles Mehr für mich geopfert hat. Du hast mir den Rücken freigehalten und ohne dich hätte ich es wahrscheinlich nicht geschafft. Danke!

Abstract

Development of a Concept for Robust Design of New Chassis Architectures

Automobile manufactures are competing for the favor of customers which leads to an increasing diversification of vehicle models available in the market. On this occasion, architectures are used more frequently which decreases production and development costs by standardizing components and structures to be used in different vehicle models. Thus, by using architectures there is no longer a contradiction between individualisation and cost efficiency.

The common components need to fullfil the different requirements of all vehicle models based on a single architecture as best as possible. Hence, architecture design profits most by early usage of simulation and computer-based optimization. For this purpose, new processes differing from those used for classical optimization problems are needed. Definition of an efficient design process considering the hierachical structure of individual and common parameters is required. A manufacturer being able to bring new vehicle models quickly and with sufficient quality to market has an advantage. For achieving this, a robust architecture must be designed.

Based on a specific parametric for chassis architectures several optimization strategies are analyzed for a test problem and the most promising is chosen and further improved. In order to enable robust architecture opimization with respect to multiple criteria, the optimization strategy is extended to solve multi-objective problems. Finally, the developed strategy is applied to an industrial problem to show its potential and applicability.

Zusammenfassung

Entwicklung eines Konzepts zur robusten Optimierung von neuen Fahrwerksarchitekturen

Der Wettbewerbsdruck um die Gunst der Kunden im Automobilbau führt zu einer stetig ansteigenden Diversifikation von Fahrzeugmodellen. Hierbei werden vermehrt Architekturen eingesetzt, die Produktions- und Entwicklungskosten durch die Vereinheitlichung von Bauteilen und Strukturen verringern, die in vielen verschiedenen Fahrzeugmodellen eingesetzt werden können. Somit stellen Individualisierung und Kosteneffizienz keinen Widerspruch mehr dar.

Ein wesentlicher Erfolgsfaktor bei der Entwicklung einer Architektur ist der frühe Einsatz von Simulation und computerbasierter Optimierung, um die vereinheitlichten Teile möglichst gut auf die verschiedenen Anforderungen der Fahrzeugmodelle anpassen und auslegen zu können. Die hierfür benötigten Prozesse unterscheiden sich von denen, die für klassische Optimierungsprobleme eingesetzt werden. Die hierarchische Struktur individueller und Architekturrelevanter Größen muss Berücksichtigung finden und in einen möglichst effizienten Auslegungsprozess münden. Zusätzlich ergibt sich ein großer Vorteil, wenn ein Hersteller kurzfristig und schnell neue Fahrzeugmodelle in hinreichender Qualität auf den Markt bringen kann, wofür eine besonders robuste Architektur benötigt wird.

Auf Basis einer in dieser Arbeit definierten Parametrik für Fahrwerksarchitekturen werden verschiedene Optimierungsstrategien untersucht und eine möglichst erfolgversprechende identifiziert sowie mittels geeigneter, effizienzverbessernder Maßnahmen erweitert. Die Optimierungsprobleme werden hierbei mehrkriteriell gelöst, um neben der deterministischen Optimierung auch die robuste Architekturoptimierung zu ermöglichen. Letztendlich zeigt die praxisnahe Optimierung einer Fahrwerksarchitektur das Potential und die Anwendbarkeit der entwickelten Strategie.

Inhaltsverzeichnis

Symbolverzeichnis

Lateinische Symbole

$a_{\hat{d}}$	Fakor zur Beschreibung der Ausdehnung einer Pareto-Front
$\boldsymbol{A}$	Systemmatrix
c	Federsteifigkeit
c_A	Federsteifigkeit des Aufbaus
c_{dyn}	dynamische Steifigkeit
c_F	Federsteifigkeit des Fahrschemels
c_{F2}	innere Federsteifigkeit des Fahrschemels
c_R	Federsteifigkeit des Rads
$\tilde{c}$	statische Entwurfssteifigkeit
C_i	körperfestes Koordinatensystem des i-ten Körpers
d	Dämpferkonstante
d_A	Dämpferkonstante des Aufbaus
d_F	Dämpferkonstante des Fahrschemels
d_{F2}	innere Dämpferkonstante des Fahrschemels
$\hat{d}$	charakteristische Distanz
D	Anzahl der Derivate
D_n	Dämpfungsmaß der n-ten Schwingmode
D_R	Dämpfungsmaß des Rads
$\boldsymbol{D}$	Dämpfungsmatrix
$\boldsymbol{e}$	Einheitvektor
EG	Eigenlenkgradient
$\boldsymbol{E}$	Einheitsmatrix
f	Lagefreiheitsgrad
$f, \hat{f}$	Gütekriterium, approximiertes Gütekriterium
f_n^{eig}	Eigenfrequenz der n-ten Schwingmode
f_A^{eig}	Aufbaueigenfrequenz
f_R^{eig}	Radeigenfrequenz
$\boldsymbol{f}, \hat{\boldsymbol{f}}$	Vektorkriterium, approximiertes Vektorkriterium
$\boldsymbol{f}^l, \boldsymbol{f}^r$	eingeprägte Kraft und Reaktionskraft
F	Kraft
F_p	Menge nicht dominierter Entwürfe im Kriterienraum
F_{Zst}	Zahnstangenkraft
h	Anzahl der Entwurfsvariablen
i	imaginäre Einheit $i^2 = -1$
i_L	Gesamtlenkübersetzung

I_i	Stützstellenmenge der i-ten Iteration
$\boldsymbol{I}$	Trägheitstensor
J_i	Anzahl der Stützstellen der i-ten Iteration
$\boldsymbol{J}_T, \boldsymbol{J}_R$	Jacobimatrizen der Translation und Rotation
k_c	Faktor der dynamischen Verhärtung
$\boldsymbol{k}$	Vektor der verallgemeinerten Zentrifugal-, Kreisel- und Corioliskräfte
K	Anzahl nicht dominierter Entwürfe
K_i	i-ter Körper eines Mehrkörpersystems
$\bar{K}$	Anzahl dominierter Entwürfe
$\boldsymbol{K}$	Steifigkeitsmatrix
$\boldsymbol{l}^l, \boldsymbol{l}^r$	eingeprägtes Moment und Reaktionsmoment
m	Masse
m_A	Masse des Aufbaus
m_F	Masse des Fahrschemels
m_R	Masse des Rads
M	Anzahl neuer Stützstellen pro Iteration
$\boldsymbol{M}$	Massenmatrix
N	Anzahl der Gütekriterien
O_I	Intertialsystem
p	Anzahl der Körper eines Mehrkörpersystems
p	Parameter, Entwurfsvariable
$\boldsymbol{p}, \boldsymbol{q}$	Vektor der Entwurfsvariablen
$\Delta\boldsymbol{p}$	Vektor der stochastischen Variablen
P	zulässiger Entwurfsraum
$\boldsymbol{q}$	Vektor der verallgemeinerten eingeprägten Kräfte
r_B	Bahnradius
$\boldsymbol{r}$	Vektor der stochastischen Variablen, Ortsvektor
s	Laplace-Faktor
$\boldsymbol{s}$	Vektor der Verdrehung
S	Anzahl der Segmente
$\boldsymbol{S}$	Drehmatrix
$t_{\dot{\psi}}$	Auswertezeitpunkt der Giergeschwindigkeit
w	innerer Freiheitsgrad des Federungssystems
$\boldsymbol{w}$	Koppelvektor des inneren Freiheitsgrads
W^r	Arbeit der Reaktionskräfte
$\boldsymbol{x}$	Zustandsvektor
$\hat{\boldsymbol{x}}_n$	Eigenvektor der n-ten Schwingmode
$\boldsymbol{y}$	verallgemeinerte Koordinaten
z_A	Freiheitsgrad des Aufbaus
z_F	Freiheitsgrad des Fahrschemels
z_R	Freiheitsgrad des Rads

Griechische Symbole

δ	mittlerer Spurwinkel
δ_r, δ_l	Spurwinkel des rechten und linken Rads
δ_A	Ackermannwinkel
δ_{EG}	Eigenlenkbedarf
δ_L	Lenkradwinkel

ε_f	Fehlerrate der Stützstellensuche
ε_{tol}	Fehlertoleranz
$\boldsymbol{\varepsilon}$	Approximationsfehler der Antwortflächen
λ_n	Eigenwert der n-ten Schwingmode
Λ	Lebesgue-Maß
μ_f	Erwartungswert der Gütefunktion f
$\boldsymbol{\mu}_r$	Vektor der Erwartungswerte von $\boldsymbol{r}$
σ_f	Standardabweichung der Gütefunktion f
$\boldsymbol{\Sigma}_r$	Kovarianzmatrix der Variablen $\boldsymbol{r}$
φ	Verlustwinkel
$\tilde{\varphi}$	Entwurfsverlustwinkel
ψ	Gierwinkel
ω	Kreisfrequenz
ω_a	spezifische Kreisfrequenz
$\boldsymbol{\omega}$	Winkelgeschwindigkeitsvektor

Mathematische Symbole

$\dot{a}$	Ableitung nach der Zeit da/dt
$\ddot{a}$	zweite Ableitung nach der Zeit d^2a/dt^2
δa	Lagevariation von a
$\mathbf{0}$	Nullmatrix, Nullvektor
$\mathcal{D}$	Menge dominierter Entwürfe
$\mathcal{L}\{a(t)\}$	Laplace-Transformation einer Funktion $a(t)$
$\mathcal{N}$	Menge nicht dominierter Entwürfe
$\mathcal{S}$	S-Metrik, Hypervolumen
$\boldsymbol{\mathcal{S}}$	Vektor der S-Metriken

Indizes

$\bullet^{(A)}$	architekturbezogene Variable
$\bullet^{(s)}$	segmentbezogene Variable
$\bullet^{(s,d)}$	derivatbezogene Variable eines Segments
$\bullet^{KV}$	Größen des Kelvin-Voigt-Modells
$\bullet_{MKS}$	Größen des Anwendungsbeispiels
$\bullet_P$	Pareto-optimale Lösung
$\bullet_{RDO}$	Größen der robusten Optimierung
$\bullet_{\mathcal{S}}$	S-Metrik-bezogene Größen
$\bullet_{SSO}$	Größen der Subraumoptimierung
$\bullet_u, \bullet_o$	untere und obere Schranke einer Entwurfsvariablen
$\bullet^*$	optimale Lösung
$\bullet^{**}$	utopische Lösung

Akronyme

ABS	Antiblockiersystem
ALHS	Advanced Latin Hypercube Sampling
ARSM	Adaptive Response Surface Modeling
ATC	Analytical Target Cascading
CFD	Computational Fluid Dynamics
CMF	Common Module Family
CSSO	Concurrent Subspace Optimization
DMU	Digital Mock-Up
DoE	Design of Experiments
EA	Evolutionäre Algorithmen
FEM	Finite-Elemente-Methode
GA	Genetische Algorithmen
KBA	Kraftfahrt-Bundesamt
KV	Kelvin-Voigt
LAPACK	Linear Algebra Package
LHS	Latin Hypercube Sampling
MAC	Model Assurance Criterion
MDO	Multidisciplinary Optimization
MFA	Mercedes Frontwheel Architecture
MHA	Mercedes High Architecture
MKS	Mehrkörpersimulation, -system
MLB	Modularer Längsbaukasten
MOP	Metamodel of Optimal Prognosis
MQB	Modularer Querbaukasten
MRA	Mercedes Rearwheel Architecture
MSA	Mercedes Sport Architecture
MSB	Modularer Standardantriebsbaukasten
NCBF	Non-Centroidal Body Fixed
PFD	Product and Family Design
Pkw	Personenkraftwagen

PSO	Particle Swarm Optimization
RA	Reliability Analysis
RDA	Robust Design Analysis
RDO	Robust Design Optimization
RMS	Root Mean Square
RS	Response Surface
RSM	Response Surface Modeling
SPEA2	Strength Pareto Evolutionary Algorithm
SUV	Sport Utility Vehicle

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

In der modernen Fahrzeugentwicklung haben computerbasierte und automatisierbare Entwicklungsmethoden einen immer größer werdenden Einfluss. Dieser Trend ist nicht zuletzt auf erhöhten Kostendruck, zu verkürzende Entwicklungszeiten, Fortschritte in der Simulationstechnik und gesteigerte Rechenleistung zurückzuführen. Voraussetzung dafür ist der Einsatz virtueller digitaler Prototypen, da so teure physikalische Prototypen eingespart, technische Unzulänglichkeiten des Produkts frühzeitig erkannt und ein insgesamt höherer Reifegrad des Produkts erreicht werden können.

Das Auslegungspotenzial eines digitalen Prototyps ist enorm, da durch Forschung und Entwicklung eine Vielzahl von Verfahren und Methoden verschiedener technischer Disziplinen entwickelt wurden. Ein Aspekt der Absicherung mithilfe digitaler Prototypen ist die kinematische und dynamische Analyse, wobei zumeist die Mehrkörpersimulation (MKS) eingesetzt wird. Mithilfe der MKS können kritische Systemzustände aufgedeckt und Maßnahmen zu deren Behebung untersucht, aber auch Funktionalität und Nutzbarkeit eines Produkts verbessert werden. Die Möglichkeit der Bewertung digitaler Prototypen ist insbesondere bei komplexen mechanischen Systemen sinnvoll. So werden in der Automobilbranche beispielsweise Fahrdynamik, Fahrkomfort und Fahrsicherheit von Personenkraftwagen (Pkw) bereits in der frühen Entwicklungsphase analysiert und Verbesserungspotenziale ausgewiesen.

Ein weiterer Trend in der Produktentwicklung und insbesondere in der Auto-

mobilentwicklung ist die Standardisierung. Das Ziel ist, mit möglichst geringem Investitionsvolumen und kurzer Entwicklungszeit viele verschiedene Modelle auf dem Markt zu platzieren. Um dies zu erreichen, müssen vermehrt vereinheitlichte Komponenten mit standardisierten Schnittstellen entwickelt und eingesetzt werden. Die sogenannten Plattformen, die dies ermöglichen, haben eine hohe Komplexität, da die Auslegung der gemeinsam nutzbaren Komponenten so gewählt werden muss, dass alle darauf basierenden Produkte die gesetzten Zielvorgaben erreichen.

An dieser Stelle können digitale Prototypen ihr volles Potenzial ausspielen, da sie die effiziente Untersuchung einer Vielzahl von Varianten erlauben. Die Folge einer geänderten Plattform kann so bereits im Vorfeld digital abgeschätzt werden. Bei einem komplexen Produkt wie einem Automobil ist es allerdings schwer vorherzusehen, wie sich die Änderung eines Fahrzeugs auswirken wird. Diese Problematik wird bei der Entwicklung einer Plattform für mehrere Fahrzeugmodelle noch verschärft. Je nach Art und Umfang einer Plattform gibt es Kopplungen und Abhängigkeiten zwischen plattformrelevanten und produktspezifischen Parametern, so dass kleine Änderungen der Plattform das Verhalten mehrerer Fahrzeuge maßgeblich beeinflussen können. Daher bedarf es Methoden, welche die Ingenieure bei der Entwicklung solcher hochdimensionalen und gekoppelten Probleme unterstützen.

Das Gebiet der numerischen Optimierung bietet automatisierte, computerbasierte und effiziente Methoden zur Lösung von Problemen dieser Art an. Diese Verfahren können hier aber nicht direkt eingesetzt werden, sondern es werden Methoden benötigt, die mit Fahrzeugplattformen und den damit verbundenen Besonderheiten umgehen können. Zudem müssen diese Methoden auf die in der Entwicklung eingesetzte Softwareinfrastruktur adaptiert und digitale parametrische Fahrzeugmodelle erstellt werden, um Fahrzeugvarianten automatisiert simulieren zu können. Unter Berücksichtigung der Struktur einer Plattform bei digitalen Prototypen und Optimierungsprozessen kann mithilfe der Simulation, insbesondere in der frühen Entwicklungsphase, eine effiziente Plattformentwicklung realisiert werden.

1.1 Architekturen in der Fahrzeugentwicklung

Produktplattformen werden heutzutage in verschiedenen Bereichen der produzierenden Industrie eingesetzt. Die Entwicklung von Produktplattformen, die in dieser Arbeit kurz als Plattformen bezeichnet werden, wurde von Unternehmen vorangetrieben, die eine größere Produktvielfalt forcierten, um auf individuelle Kundenwünsche eingehen zu können, aber zur Erhaltung der Wettbewerbsfähigkeit auf eine effiziente industrielle Massenproduktion nicht verzichten wollten. Diese Kombination aus variantenreichen Produkten und Massenproduktion wird nach Pine [60] als kundenindividuelle Massenproduktion bezeichnet, wobei die Automobilbranche die Pionierarbeit auf diesem Gebiet leistete [96].

Im frühen 20. Jahrhundert war Ford der erste Automobilhersteller, der die Massenfertigung in Form der Fließbandfertigung einsetzte. Allerdings gab es damals nur ein einheitliches Fahrzeugmodell, so dass keine individuellen Kundenwünsche erfüllt werden konnten. In den darauffolgenden Jahren boten Hersteller zunächst verschiedene Lackierungen und später auch optionale Ausstattungen an, wodurch verschiedene Varianten eines Fahrzeugmodells entstanden. Um weitere Kunden anzusprechen, wurden sich in Größe und Design voneinander unterscheidende Modelle entwickelt. Angetrieben durch den natürlichen Wettbewerb entwickelte und verbreitete sich die kundenindividuelle Massenproduktion unter den Herstellern [96].

Heutzutage wird gefordert, dass Pkw-Hersteller flexibel genug sind, um den individuellen Wünschen der Kunden in verschiedenen Teilen der Welt gerecht zu werden. Damit die große Anzahl von Fahrzeugvarianten ohne signifikante Mehrkosten realisiert werden kann, müssen Bauteile, Funktionsgruppen und deren Schnittstellen vereinheitlicht bzw. modularisiert werden [96]. Das Ziel ist, mit möglichst wenig Teilen eine möglichst große Produktdiversität zu erreichen. Neben der günstigen Herstellung vieler Fahrzeugvarianten bietet der Einsatz einer Plattform zusätzlich den Vorteil, schnell auf sich ändernde Marktanforderungen reagieren und in kurzer Zeit ein neues Produktportfolio entwickeln zu können [46].

In der jüngeren Vergangenheit ist die Anzahl der am Markt angebotenen Modelle deutlich gestiegen. Im Wettbewerb um neue Kunden versuchen die Hersteller, jede mögliche Marktnische mit neuen Modellen zu besetzen. Da Plattformen bereits etabliert sind, wird versucht, den erhöhten Entwicklungsaufwand durch eine Verringerung der Anzahl von Plattformen zu kompensieren. Das heißt, dass immer mehr Fahrzeugmodelle auf immer weniger Plattformen entwickelt werden. Dieser Trend lässt sich anhand von Analysen verschiedener Marktforschungsinstitute [62, 32, 37, 35] belegen. In Abb. 1.1 ist zu sehen, dass die Anzahl der angebotenen Fahrzeugmodelle in den letzten Jahren deutlich gestiegen ist. Dagegen sank die Anzahl der Plattformen und wird, laut der Prognosen der Institute, auch in Zukunft weiter sinken. Dieser Trend verdeutlicht, welch hohen Stellenwert Fahrzeughersteller Plattformen bei zukünftigen Fahrzeugentwicklungen zuschreiben.

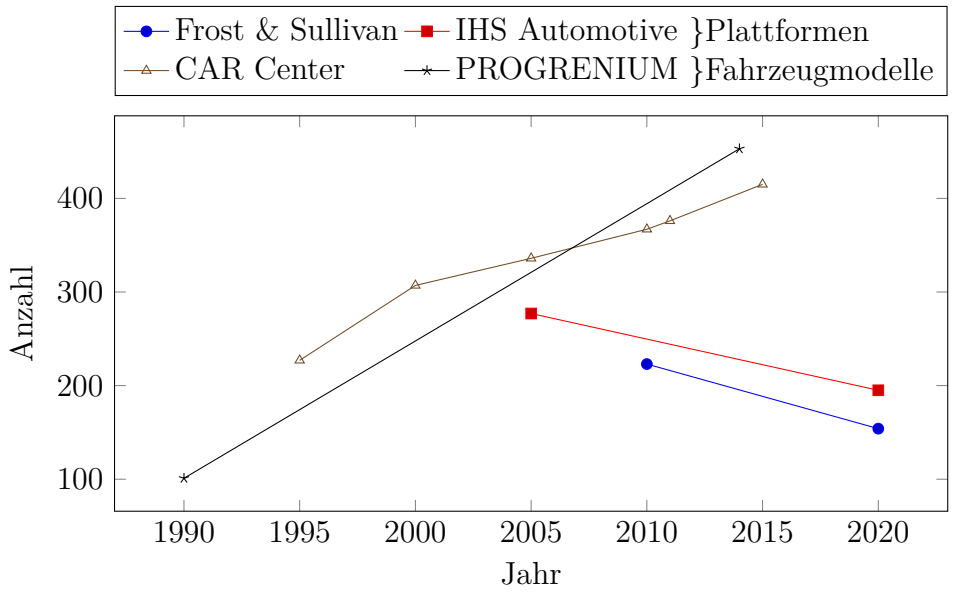

Abbildung 1.1: Analysen verschiedener Marktforschungsinstitute zur Entwicklung der Anzahl von Plattformen und Fahrzeugmodellen

Einer der führenden Hersteller, die Plattformen einsetzen, ist der Volkswagen Konzern. Bei Volkswagen werden Komponenten wie Motoren, Getriebe, Bremsen, Sitze und Achsen unter Fahrzeugmodellen der verschiedenen Konzernmarken geteilt [22]. Im Jahr 2012 ist Volkswagen mit der Einführung des Modularen Querbaukastens (MQB) noch einen Schritt weiter gegangen. Der

MQB kann in Fahrzeugen mit variablen Abmessungen, Radständen, Spurweiten, Rädergrößen und Sitzpositionen eingesetzt werden. Der Konzern setzt dabei jeweils auf Fahrzeugtyp und Antriebsvariante angepasste Plattformen ein. So gibt es neben dem MQB für frontgetriebene Pkw mit quer eingebautem Motor den Modularen Längsbaukasten (MLB) für Fahrzeuge mit längs eingebauten Motoren sowie den Modularen Standardantriebsbaukasten (MSB) für Modelle mit Standardantrieb [82]. Die Allianz von Renault und Nissan setzt ebenfalls Plattformen für unterschiedlich große Fahrzeuge ein. Die als Common Module Family (CMF) bezeichnete Plattform unterteilt sich in CMF-A für Kleinwagen, CMF-B für Mittelklassemodelle und CMF-C/D für große Fahrzeuge [43]. Daimler setzt eine ähnliche Strategie wie Volkswagen ein; so gibt es die Mercedes Frontwheel Architecture (MFA) für frontgetriebene Pkw, die Mercedes Rearwheel Architecture (MRA) für Fahrzeuge mit Standardantrieb, die Mercedes High Architecture (MHA) für Sport Utility Vehicle (SUV) und die Mercedes Sport Architecture (MSA) für Sportwagen [41]. Die genannten Hersteller sind nur einige Beispiele für Plattform-einsetzende Firmen. Im Prinzip nutzt jeder Automobilhersteller, wenn auch in unterschiedlicher Ausprägung und Komplexität, Plattformen. Unterschiede lassen sich bei Betrachtung der Plattformbezeichnungen der verschiedenen Hersteller erkennen. So fallen die unterschiedlichen Begriffe wie *Architektur*, *Familie*, *Baukasten* und *Modul* auf. Für eine präzise Beschreibung von Typ und Struktur verschiedener Plattformen ist daher eine klare Abgrenzung und Definition der genannten Begriffe notwendig.

In der Literatur haben sich im Rahmen vieler Forschungsaktivitäten diverse Definitionen einer Plattform herausgebildet. Baldwin und Woodard [4] haben festgestellt, dass diese disziplinübergreifend nur schwer in Einklang zu bringen sind. Allerdings gibt es einige wenige Gemeinsamkeiten, die die Bedeutung einer Plattform im Kern beschreiben und mit den bereits diskutierten Einsatzzwecken korrelieren. Ihnen zufolge ist eine Plattform

> „[...] the conservation or reuse of a core component to achieve economies of scale while reducing the cost of creating a wide variety of complementary components.“ [4, S. 3]

Plattformen werden also hauptsächlich aus wirtschaftlichen Gründen eingesetzt.

Dennoch weisen sie einige wichtige Eigenschaften auf, die für die technische Auslegung, auf der der Fokus dieser Arbeit liegt, wichtig sind. Nach Meyer und Lehnerd [50] lautet die aus technischer Sicht formulierte und auf das Produkt bezogene Definition einer Plattform:

> „A product platform is a set of common components, modules, or parts from which a stream of derivative products can be efficiently created and launched.“ [50, S. 7]

Die Autoren beziehen sich in ihrer Definition auf den bereits diskutierten Einsatz von Gleichteilen, verwenden aber auch den Begriff Modul, der ebenfalls in den oben genannten Plattformbezeichnungen auftritt. Für die Erläuterung eines Moduls muss zwischen unterschiedlichen Typen von Plattformen differenziert werden. Simpson et al. [80] unterscheiden zwischen skalierbaren und modularen Plattformen.

Skalierbare Plattformen werden typischerweise in Produkten eingesetzt, die die gleiche Funktion aufweisen, sich aber in Bezug auf Größe oder Leistung unterscheiden. Ein oft verwendetes Beispiel aus der Industrie ist eine von Black&Decker entwickelte Familie von universellen Elektromotoren. Hierbei können verschieden starke Elektromotoren über variable Ankerblechstapellängen hergestellt werden, die sich im übrigen Aufbau, wie beispielsweise dem Gehäusedurchmesser, gleichen [49]. Weitere Beispiele für skalierbare Plattformen sind Boeing, die verschieden große Flugzeuge über das Stretchen des Rumpfes realisieren [67], und Rolls Royce, die Triebwerke skalieren, um diese mit unterschiedlichen Schüben anbieten zu können [66].

Produkte einer modularen Plattform werden durch Austausch, Hinzufügen oder Entfernen von funktionalen Modulen variiert [80]. Ein Computer besitzt beispielsweise standardisierte Stecker und Steckplätze, über die unterschiedliche Prozessoren, Grafik-, Sound- und Netzwerkkarten angebunden werden. Auf diese Weise können auf individuelle Vorlieben oder spezifische Anwendungsgebiete angepasste Computer angeboten werden. Ein Modul hat in diesem Sinne eine oder mehrere Funktionen und eine standardisierte Schnittstelle, mit der es mit einer Plattform verbunden wird. Automobilhersteller setzen fast ausschließlich modulare Plattformen ein. So dienen die oben genannten Komponenten wie

Motoren, Getriebe, Bremsen und Sitze als Module, die in verschiedenen Leistungsstufen oder Ausprägungsformen in allen auf einer Plattform basierenden Fahrzeugen eingesetzt werden können.

Eine gesonderte Betrachtung erfordern Fahrzeuge, bei denen Teile der Radaufhängung zur Plattform gehören. In diesem Fall kann einem Modul oder einer Schnittstelle keine eigenständige Funktion zugewiesen werden, da diese erst in Verbindung miteinander eine bestimmte Funktion erfüllen und somit stark voneinander abhängig sind. Hier liegt der Fokus auf der Schnittstelle und weniger auf dem damit verbundenen Bauteil. Die Anordnung und Ausprägung von Schnittstellen innerhalb einer Plattform werden durch die Architektur beschrieben, wobei es modulare und integrale Architekturen gibt [80].

Beim beschriebenen Beispiel eines Computers kommt eine modulare Architektur zum Einsatz, da die Schnittstelle nur eine Funktion hat, nämlich das Aufnehmen eines Moduls. Verschiedene Module mit jeweils unterschiedlichen Funktionen werden über die Schnittstelle angebunden und können beliebig ausgetauscht werden. Göpfert [34] spricht in diesem Zusammenhang von einer funktionalen Unabhängigkeit der Komponenten. Bei einer integralen Architektur kann eine feste Zuordnung der Funktionen nicht erfolgen. Bei ihr spielt das Zusammenwirken von Schnittstellen und damit verbundenen Komponenten eine wichtige Rolle. Die Schnittstellen einer integralen Architektur dürfen deshalb nie unabhängig von den zugehörigen Komponenten entwickelt werden, da Funktionsweise und Qualität eines Produkts so nicht gewährgeleistet werden können. Im Gegensatz zur modularen Architektur wird hier laut Göpfert [34] von einer funktionalen Abhängigkeit der Komponenten gesprochen. Die Architektur bildet demzufolge die grundlegende Struktur, auf die Plattform, Module und Komponenten angepasst werden. Ihre Auslegung entscheidet über den Grad der Flexibilität, mit der darauf basierende Fahrzeuge entwickelt werden können.

Alle Produkte, die auf einer Plattform basieren werden unter dem Begriff Produktfamilie zusammengefasst. Simpson et al. [80] beschreiben eine Produktfamilie als

> „[...] a group of related products that is derived from a product platform to satisfy a variety of market niches.“ [80, S. 3]

Nach Einführung der Grundbegriffe der Plattformentwicklung wird klar, dass die verschiedenen, oben genannten Plattformbezeichnungen im Grunde das Gleiche meinen. Das Ziel von Daimler ist es, mithilfe der Architekturen eine möglichst große und heterogene Produktfamilie zu entwickeln, während Renault-Nissans Common Module Family auf eine einheitliche und modulare Architektur rückschließen lässt. Dass auch Volkswagen eine ähnliche Struktur einsetzt, ist zu vermuten, allerdings wurde der Begriff *Baukasten* bisher nicht erläutert. Hierfür bedarf es einer genaueren Betrachtung der Strategien, mit denen Plattformen in der Automobilbranche eingesetzt werden.

Wie bereits erwähnt, spielt es eine wichtige Rolle, ob in einer Plattform eine modulare oder eine integrale Architektur zum Einsatz kommt. Beide Formen sind in der modernen Automobilentwicklung präsent. Welcher Typ einer Architektur eingesetzt wird, hängt von mehreren Aspekten ab. Mit die wichtigsten Aspekte sind die Größe und Heterogenität der zu entwickelnden Fahrzeugfamilie. Im Fahrzeugbau erstrecken sich Familien über wenige Derivate bis hin zu mehreren Segmenten mit wiederum mehreren Derivaten.

Segmente und Derivate werden zur Kategorisierung von Fahrzeugmodellen verwendet und bilden innerhalb einer Familie eine hierarchische Struktur. Einem Fahrzeugsegment, auch als Fahrzeugklasse bezeichnet, werden Fahrzeugmodelle vom Kraftfahrt-Bundesamt (KBA) „[...] anhand optischer, technischer und marktorientierter Merkmale [...]“ zugeordnet [47]. Das KBA unterscheidet hierbei zwischen 12 Segmenten: Minis, Kleinwagen, Kompaktklasse, Mittelklasse, Obere Mittelklasse, Oberklasse, SUVs, Geländewagen, Sportwagen, Vans, Utilities und Wohnmobile [48]. Eine exakte Beschreibung der Merkmale, nach denen ein Fahrzeug einem Segment zugeordnet wird, gibt es nicht. Typischerweise unterscheiden sich Fahrzeuge in verschiedenen Segmenten durch Länge, Höhe, Radstand, Gewicht und Geländegängigkeit. Laut Heißing et al. [38] sind zudem Komfortmaße wie Schulterbreite vorne, Fußraum hinten, Innenraumlänge (Komfortmaß) und Kofferraumvolumen wichtige Unterscheidungsmerkmale.

Innerhalb eines Segments kann es verschiedene Derivate eines Herstellers geben. Üblicherweise ist ein Derivat ein Fahrzeugmodell, das sich nur geringfügig von einem bereits existierenden Modell unterscheidet, weshalb beide dem gleichen Segment zugeordnet werden. Die ähnlichen Merkmale kommen nicht von ungefähr; so verwenden Hersteller für verschiedene Derivate zumeist die gleiche technische Basis. Damit wird deutlich, dass die Produktfamilie eines Segments mit zwei Derivaten bereits auf einer gemeinsamen Plattform aufbaut, nämlich dieser technischen Basis. Dieses Vorgehen wird gemeinhin als Plattformstrategie bezeichnet und stellt die einfachste, aber auch unflexibelste Strategie dar. Die Plattform umfasst hierbei das gesamte Chassis, wodurch alle Derivate einheitliche Abmaße, wie Radstand und Spurweite, aufweisen. Die eigenständigen Derivate werden durch das Aufsetzen von Hüten, die den vom Kunden sichtbaren und fühlbaren Teil darstellen, komplettiert [21]. Es wird deshalb auch von einer starren Architektur gesprochen, bei der Synergieeffekte zumeist nur innerhalb eines Fahrzeugsegments genutzt werden [63]. Ein Beispiel für eine starre Architektur ist die MFA Plattform von Daimler, deren Struktur in Abb. 1.2 dargestellt ist. Normalweise teilt das KBA die Derivate dieser Architektur in unterschiedliche Segmente ein. Zur Verdeutlichung der Einschränkungen einer starren Architektur werden diese in Abb. 1.2 aber dem selben Segment zugeordnet. Gedanklich handelt es sich beim GLA aufgrund seiner Größe um ein *Kompaktklasse-SUV*, bei der B-Klasse um einen *Kompaktklasse-VAN*.

Laut Volkswagen ist die nächste Komplexitätsstufe die Modulstrategie [63]. Bei der Modulstrategie können Synergien teilweise über Segmente hinweg genutzt werden, da wie bereits erwähnt Module wie Sitze und Bremsen in Derivaten verschiedener Segmente eingesetzt werden. Sie ist weitestgehend unabhängig von der zugrunde liegenden Architektur, sofern hinreichend definierte Schnittstellen für den Austausch von Modulen vorhanden sind. Der Stand der Technik ist die sogenannte Baukastenstrategie, mit der Synergieeffekte vollständig über Segmente hinweg genutzt werden können [63]. Hierbei sind Abmessungen wie Länge, Breite und Höhe, aber auch der Radstand, die Spur und die Rädergröße, variabel. Bedient die Produktfamilie mehrere Segmente, so ist eine große Spreizung von Radständen und Spurweiten notwendig. Damit in Bezug auf Fahrkomfort und Fahrdynamik keine zu großen Kompromisse eingegangen

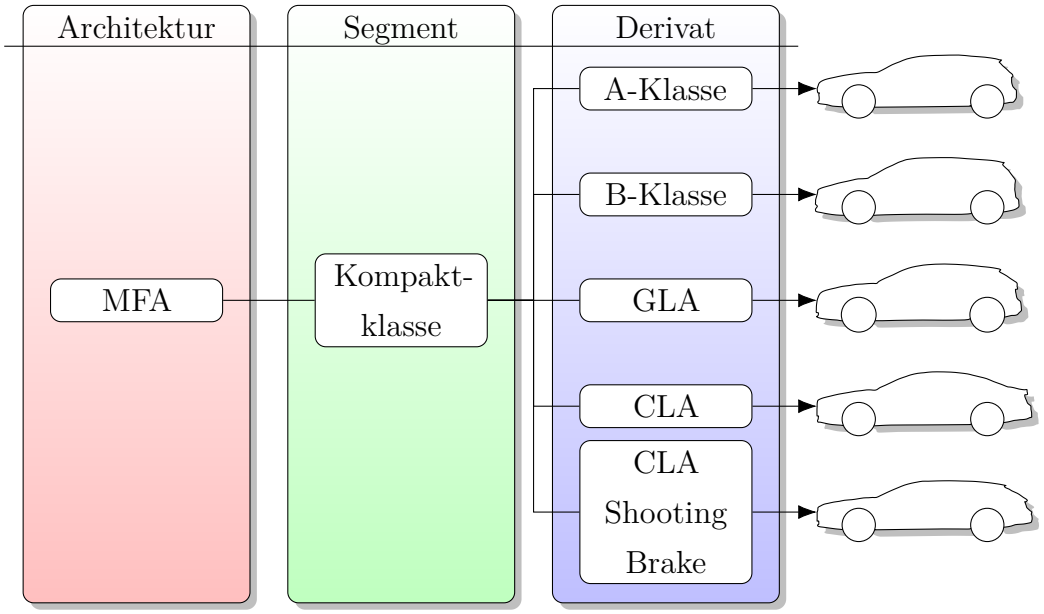

Abbildung 1.2: Mercedes Frontwheel Architecture (MFA)

werden müssen, sind Teile der Fahrwerksarchitektur variabel gestaltbar. In diesem Zusammenhang wird von einer flexiblen Architektur gesprochen, wofür Daimlers MRA in Abb. 1.3 ein Beispiel ist. Hier enthalten Derivate verschiedener Segmente zu großen Teilen einheitliche Komponenten und Module.

Um Plattformen in der frühen Entwicklungsphase effizient auszulegen, ergibt sich die Notwendigkeit, automatisierte Optimierungsalgorithmen einzusetzen. Die Stärke dieser Verfahren liegt dort, wo der Mensch sich üblicherweise schwer tut, nämlich beim Lösen von Problemen mit einer Vielzahl von Parametern, die sich zusätzlich gegenseitig beeinflussen können. Im nächsten Kapitel wird daher ein Überblick über die numerische Optimierung mit dem Fokus auf der sogenannten Plattform- bzw. Architekturoptimierung gegeben.

1.2 Stand der Technik in der Architekturoptimierung

In der Vergangenheit wurde im Ingenieurwesen die optimale Lösung eines Problems durch einen intuitiven und auf Erfahrungen basierenden Prozess mithilfe von physikalischen Prototypen erreicht. Dieser Prozess besteht im

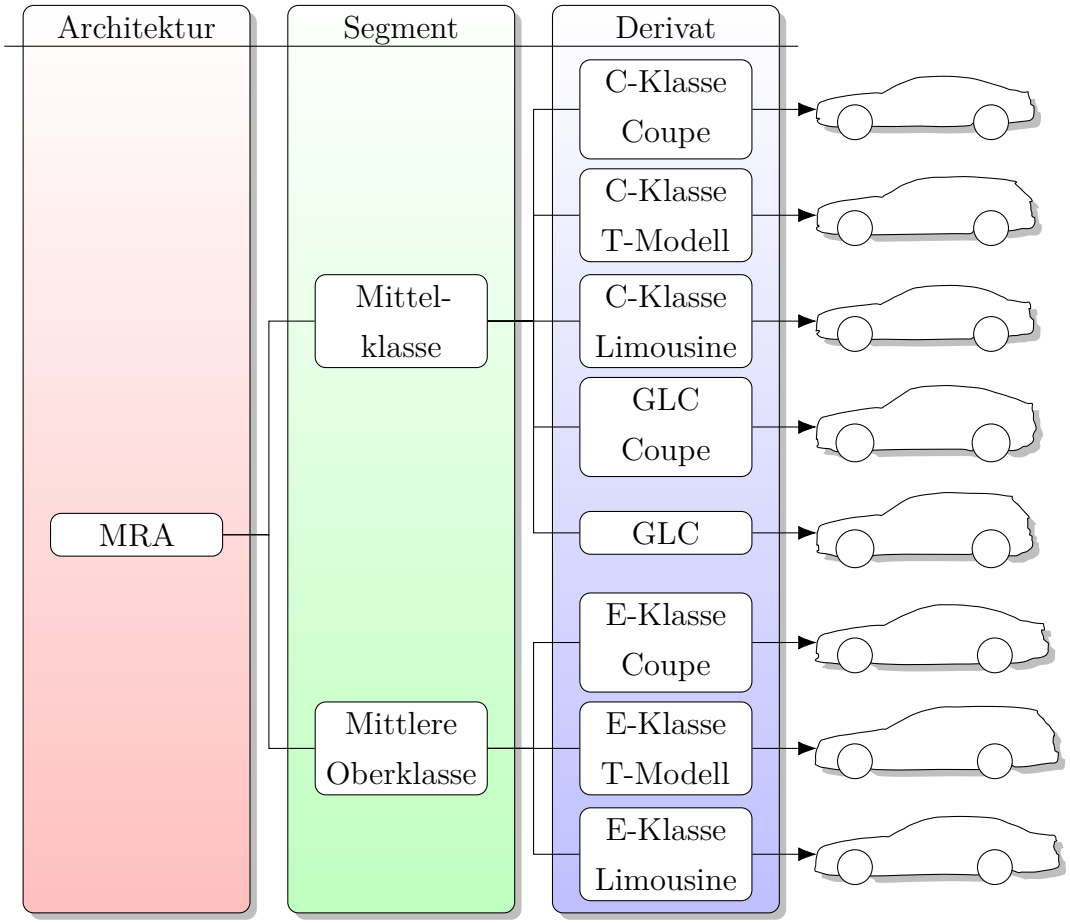

Abbildung 1.3: Mercedes Rearwheel Architecture (MRA)

Wesentlichen aus dem Variieren eines oder mehrerer Parameter und dem entsprechenden Anpassen des Prototypen sowie der Dokumentation des daraus resultierenden Verhaltens. Im modernen Ingenieurwesen wird dieser Prozess zu großen Teilen mithilfe rechnerischer Verfahren durchgeführt, wodurch Kosten und Zeitaufwand reduziert werden [5]. Dieser Prozess kann unter dem Oberbegriff der Optimierung zusammengefasst werden, wobei sich dieser zumeist auf rechnerische Verfahren bezieht. Das Gebiet der Optimierung ist ein sehr großes, weshalb hier nur ein oberflächlicher Überblick über Verfahren und Methoden gegeben werden kann. Der Fokus dieses Kapitels liegt auf dem für diese Arbeit wichtigen Gebiet der Plattform- und Familienoptimierung, auf die im vorderen Teil eingegangen wird. Zur Erläuterung wichtiger Begriffe folgt im hinteren Teil eine Betrachtung der Optimierung im Allgemeinen.

Das Forschungsgebiet Product and Family Design (PFD) befasst sich mit Methoden zur systematischen Auslegung von Produkt-Plattformen. Das in Abschnitt 1.1 beschriebene Interesse der Industrie hat in den vergangenen Jahren zu wachsenden Forschungsaktivitäten und einer Vielzahl von Veröffentlichungen geführt. Einleitend sind die Bücher und Studien von Simpson et al. [79, 81] und Jiao et al. [40] zu erwähnen, die einen umfassenden Überblick über aktuelle Entwicklungen und den Stand der Technik geben. Wie Schuh et al. [71] feststellen, gibt es Methoden zur Plattformentwicklung aus technischer und organisatorischer Perspektive. Diese Arbeit befasst sich mit computerbasierten automatisierten Methoden, weshalb der Zweig der organisatorischen Methoden nicht weiter betrachtet wird.

PFD-Strategien verwenden Algorithmen und Verfahren aus den Teilgebieten der Optimierung, auf die später genauer eingegangen wird. Diese werden adaptiert und kombiniert, um Plattform-Optimierungsprobleme auf effiziente Weise zu lösen. Pirmoradi et al. [61] unterscheiden zwischen drei unterschiedlichen Klassen von Plattform-Optimierungsproblemen:

1. Aus einer Menge vordefinierter Module werden diejenigen ausgewählt, mit denen daraus entstehende Produkte einer Plattform die gesetzten Anforderungen bestmöglich erreichen;

2. Bei bekannter Topologie der Architektur werden optimale Entwurfsvariablen von Plattform, Architektur und Produkten bestimmt;

3. Die Festlegung der Topologie der Architektur und die Optimierung der Plattform und den darauf basierenden Produkten wird zeitgleich vorgenommen.

Für Probleme der letzten beiden Klassen, die zur Zeit das größere Forschungsinteresse genießen, haben sich zwei grundlegende Herangehensweisen entwickelt: einstufige und zweistufige [77]. Bei einstufigen Verfahren werden Plattform, Architektur und Produktfamilie in einem Optimierungskreislauf gleichzeitig optimiert, bei zweistufigen werden Plattform und der plattformrelevante Teil der Architektur sowie die Produktfamilie und der produktrelevante Teil der Architektur sequenziell in separaten Kreisläufen optimiert. Simpson [77] hat

in seinen Untersuchungen festgestellt, dass einstufige Verfahren bessere Ergebnisse liefern, aber aufgrund der höheren Dimension des Optimierungsproblems schwerer zu lösen sind. Ein weiteres, a priori definierte Architekturtopologien betreffendes Verfahren ist das von Kokkolaras et al. [46] erweiterte Analytical Target Cascading (ATC), das ursprünglich von Kim et al. [45] vorgestellt wurde. Hierbei wird das Optimierungsproblem in mehrere Ebenen unterteilt, die System, Subsystem und Komponenten separat betrachten. Die Optimierungsziele der obersten Ebene werden an alle untergeordneten Ebenen vererbt, wobei die unteren Subsystem- und Komponentenebenen zusätzlich eigene Ziele haben können. Dies bedeutet, dass jede Optimierung der unteren Ebenen ihre Kriterien verbessert, während die Ziele der übergeordneten Ebenen erfüllt werden müssen. Die Strategie erzielt gute Ergebnisse bei Problemen mit vielen Variablen, da das gegebene Optimierungsproblem durch die hierarchische Struktur in mehrere Optimierungsprobleme mit jeweils geringerer Dimension unterteilt wird. Zu beachten ist, dass aufgrund der Vererbung der Ziele insgesamt mehr Variablen und Kriterien benötigt werden. Zusätzlich verliert die Strategie bei gekoppelten und voneinander abhängigen Systemen und Komponenten an Funktionalität.

Bei Problemen der dritten Klasse werden häufig zusätzliche Parameter eingeführt, die steuern, ob eine Entwurfsvariable produktspezifisch ist oder vereinheitlicht für die gesamte Plattform gilt. Typischerweise werden mehrkriterielle Probleme zur Bestimmung des besten Kompromisses zwischen Performanz und Vereinheitlichung gelöst. Fellini et al. [27] transformieren ein mehrkriterielles Problem auf eine Sequenz von skalaren Optimierungsproblemen und setzen eine auf einem einkriteriellen Algorithmus basierende mehrschrittige Methode ein, die zusätzliche Randbedingungen und Referenzsimulationen benötigt, um eine Plattform zu optimieren. Tarkian et al. [84] optimieren eine Familie von Industrierobotern unter Verwendung eines mehrkriteriellen genetischen Algorithmus (GA), wobei nur diskrete Entwurfsvariablen betrachtet werden, um die Bestimmung des Grades der Vereinheitlichung zu vereinfachen. Neuere Methoden steuern den Grad der Vereinheitlichung über Gene innerhalb der Chromosomen genetischer Algorithmen, wie bei Chen und Wang [18] oder Chen und Martinez [17]. Friedrich und Menzel [31] verwenden eine kaskadierte Strategie, wobei die Plattform in einem äußeren Optimierungskreislauf und die

Produkte der Produktfamilie in einem inneren jeweils mehrkriteriell optimiert werden. Einen ähnlichen kaskadierten Ansatz verfolgt Wang [91], wobei seine Strategie einen übergeordneten GA einsetzt, um die Gene zur Bestimmung der Vereinheitlichung zu variieren. Für jedes so generierte Individuum wird anschließend eine Particle Swarm Optimization (PSO) durchgeführt, um die optimalen Werte der Entwurfsvariablen zu bestimmen.

Strategien für robuste Plattformoptimierung werden in der Literatur bislang wenig betrachtet. Moon und McAdams [52] betrachten die Entwicklung einer Plattform bei unsicheren Marktverhältnissen. Simpson et al. [78] nutzen Ansätze aus der robusten Optimierung für die Auslegung einer Familie von Flugzeugen. Hierbei liegt aber nicht die Robustheit des Produkts gegenüber Unsicherheiten im Fokus, sondern eine einfachere Formulierung des Plattformoptimierungsproblems. Die Anzahl der Passagiere wird als streuender Parameter betrachtet und Ziele für Mittelwert und Standardabweichung von Kriterien, wie dem Tankinhalt, definiert, um unterschiedlich große, auf einer Plattform basierende Flugzeuge zu optimieren.

Im Allgemeinen kann die Optimierung nach Cavazzuti [16] in drei Teilgebiete unterteilt werden:

- Design of Experiments,
- Optimierungsalgorithmen und
- Robust Design Analysis.

Diese Teilgebiete nutzen teilweise Inhalte der anderen, haben aber unterschiedliche Ziele und Herangehensweisen.

Design of Experiments (DoE), auf deutsch Versuchspläne, wurden bereits in den 1920er Jahren entwickelt, wobei Fisher [28] 1935 das erste, die Thematik behandelnde Fachbuch herausbrachte. Weitere wichtige Meilensteine der DoE sind die Arbeit von Box und Wilson [10], die die industrielle Anwendung forcierten und das Response Surface Modeling (RSM) entwickelten, sowie Taguchi [83], der in den 80er Jahren die industrielle Entwicklung auf Basis von DoE

populär machte und den daraus resultierenden Gewinn von Entwicklungsqualität aufzeigte [75]. Das Ziel einer DoE ist es, eine Anzahl Parametersätze, auch Stichproben genannt, zu generieren, mit denen ein Maximum an Informationen der zu suchenden und die Antwortgrößen beschreibenden Funktionen gefunden werden kann [16]. Mithilfe dieser Informationen können verschiedene Strategien, wie beispielsweise Korrelationsanalysen zur Bestimmung der Sensitivitäten von Entwurfsparametern bezüglich gewählter Gütekriterien, bedient werden. Ist eine anschließende Optimierung des Systems geplant, können die Antwortgrößen innerhalb des definierten Entwurfsraums mit RSM-Verfahren approximiert oder der Entwurfsraum für die Optimierung auf das Wesentliche reduziert werden. Zu guter Letzt bietet ein DoE die Möglichkeit, aussichtsreiche Parametersätze des Entwurfsraums als Startlösungen für eine Optimierung zu verwenden, um deren Konvergenz zu beschleunigen [9]. In den vergangenen Jahrzehnten wurde eine Vielzahl von DoE-Verfahren entwickelt, wobei das Latin Hypercube Sampling (LHS) eine der verbreitetsten Methoden ist. Das in dieser Arbeit verwendete LHS wird in Abschnitt 4.3 beschrieben, für einen umfassenderen Überblick sei auf die Bücher von Siebertz [75] und Cavazzuti [16] verwiesen.

Das Teilgebiet der Optimierungsalgorithmen bildet den wichtigsten und größten Teil der Optimierung. Grundsätzlich können diese in verschiedene Gruppen unterteilt werden, hier allerdings wird eine Beschränkung auf deterministische und stochastische Algorithmen vorgenommen. Deterministische Algorithmen basieren zumeist auf linearer Algebra, wobei sie zum Teil Gradienten der zu optimierenden Funktion benötigen. Ein Vorteil dieser Algorithmen ist ihre Effizienz. Sie sind aber nicht zu empfehlen, wenn das globale Optimum einer Funktion gefordert wird, die Funktion unstetig ist oder mehrere Kriterien gleichzeitig zu optimieren sind [16].

Der NLPQLP Algorithmus von Schittkowski [69] für glatte, nichtlineare und beschränkte Optimierungsprobleme kann aufgrund seiner Effizienz und Zuverlässigkeit zum Stand der Technik gezählt werden. Für unbeschränkte nichtlineare Optimierungsprobleme sind die heuristische Simplex Methode von Nelder und Mead [55] und die zu den quasi-Newton Methoden gehörende und nach ihren Entwicklern benannte Broyden-Fletcher-Goldfarb-Shanno Methode

[13, 12, 29, 33, 74] zu erwähnen. Ausführliche Beschreibungen und Diskussionen von determinstischen Algorithmen mit weiterführenden Literaturhinweisen und Praxisbezug gibt es in [16] und [56].

Stochastische Algorithmen verwenden entsprechend ihrer Namensgebung Zufallsprozesse, um eine Funktion zu optimieren. Sie sind dadurch in der Lage, globale Optima zu finden, und können für mehrkriterielle Optimierungsprobleme eingesetzt werden, konvergieren aber langsamer als deterministische Algorithmen. Die bekanntesten und am weitesten verbreitetsten stochastischen Algorithmen sind die von der Natur inspirierten GA bzw. evolutionären Algorithmen (EA) sowie die schwarmbasierten Algorithmen aus dem Bereich der PSO. Das anhaltend große Interesse an diesen Algorithmen hat zu einer Vielzahl an Veröffentlichungen und Entwicklungen geführt, die hier nicht in hinreichender Weise zusammengefasst und diskutiert werden können. Weiterführende Literaturempfehlungen sind die Werke von Weise [93], Siebertz [75] und Cavazutti [16].

Ziel der Robust Design Analysis (RDA) ist die Bestimmung eines robusten oder ausfallsicheren Optimums bei unsicheren System- und Umgebungsvariablen. Hierbei muss zwischen zwei Teilgebieten, der Robust Design Optimization (RDO) und der Reliability Analysis (RA), unterschieden werden. Letztere, auf deutsch Zuverlässigkeitsanalyse, bietet Methoden und Verfahren, um die Wahrscheinlichkeit des Nicht-Erreichens eines vordefinierten Kriteriums, die sogenannte Ausfallwahrscheinlichkeit, zu berechnen. Typischerweise wird ein aus einer vorangegangenen Optimierung bestimmtes System mittels einer RA untersucht [16].

Eine RDO optimiert Mittelwert und Varianz eines mit Unsicherheit behafteten Systems direkt. Die Unsicherheit wird durch Parameterkombinationen dargestellt, die mit vordefinierten Verteilungsfunktionen mittels oben genannter DoE-Verfahren generiert werden. Das Ergebnis einer solchen Optimierung wird als robustes Optimum bezeichnet, das in der Regel weniger leistungsfähig ist als ein deterministisches Optimum, dafür aber eine geringere Anfälligkeit gegen Störungen aufweist [16]. Eine umfassende Studie von Beyer und Sendhoff [8] behandelt Historie, Methoden und Verfahren der RDA genauer.

1.3 Inhalt und Ziel der Arbeit

Das Ziel dieser Arbeit ist die Entwicklung eines Konzepts für die effiziente robuste Optimierung einer Pkw-Fahrwerksarchitektur. Bei den betrachteten Plattform-Optimierungsproblemen handelt es sich um Probleme der oben erläuterten zweiten Klasse mit a priori definierten Topologien.

Ein wichtiger Beitrag der Arbeit bezieht sich auf die Aussage von Simpson [77], nämlich dass Architektur- oder Plattformoptimierungen bisher wenig oder gar keinen Einsatz in der Industrie finden. Ein Teilziel ist die Anwendbarkeit des entwickelten Konzepts in Bezug auf die Optimierung komplexer MKS-Fahrzeugmodelle, wie sie in der Industrie entwicklungsbegleitend eingesetzt werden. Anwendungen im Fahrzeugbereich sind zumeist aus wirtschaftlichen und organisatorischen Bereichen. Ein Beispiel für eine technische Anwendung ist die von Fellini et al. [26], in der detaillierte Karosseriemodelle von Fahrzeugen optimiert werden. Schuh et al. [71] entwickeln eine Fahrwerksarchitektur für Elektrofahrzeuge aus Produktionssicht. MKS-einsetzende, Pkw-Fahrwerke betreffende Anwendungen von Plattformoptimierungen sind allerdings nicht bekannt.

Wie bereits erwähnt, handelt es sich bei einer Fahrwerksarchitektur um eine integrale Architektur, die verschiedenen Entwurfsvariablen einer Optimierung sind also voneinander abhängig. Das in Abschnitt 1.2 erläuterte ATC ist aufgrund der Vererbung von Zielbereichen in unteren Ebenen nicht für Probleme dieser Art geeignet. Es bedarf daher eines anderen Konzepts für die Optimierung von integralen Architekturen.

Das letzte Teilziel ist die robuste Optimierung einer Architektur. Dieser Bereich wird zukünftig an Bedeutung gewinnen, ist aber noch nicht im breiten Interesse der Forschung. Khire et al. [44] haben dies und die Vorteile einer robusten Plattformoptimierung erkannt und sehen die Notwendigkeit, RDO-Methoden mit PFD-Strategien zu kombinieren.

Der Inhalt der Arbeit gliedert sich gemäß dieser Teilziele. Zunächst werden Testprobleme und Fahrzeugmodelle vorgestellt, die im hinteren Teil der Arbeit für verschiedene Analysen herangezogen werden. Anschließend folgt die Einführung einer geeigneten Parametrik, um die verschiedenen Typen von Parametern einer Architektur zu charakterisieren und deren Behandlung innerhalb einer Optimierung zu definieren. Im Weiteren werden verschiedene Konzepte zur Architekturoptimierung vorgestellt. Ausgehend von Analysen verschiedener Strategien wird eine für weitere Untersuchungen geeignete ausgewählt. Für das Teilziel der robusten Architekturoptimierung wird ein mehrkriterieller Ansatz gewählt und implementiert. Um auch zeitaufwändige MKS-Simulationen integrieren zu können, werden verschiedene, Effizienz-steigernde Methoden untersucht. Die so entwickelte Strategie wird dann zu einer robusten Architekturoptimierung erweitert und auf ein Testproblem angewendet. Zuletzt erfolgt die Anwendung auf in der Industrie eingesetzte komplexe MKS-Fahrzeugmodelle.

Kapitel 2

Evaluationsmodelle für die Architekturoptimierung

Digitale Prototypen sind ein in der Fahrzeugentwicklung etabliertes Hilfsmittel. Sie bedienen alle CAE-Berechnungsdisziplinen im frühen Entwicklungsprozess. Die Funktionen eines Fahrzeugs können mithilfe des digitalen Prototypen dargestellt, analysiert und verbessert werden. Die das Fahrwerk betreffenden Funktionen sind Fahrkomfort, Fahrsicherheit und Fahrdynamik [11].

Für die Bewertung dieser Funktionen gibt es eine Vielzahl von Kriterien, die in Normen, in der Literatur oder vom jeweiligen Hersteller selbst definiert sind. Je nach Phase des Entwicklungsprozesses und abhängig von der Ingenieursdisziplin werden unterschiedlich komplexe Evaluationsmodelle zur Berechnung dieser Kriterien verwendet. Während in Forschung und Methodenentwicklung zumeist recht einfache Simulationsmodelle zum Einsatz kommen, setzen Automobilhersteller häufig komplexe Gesamtfahrzeugmodelle ein.

In der Entwicklungsabteilung werden kurzfristig vertrauenswürdige Ergebnisse gefordert, die oft nur mit aufwändigen Modellen und langen Rechenzeiten realisiert werden können. Zusätzlich erfordert die automatisierte Suche eines Optimums eine Vielzahl an Simulationen, die das Rechenzeitproblem weiter verschärfen. Einen Beitrag zur Lösung dieses Zielkonflikts können effiziente Optimierungsprozesse und -algorithmen liefern, die im hinteren Teil der Arbeit diskutiert werden. Einen weiteren Beitrag können die in der Fahrwerksent-

wicklung eingesetzten Simulationstechniken, Simulationsmodelle und Kriterien leisten, die im Folgenden genauer betrachtet werden.

2.1 Simulation in der Fahrzeugentwicklung

Ein Produktentstehungsprozess umfasst Produktentwicklung, Produktionsentwicklung und Produktherstellung, die jeweils in verschiedene Phasen untergliedert sind. In allen diesen Abschnitten des Entstehungsprozesses werden Simulationswerkzeuge eingesetzt, wobei sich die folgende Arbeit auf die Produktentwicklung von Pkw bezieht.

Die Produktentwicklung wird nach Wedeniwski [92] in Vorentwicklung, Konzeptentwicklung und Serienentwicklung mit den sich teilweise überschneidenden Planungs-, Konzept/Entwurfs- und Serienentwicklungsphasen unterteilt, siehe Abb. 2.1. In der Planungsphase werden die grundlegenden Anforderungen und die Positionierung eines neuen Fahrzeugs diskutiert und festgelegt. Hierbei werden erste, die Entscheidungsfindung unterstützende analytische oder mithilfe sehr einfacher Modelle berechnete Abschätzungen getroffen. In der Konzept- und Entwurfsphase folgt die Definition aufwändigerer Simulationsmodelle, die im Laufe der Entwicklung bis in die fortgeschrittene Serienentwicklungsphase hinein eingesetzt und erweitert werden. Mithilfe dieser Modelle wird das reale Fahrzeug kontinuierlich verbessert, wobei verschiedene Berechnungsdisziplinen wie die Finite-Elemente-Methode (FEM), Computational Fluid Dynamics (CFD), MKS und Digital Mock-Up (DMU) zum Einsatz kommen [92]. Im Rahmen dieser Arbeit werden niederfrequente Ereignisse bis 30 Hz untersucht,

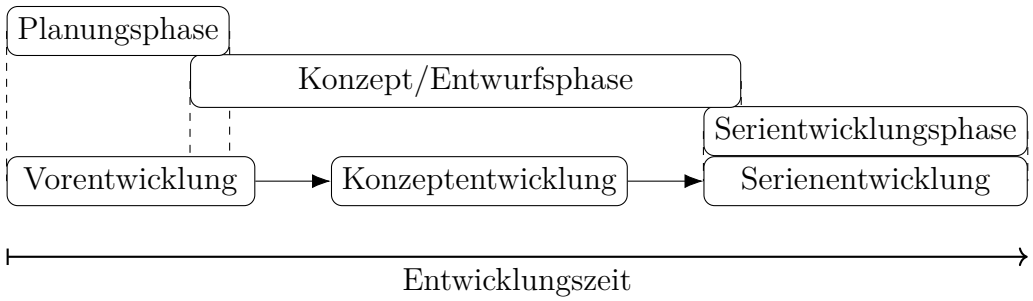

Abbildung 2.1: Die wesentlichen Phasen eines Produktentwicklungsprozesses

weshalb der Fokus auf der Mehrkörperdynamik liegt, die sich für diesen Frequenzbereich etabliert hat [97].

Für die Erzeugung vergleichbarer Simulationsergebnisse gibt es Meilensteine jeweils fester Entwicklungsstände, die von den verschiedenen Berechnungsgewerken bewertet werden. Im Laufe der Entwicklung folgt das Testen realer Komponenten auf Prüfständen oder in sogenannten Aggregateträgern, mittels derer die gewonnenen Simulationsergebnisse validiert und die Prognosefähigkeit der Simulation verbessert werden können. Denn trotz hochentwickelter Verfahren und wachsender Rechenleistung hat die Simulation Grenzen, so dass noch nicht auf die Prüfung realer Komponenten und Prototypen verzichtet werden kann [11, 38].

2.2 Kriterien zur Bewertung von Fahrzeugeigenschaften

Wie bereits in der Einleitung von Kapitel 2 angedeutet, gibt es eine Vielzahl von standardisierten Fahrmanövern und Kriterien zur Fahrzeugbewertung. Oftmals werden im Fahrversuch ausgeführte Manöver in die Simulation überführt, damit ein möglichst direkter Abgleich von Ergebnissen möglich ist. Aufgrund der großen Anzahl gängiger Fahrmanöver und Kriterien wird hier auf eine Aufzählung verzichtet. Die verwendeten Kriterien und deren Berechnung werden im Rahmen der Analysen in Kapitel 4 - 6 sowie in Abschnitt 2.3 und 2.4 näher erläutert. Umfangreiche, das allgemeine Fahrverhalten betreffende Aufzählungen und Erläuterungen von Wirkzusammenhängen von Fahrmanövern und Kriterien gibt es von Heißing [38] sowie Mitschke und Wallentowitz [51]. Zeller [97] betrachtet den Fahrkomfort detaillierter und listet die wesentlichen Schwingungsphänomene im Kraftfahrzeug auf.

Generell wird zwischen Quer- und Längsdynamik, die unter dem Oberbegriff Fahrdynamik zusammengefasst werden, und der Vertikaldynamik unterschieden, die maßgeblich den Fahrkomfort beeinflusst. Eine Einteilung aller Phänomene in dieses Raster ist aber nicht möglich; so gibt es beispielsweise das sogenannte

Trampeln oder Stempeln, das je nach Antriebskonzept bei starkem Beschleunigen oder Bremsen auftritt, also der Längsdynamik zugeordnet werden kann, aber einen erheblichen Einfluss auf das Wohlbefinden der Insassen und somit auf den Fahrkomfort hat [97].

In Abschnitt 2.1 wurde der Einsatz einfacher Berechnungsmodelle in der frühen Entwicklung erwähnt. Zur Abschätzung fahrkomfortrelevanter Kriterien seien hier das Viertelfahrzeugmodell und für fahrdynamikrelevante Kriterien das Einspurmodell erwähnt, die häufig in Veröffentlichungen Anwendung finden und sehr detailliert in Standardwerken beschrieben sind [97, 51, 38]. Das Einspurmodell eignet sich für die Abschätzung von Kriterien zur Bewertung der Querdynamik, wie beispielsweise des Eigenlenkgradienten, der ein Maß für Agilität und Fahrsicherheit darstellt. Das Viertelfahrzeugmodell, das in Abschnitt 2.3 genauer behandelt wird, kann zur Abschätzung von Eigenfrequenzen und Dämpfungsmaßen von Fahrzeugaufbau und Rad herangezogen werden, dient aber auch der Bewertung von Radlastschwankungen und Schwingverhalten bei kontinuierlichen Fahrbahnanregungen oder dem Überfahren von Einzelhindernissen.

Ein wichtiger Bestandteil der entwicklungsbegleitenden Simulation ist die Berechnung charakteristischer Größen der Kinematik und Elastokinematik des Fahrwerks. Die Kinematik meint hierbei die durch Lenker und Drehgelenke beschränkte Bewegung des Rades beim Einfedern oder Lenken, woraus Größen wie Übersetzungsverhältnisse und Drehachsen abgeleitet werden. Bei der Elastokinematik werden die starren Gelenke teilweise durch nachgiebige Gummilager ausgetauscht, die heutzutage in jedem Großserienfahrzeug im Fahrwerk verwendet werden. Hierbei wird die Radstellungsänderung unter Einprägung von Längs- oder Querkräften bewertet, wodurch beispielsweise Rückschlüsse auf das Eigenlenkverhalten oder den Geradeauslauf des Fahrzeugs gezogen werden können. Der große Vorteil dieser Betrachtungen ist, dass die Modellierung jeweils nur einzelner, fest eingespannter Achsen notwendig ist und diese statisch, also ohne Berücksichtigung der Zeit, simuliert werden können. Dies führt zu einer deutlich geringeren Rechenzeit, verglichen mit einer dynamischen Gesamtfahrzeugsimulation, bei der mindestens die doppelte Anzahl von Freiheitsgraden

existiert und die Manöver zumeist über einen längeren Simulationszeitraum bewertet werden [11].

In den Entwicklungsabteilungen der Automobilhersteller werden auf Erfahrungen basierende Werte oder Wertebereiche für in Kinematik- und Elastokinematiksimulationen bestimmbare Kriterien zur Verbesserung des Fahrwerks herangezogen. Dies hilft insbesondere in der frühen Entwicklungsphase, wenn noch kein vollständig parametriertes Gesamtfahrzeugmodell existiert und regelmäßig Anpassungen an Gelenkpositionen oder Gummilagersteifigkeiten vorgenommen werden [38].

Vollfahrzeugsimulationen folgen im weiteren Verlauf der Entwicklung. Hierfür müssen nicht nur umfangreiche und das Fahrzeug beschreibende Daten vorhanden sein, sondern auch Straßen und Reifen entsprechend der Vorgaben modelliert werden. Aufbau und Simulation eines Gesamtfahrzeugmodells sind folglich sehr aufwändig und komplex, siehe Abschnitt 2.4, liefern aber tiefgreifende Erkenntnisse über das Schwingungs- und Fahrverhalten.

Abschließend muss erwähnt werden, dass es trotz weit entwickelter Soft- und Hardware noch immer Kriterien gibt, die sich nicht virtuell bewerten lassen. Fahrverhalten oder Fahrkomfort werden von Testfahrern subjektiv bewertet, wobei das Gefühlte teilweise nicht mit aus Simulationen gewonnenen Erkenntnissen korreliert [38].

2.3 Einfaches Testproblem

Ein einfaches Fahrzeugmodell, das in der sehr frühen Entwicklungsphase und für akademische Zwecke verwendet wird, ist das Viertelfahrzeugmodell. Bei dem Modell wird, wie der Name vermuten lässt, nur ein Viertel eines Fahrzeugs betrachtet. Es wird zur Abbildung der Vertikaldynamik eines Pkw herangezogen. Typischerweise werden nur die Massen von Rad und Aufbau, zugehörige Verbindungselemente wie Feder und Dämpfer und die Materialeigenschaften des Reifens berücksichtigt. Damit ergibt sich ein Zweimassenschwinger, der als einfachste Repräsentation eines Fahrzeugmodells gesehen werden kann. Eine

ausführliche Analyse des aus zwei Massen bestehenden Viertelfahzeugmodells und die Berechnung verschiedener Manöver werden von Zeller [97] beschrieben.

Die Vorteile des Modells sind geringe Komplexität und kurze Berechnungsdauer. Allerdings sind die Ergebnisse nur bedingt auf die Realität übertragbar, da das Viertelfahrzeugmodell auf einigen Annahmen basiert, die zu beachten sind. Zum einen ist das Fahrzeug um die Längsachse symmetrisch und das linke und das rechte Rad werden identisch angeregt, es treten also keine Wankbewegungen auf. Die Vorder- und Hinterachse haben identische Eigenschaften und bewegen sich parallel zueinander. Zudem hat das Fahrzeug einen unendlich großen Radstand, so dass Nickschwingungen vernachlässigt werden können [51].

Zur Analyse verschiedener Strategien zur Architekturoptimierung wird für diese Arbeit ein erweitertes, aus drei Massen bestehendes Viertelfahrzeugmodell verwendet. In der Literatur bildet die zusätzliche Masse typischerweise die Insassen eines Fahrzeugs ab, die sich auf einem elastisch an die Karosserie angebundenen Sitz befinden. Andere Beispiele sind abgekoppelte Antriebe, Hilfsrahmen oder Fahrerhäuser, die in großen Nutzfahrzeugen eingesetzt werden [51]. In dieser Arbeit wird die zusätzliche Masse als Hilfsrahmen, der auch als Fahrschemel bezeichnet wird, betrachtet.

Das hier verwendete, in Abb. 2.2 dargestellte Modell bezieht sich auf einen Pkw mit einem durch Gummilager elastisch abgekoppelten Fahrschemel m_F. Fahrschemel werden heutzutage in vielen Pkw aus gehobenen Fahrzeugsegmenten zur Verbesserung des Geräuschkomforts eingesetzt [38]. Um die Zugehörigkeit zu unterschiedlichen Segmenten zu repräsentieren, werden zwei Fahrzeugmodelle unterschieden; eines mit einem einfachen Federungssystem in Abb. 2.2a und eines mit einem komplexeren System in Abb. 2.2b. Die drei Freiheitsgrade des Aufbaus z_A, des Fahrschemels z_F und des Rads z_R beschreiben die Auslenkung der Massen aus der jeweiligen Gleichgewichtslage. Systemparameter sind die Massen m_A, m_F und m_R sowie die zugehörigen Feder- und Dämpferkonstanten von Federungssystem, Reifen und Gummilagern. Das System in Abb. 2.2b besitzt zusätzlich den inneren Freiheitsgrad w, die Steifigkeit c_{F2} und Dämpfung d_{F2} zur Modellierung des komplexeren Federungssystems. Beide Fahrzeugmo-

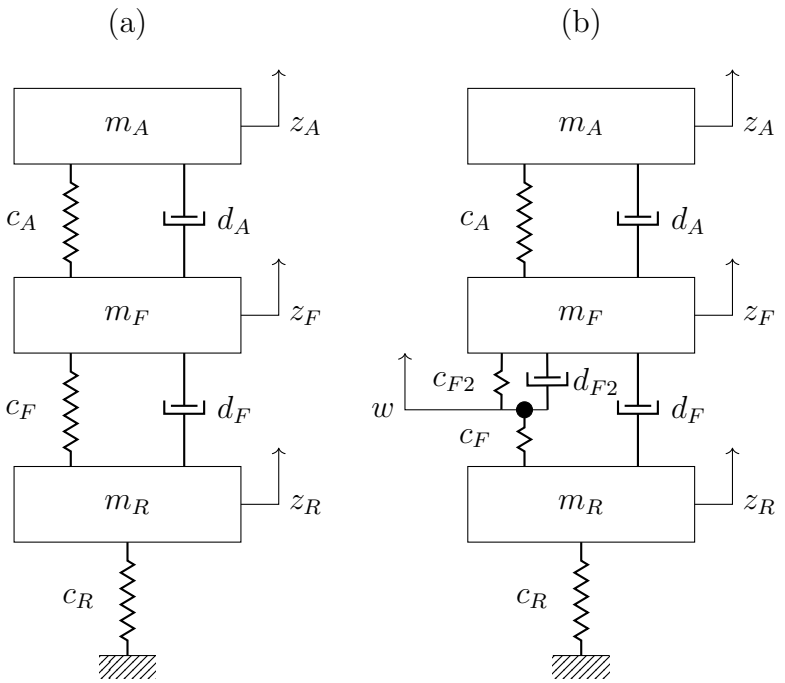

Abbildung 2.2: Viertelfahrzeugmodell mit (a) einfachem und (b) komplexerem Federungssystem

delle können über Systeme linearer Differentialgleichungen analysiert werden, die im Folgenden analytisch hergeleitet werden.

Für beide Systeme in Abb. 2.2 ergibt sich aus dem Impulssatz für die drei Massen jeweils

$$\begin{aligned} m_R\ddot{z}_R &= F_F - F_R, \\ m_F\ddot{z}_F &= F_A - F_F, \\ m_A\ddot{z}_A &= -F_A. \end{aligned} \tag{2.1}$$

Für das System in Abb. 2.2a sind die Kräfte der Koppelelemente $F_R = c_R z_R$, $F_F = c_F(z_F - z_R) + d_F(\dot{z}_F - \dot{z}_R)$ und $F_A = c_A(z_A - z_F) + d_A(\dot{z}_A - \dot{z}_F)$. Eingesetzt erhält man die gekoppelten Bewegungsgleichungen

$$\begin{aligned} m_R\ddot{z}_R + d_F\dot{z}_R - d_F\dot{z}_F + (c_R + c_F)z_R - c_F z_F &= 0, \\ m_F\ddot{z}_F - d_F\dot{z}_R + (d_F + d_A)\dot{z}_F - d_A\dot{z}_A - c_F z_R + (c_F + c_A)z_F - c_A z_A &= 0, \\ m_A\ddot{z}_A - d_A\dot{z}_F + d_A\dot{z}_A - c_A z_F + c_A z_A &= 0. \end{aligned} \tag{2.2}$$

Diese lassen sich kompakt in Matrixschreibweise darstellen als

$$\boldsymbol{M}\ddot{\boldsymbol{y}} + \boldsymbol{D}\dot{\boldsymbol{y}} + \boldsymbol{K}\boldsymbol{y} = \boldsymbol{0} \tag{2.3}$$

mit $\boldsymbol{y} = \begin{bmatrix} z_R & z_F & z_A \end{bmatrix}^T$, der Massenmatrix $\boldsymbol{M}$, Dämpfungsmatrix $\boldsymbol{D}$ und Steifigkeitsmatrix $\boldsymbol{K}$:

$$\begin{gathered} \boldsymbol{M} = \begin{bmatrix} m_R & 0 & 0 \\ 0 & m_F & 0 \\ 0 & 0 & m_A \end{bmatrix}, \quad \boldsymbol{D} = \begin{bmatrix} d_F & -d_F & 0 \\ -d_F & d_F + d_A & -d_A \\ 0 & -d_A & d_A \end{bmatrix}, \\ \boldsymbol{K} = \begin{bmatrix} c_R + c_F & -c_F & 0 \\ -c_F & c_F + c_A & -c_A \\ 0 & -c_A & c_A \end{bmatrix}. \end{gathered} \tag{2.4}$$

Da es sich um ein gewöhnliches mechanisches Schwingungssystem handelt, kann es in die Zustandsraumdarstellung $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ mit folgender Struktur überführt werden:

$$\begin{bmatrix} \dot{\boldsymbol{y}} \\ \ddot{\boldsymbol{y}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{E} \\ -\boldsymbol{M}^{-1}\boldsymbol{K} & -\boldsymbol{M}^{-1}\boldsymbol{D} \end{bmatrix} \begin{bmatrix} \boldsymbol{y} \\ \dot{\boldsymbol{y}} \end{bmatrix}, \tag{2.5}$$

wobei $\boldsymbol{E}$ die Einheitsmatrix repräsentiert [54].

Für das Fahrzeugmodell in Abb. 2.2b muss der zusätzliche Freiheitsgrad w berücksichtigt werden. Die Systemmatrix $\boldsymbol{A}$ kann jedoch nicht analog zu (2.5) aufgestellt werden, da die Massenmatrix aufgrund des masselosen Knotens singulär würde. Nach Müller und Schiehlen [54] handelt es sich hier um ein allgemeines lineares System, bei dem w zwar eine zusätzliche Zustandsgröße ist, die jedoch nur einem halben Freiheitsgrad entspricht. Analog zu (2.1) ergibt sich zunächst

$$m_R\ddot{z}_R = F_F - F_R, \tag{2.6}$$

$$0 = F_{F2} - F_{c_F}, \tag{2.7}$$

$$m_F\ddot{z}_F = F_A - F_{d_F} - F_{F2}, \tag{2.8}$$

$$m_A\ddot{z}_A = -F_A. \tag{2.9}$$

Folgende Kräfte der Koppelelemente ändern sich gegenüber (2.1) zu $F_F = F_{c_F} + F_{d_F}$, $F_{c_F} = c_F(w - z_R)$, $F_{d_F} = d_F(\dot{z}_F - \dot{z}_R)$ und $F_{F2} = c_{F2}(z_F - w) +$

$d_{F2}(\dot{z}_F - \dot{w})$. Eingesetzt in (2.7) erhält man für den zusätzlichen Knoten die Gleichgewichtsbedingung

$$c_F(w - z_R) = c_{F2}(z_F - w) + d_{F2}(\dot{z}_F - \dot{w}), \tag{2.10}$$

oder Differentialgleichung 1. Ordnung

$$\dot{w} = -\frac{c_F}{d_{F2}} z_R - \frac{c_{F2}}{d_{F2}} z_F + \dot{z}_F + \frac{c_F + c_{F2}}{d_{F2}} w. \tag{2.11}$$

Durch Einsetzen von (2.10) in (2.8) kann $\dot{w}$ eliminiert werden, wodurch statt (2.2) die Gleichungen

$$\begin{aligned} m_R\ddot{z}_R + d_F\dot{z}_R - d_F\dot{z}_F + (c_R + c_F)z_R - c_F w = 0, \\ m_F\ddot{z}_F - d_F\dot{z}_R + (d_F + d_A)\dot{z}_F - d_A\dot{z}_A - c_F z_R + c_A z_F - c_A z_A + c_F w = 0, \\ m_A\ddot{z}_A - d_A\dot{z}_F + d_A\dot{z}_A - c_A z_F + c_A z_A = 0 \end{aligned} \tag{2.12}$$

entstehen. Die Bewegungsgleichung lautet dann in Matrixschreibweise

$$\boldsymbol{M}\ddot{\boldsymbol{y}} + \boldsymbol{D}\dot{\boldsymbol{y}} + \boldsymbol{K}\boldsymbol{y} + \boldsymbol{w}w = \boldsymbol{0} \tag{2.13}$$

mit Massen- und Dämpfungsmatrix (2.4), aber einer geänderten Steifigkeitsmatrix und einem zusätzlichen Vektor

$$\boldsymbol{K} = \begin{bmatrix} c_R + c_F & 0 & 0 \\ -c_F & c_A & -c_A \\ 0 & -c_A & c_A \end{bmatrix}, \quad \boldsymbol{w} = \begin{bmatrix} -c_F \\ c_F \\ 0 \end{bmatrix}. \tag{2.14}$$

Die Zusammenfassung von (2.13) und (2.11) liefert dann die Zustandsraumdarstellung

$$\begin{bmatrix} \dot{\boldsymbol{y}} \\ \ddot{\boldsymbol{y}} \\ \dot{w} \end{bmatrix} = \left[\begin{array}{ccc:ccc:c} & \boldsymbol{0} & & & \boldsymbol{E} & & \boldsymbol{0} \\ & -\boldsymbol{M}^{-1}\boldsymbol{K} & & & -\boldsymbol{M}^{-1}\boldsymbol{D} & & -\boldsymbol{M}^{-1}\boldsymbol{w} \\ -\frac{c_F}{d_{F2}} & -\frac{c_{F2}}{d_{F2}} & 0 & 0 & 1 & 0 & \frac{c_F + c_{F2}}{d_{F2}} \end{array}\right] \begin{bmatrix} \boldsymbol{y} \\ \dot{\boldsymbol{y}} \\ w \end{bmatrix}. \tag{2.15}$$

Mit (2.5) und (2.15) lassen sich die Systeme in Abb. 2.2 je nach geforderten Kriterien allgemein lösen oder deren Eigenwerte analysieren. Im Rahmen dieser Arbeit steht die Eigenwertanalyse zur Berechnung von Gütekriterien im Fokus, die nachfolgend erläutert wird.

Mit der Exponentialfunktion $\boldsymbol{x}\,(t) = \hat{\boldsymbol{x}}\,\mathrm{e}^{\lambda t}$ als Eigenfunktion von $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ ergibt sich als Lösungsbedingung das Verschwinden der Determinante

$$\det(\lambda \boldsymbol{E} - \boldsymbol{A}) = 0. \tag{2.16}$$

Die sich ergebenden komplexen Eigenwerte $\lambda_n \in \mathbb{C}$ implizieren die Eigenfrequenzen f_n^{eig} und Dämpfungsmaße D_n jeder Schwingungsmode:

$$f_n^{eig} = \frac{|\mathrm{Im}(\lambda_n)|}{2\pi}, \quad D_n = \frac{-\,\mathrm{Re}(\lambda_n)}{|\lambda_n|}. \tag{2.17}$$

Die Eigenwerte λ_n werden numerisch mit den Routinen des Linear Algebra Package (LAPACK) [3] berechnet. Da die Werte der Systemparameter während der Optimierung variiert werden, kann sich die Reihenfolge der berechneten Eigenwerte ändern. Die Gewährleistung einer klaren Zuordnung zu zuvor bestimmten Referenzwerten λ_n^{ref} ist daher im Allgemeinen nicht gegeben. Deshalb wird ein Kriterium zur Identifikation von Modalvektoren, das sogenannte Model Assurance Criterion (MAC) [2], verwendet:

$$\mathrm{MAC}_{n,m} = \frac{\left(\hat{\boldsymbol{x}}_n^T \hat{\boldsymbol{x}}_m^{ref}\right)^2}{\left(\hat{\boldsymbol{x}}_n^T \hat{\boldsymbol{x}}_n\right)\left(\hat{\boldsymbol{x}}_m^{ref,T} \hat{\boldsymbol{x}}_m^{ref}\right)}, \tag{2.18}$$

wobei $\hat{\boldsymbol{x}}_n$ und $\hat{\boldsymbol{x}}_m^{ref}$ die aktuellen bzw. Referenzeigenvektoren des Systems sind. Die Zuordnung erfolgt dann jeweils durch den größten MAC-Wert.

2.4 Gesamtfahrzeugmodell

Grundsätzlich wird ein Gesamtfahrzeug mit den gleichen Elementen wie das Testproblem in Abschnitt 2.3 modelliert: mit massebehafteten Körpern, masselosen Koppelelementen, wie Federn und Dämpfer, sowie Bindungen. Der große Unterschied liegt in der Dreidimensionalität und der deutlich größeren Anzahl von Elementen und Freiheitsgraden; auch bedarf die Kinematik einer großen Systemmatrix. Von einer analytischen Lösung muss unter diesen Gegebenheiten aus Gründen von Zeitaufwand und Fehleranfälligkeit abgesehen werden. Deshalb wurden effiziente Algorithmen entwickelt, die komplexe mechanische Systeme nicht nur numerisch lösen, sondern auch die dafür benötigten Bewegungsgleichungen automatisiert aufstellen [68]. Für die Anwendung dieser Algorithmen

gibt es benutzerfreundliche Programme, die den Ingenieur bei Modellierung, Simulation und grafischer Darstellung von Ergebnissen unterstützen. Gemeinhin wird die Analyse eines mechanischen Systems unter Verwendung dieser Programme und Algorithmen als Mehrkörpersimulation (MKS) bezeichnet [5].

2.4.1 Mehrkörpersimulation

Um ein Mehrkörpersystem mit p Körpern K_i, $i = 1...p$, analysieren zu können, muss es mathematisch beschrieben werden. Hierfür sind die Körper gedanklich freizuschneiden und die eingeprägten Kräfte $\boldsymbol{f}_i^e$ und Momente $\boldsymbol{l}_i^e$ sowie Reaktionskräfte $\boldsymbol{f}_i^r$ und -momente $\boldsymbol{l}_i^r$ zu beschreiben. Mit Orts- bzw. Lagevektor $\boldsymbol{r}_i$ und Trägheitstensor $\boldsymbol{I}_i$ des Schwerpunktes der Masse m_i sowie der Winkelgeschwindigkeit $\boldsymbol{\omega}_i$ lauten Impuls- und Drallsatz

$$m_i \ddot{\boldsymbol{r}}_i = \boldsymbol{f}_i^e + \boldsymbol{f}_i^r, \tag{2.19}$$

$$\boldsymbol{I}_i \dot{\boldsymbol{\omega}}_i + \tilde{\boldsymbol{\omega}}_i \boldsymbol{I}_i \boldsymbol{\omega}_i = \boldsymbol{l}_i^e + \boldsymbol{l}_i^r, \tag{2.20}$$

wobei $\tilde{\boldsymbol{\omega}}_i$ die schiefsymmetrische Matrix zu $\boldsymbol{\omega}_i$ ist, die das Kreuzprodukt vermittelt [5].

Das in dieser Arbeit verwendete Gesamtfahrzeugmodell, Abb. 2.3, besteht aus insgesamt 107 Körpern und besitzt 154 Freiheitsgrade. Modelliert sind Fahrwerksstreben, Achsschenkel, Gummilager, Gelenke, Federn und Dämpfer, Räder und Reifen sowie der Aufbau. An der Vorderachse gibt es zusätzlich eine Lenkung. Die Hinterachse besitzt einen Fahrschemel, Antriebswellen und ein Differential. Auf die komplette Modellierung des Antriebsstrangs inklusive des Motors wird verzichtet. Die Modellierung der Gummilager haben in dieser Arbeit eine besondere Bedeutung und werden in Abschnitt 2.4.2 genauer betrachtet.

Zur Aufstellung der Bewegungsgleichungen gibt es verschiedene Ansätze. Eine Möglichkeit ist, die Bewegungen in kartesischen Koordinaten zu beschreiben, also mit jeweils drei Raum- und drei Winkelkoordinaten. Generell lassen sich die Bewegungsgleichungen dann einfach darstellen, es sind für die Simulation

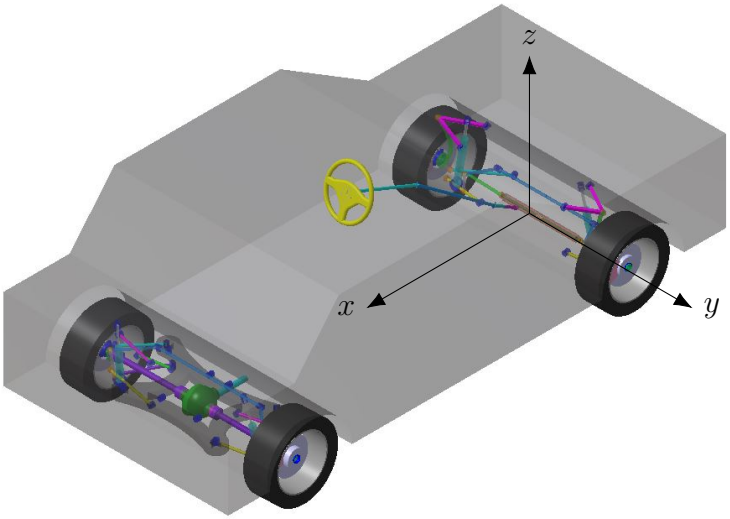

Abbildung 2.3: Gesamtfahrzeugmodell für die MKS

des Systems aber zusätzlich algebraische Zwangsbedingungen zwischen den Beschreibungsgrößen zu berücksichtigen, siehe [36]. Eine andere Möglichkeit ist die Verwendung von verallgemeinerten Koordinaten, wobei das System nur mit dem minimalen Satz von notwendigen unabhängigen Koordinaten eindeutig beschrieben wird. Daraus resultiert ein System gewöhnlicher Differentialgleichungen, für dessen Simulation weit verbreitete und effiziente Integrationsverfahren verwendet werden können, was einen großen Vorteil darstellt [5]. Die folgende Beschreibung zur Herleitung von Bewegungsgleichungen gibt nur einen kleinen Einblick in die Systematik.

Ein Ansatz, um Reaktionskräfte und -momente $\boldsymbol{f}_i^r$ und $\boldsymbol{l}_i^r$ von Bindungen in den Newton-Eulerschen Gleichungen (2.19) und (2.20) zu eliminieren, ist das d'Alembertsche Prinzip. Beschreibt man mögliche Bewegungen durch infinitesimale virtuelle Verschiebungen $\delta\boldsymbol{r}_i \neq \boldsymbol{0}$ und Verdrehungen $\delta\boldsymbol{s}_i \neq \boldsymbol{0}$, die mit den Bindungen verträglich sind, verschwindet die virtuelle Arbeit der durch ideale Bindungen entstehenden Reaktionskräfte, d.h.

$$\delta W^r = \sum_{i=1}^{p} \left(\boldsymbol{f}_i^{rT} \delta\boldsymbol{r}_i + \boldsymbol{l}_i^{rT} \delta\boldsymbol{s}_i \right) = 0. \tag{2.21}$$

Setzt man die Gleichungen (2.19) und (2.20) in diese Beziehung ein, erhält man

die Bewegungsgleichungen für holonome Mehrkörpersysteme in Variationsform

$$\sum_{i=1}^{p} \left[\delta \boldsymbol{r}_i^T \left(m_i \ddot{\boldsymbol{r}}_i - \boldsymbol{f}_i^e \right) + \delta \boldsymbol{s}_i^T \left(\boldsymbol{I}_i \dot{\boldsymbol{\omega}}_i + \tilde{\boldsymbol{\omega}}_i \boldsymbol{I}_i \boldsymbol{\omega}_i - \boldsymbol{l}_i^e \right) \right] = 0. \tag{2.22}$$

Für nichtholonome Bindungen verwendet man statt (2.21) das Jourdainsche Prinzip, bei dem statt der Lage- die Geschwindigkeitsgrößen variiert werden.

Um das System nun analysieren zu können, müssen die Variationen $\delta \boldsymbol{r}_i^T$ und $\delta \boldsymbol{s}_i^T$ in den Bewegungsgleichungen (2.22) eliminiert werden. Mit der Wahl voneinander unabhängiger Koordinaten $\boldsymbol{y} \in \mathbb{R}^f$ können alle f Freiheitsgrade eines holonomen Mehrkörpersystems mit Baumstruktur als voneinander unabhängige Teilbewegungen beschrieben werden. Diese Koordinaten müssen die Lage und Orientierung aller Körper eindeutig festlegen. Die Orientierung wird hierbei durch die Drehmatrizen $\boldsymbol{S}_i$ definiert:

$$\boldsymbol{r}_i = \boldsymbol{r}_i \left(t, \boldsymbol{y} \right), \quad \boldsymbol{S}_i = \boldsymbol{S}_i \left(t, \boldsymbol{y} \right). \tag{2.23}$$

Die virtuellen Verschiebungen $\delta \boldsymbol{r}_i = \boldsymbol{J}_{Ti} \delta \boldsymbol{y}$ und Verdrehungen $\delta \boldsymbol{s}_i = \boldsymbol{J}_{Ri} \delta \boldsymbol{y}$ ergeben sich über die Jacobimatrizen der Translation und der Rotation $\boldsymbol{J}_{Ti}$ und $\boldsymbol{J}_{Ri}$. Aus (2.22) ergibt sich dann

$$\delta \boldsymbol{y}^T \sum_{i=1}^{p} \left[\boldsymbol{J}_{Ti}^T \left(m_i \ddot{\boldsymbol{r}}_i - \boldsymbol{f}_i^e \right) + \boldsymbol{J}_{Ri}^T \left(\boldsymbol{I}_i \dot{\boldsymbol{\omega}}_i + \tilde{\boldsymbol{\omega}}_i \boldsymbol{I}_i \boldsymbol{\omega}_i - \boldsymbol{l}_i^e \right) \right] = 0. \tag{2.24}$$

Aufgrund der Unabhängigkeit der virtuellen Verschiebungen $\delta \boldsymbol{y}$ erhält man die Bewegungsgleichungen für holonome Mehrkörpersysteme als

$$\sum_{i=1}^{p} \left[\boldsymbol{J}_{Ti}^T m_i \ddot{\boldsymbol{r}}_i + \boldsymbol{J}_{Ri}^T \left(\boldsymbol{I}_i \dot{\boldsymbol{\omega}}_i + \tilde{\boldsymbol{\omega}}_i \boldsymbol{I}_i \boldsymbol{\omega}_i \right) \right] = \sum_{i=1}^{p} \left(\boldsymbol{J}_{Ti}^T \boldsymbol{f}_i^e + \boldsymbol{J}_{Ri}^T \boldsymbol{l}_i^e \right). \tag{2.25}$$

Nach Einarbeitung der nichtlinearen Kinematikbeziehungen (2.23) ergibt sich dann die Bewegungsgleichung in Form von nichtlinearen, gewöhnlichen Differentialgleichungen 2. Ordnung

$$\boldsymbol{M} \left(t, \boldsymbol{y} \right) \ddot{\boldsymbol{y}} + \boldsymbol{k} \left(t, \boldsymbol{y}, \dot{\boldsymbol{y}} \right) = \boldsymbol{q} \left(t, \boldsymbol{y}, \dot{\boldsymbol{y}} \right) \tag{2.26}$$

mit der $f \times f$-Massenmatrix $\boldsymbol{M}$, dem $f \times 1$-Vektor der verallgemeinerten Zentrifugal-, Kreisel- und Corioliskräfte und -momente $\boldsymbol{k}$ und dem $f \times 1$-Vektor der verallgemeinerten eingeprägten Kräfte und Momente $\boldsymbol{q}$, siehe z.B.

Bestle [5]. Anschließend kann das System in die Zustandsform überführt und mithilfe numerischer Integrationsverfahren gelöst werden. Holonome Systeme mit kinematischen Schleifen können u.a. durch ein differential-algebraisches Gleichungssystem mit differentiellen Kinematik- und Kinetikgleichungen sowie zusätzlichen algebraischen Schleifenschließgleichungen beschrieben werden, dessen numerische Lösung allerdings schwieriger ist. Für weiterführende Erläuterungen, auch zur Lösung holonomer Systeme mit kinematischen Schleifen in Minimalkoordinaten, sei an dieser Stelle auf das Buch von Bestle [5] verwiesen.

2.4.2 Gummilagermodellierung

Wie bereits erwähnt, spielen Gummi- bzw. Elastomerlager in der modernen Fahrzeugentwicklung eine große Rolle. Entsprechend wichtig ist die Berücksichtigung der Lager bei der Auslegung eines Fahrwerks in der Simulation, insbesondere bei der Bewertung der Elastokinematik.

Elastomere weisen ein stark voneinander abweichendes dynamisches und statisches Verhalten auf. Eine einzigartige Eigenschaft ist die kompressionsabhängige progressive Zunahme der statischen Steifigkeit. Dieses Verhalten wird typischerweise in Form eines Kraft-Weg-Diagramms dargestellt, siehe Abb. 2.4. Diese Kennlinie wird im Rahmen der Fahrwerksentwicklung auf den spezifischen Einsatzzweck des Lagers angepasst. Bei dynamischer Anregung wirkt ein Gummilager hingegen dämpfend und es ändert seine Steifigkeit. Hierbei wird von Amplituden- und Frequenzabhängigkeit gesprochen. Anregungsfrequenz und Amplitude beeinflussen jeweils die Dämpfungs- und Steifigkeitseigenschaften eines Elastomers. Abb. 2.5 zeigt das frequenzabhängige Verhalten $G(i\omega)$, das in Form von dynamischer Steifigkeit $c_{dyn} = |G(i\omega)|$ und Verlustwinkel $\varphi = \sphericalangle G(i\omega)$ dargestellt ist [59, 38].

In Forschung und Entwicklung werden verschiedene Typen von Gummilagermodellen verwendet. Diese sind zum einen sehr komplex und verlangsamen die Berechnung, können das Verhalten eines Elastomers aber sehr gut nachbilden. Auf der anderen Seite gibt es sehr einfache und schnell zu simulierende Modelle, bei denen allerdings ein Kompromiss bezüglich der Modellgüte eingegangen werden muss. Beispielhaft seien hier das nach Kelvin-Voigt (KV) bezeichnete

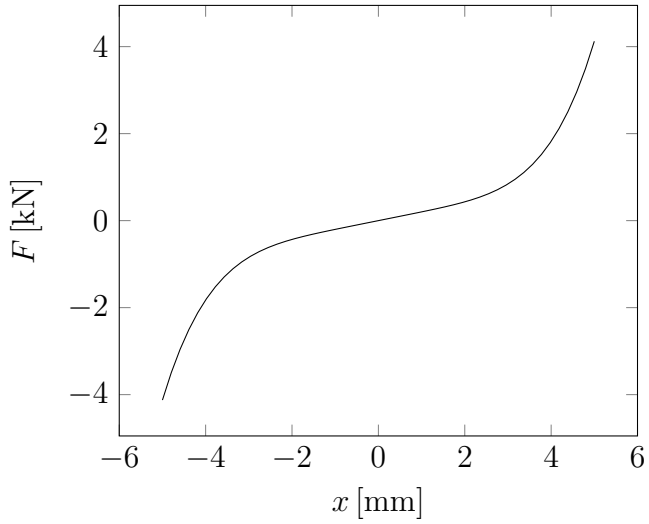

Abbildung 2.4: Progressive statische Steifigkeitscharakteristik eines Gummilagers

Modell [97], das Modell nach Pfeffer und Hofer [59] und ein von der Daimler AG entwickeltes [72] erwähnt.

Das im Rahmen dieser Arbeit verwendete Gummilagermodell ist ein KV-Modell. Das Modell besteht, wie in Abb. 2.2b dargestellt, aus einer Feder und einem dazu parallel geschalteten Dämpfer. Dieses Modell kann das reale Verhalten eines Gummilagers jedoch nur in Grenzen abbilden, beispielsweise gehen die korrekte Frequenz- und Amplitudenabhängigkeit verloren. Die Vorteile sind die geringe Anzahl von nur zwei Parametern und die hohe Effizienz bei der Berechnung. Um den Approximationsfehler zu reduzieren, wird das Modell jeweils auf eine spezifische Anregungsfrequenz $\omega_a \in [0, \tilde{\omega}]$, bei der das reale Gummilagerverhalten mit bester Genauigkeit abgebildet werden soll, parametriert. Da eine Fahrzeugachse typischerweise ausgeprägte Schwingformen besitzt, lässt sich die zugehörige Frequenz ω_a gut festlegen.

In der frühen Entwicklungsphase werden Gummilager gezielt auf neue Anforderungen hin entwickelt. Üblicherweise weisen Gummilager einer Achstopologie ähnliche Charakteristika auf, so dass auf Messungen von Lagern aus sich bereits in Serie befindlichen Fahrzeugen zurückgegriffen werden kann. Angenommen $c_{dyn}^{ref}(\omega_a)$ und $\varphi^{ref}(\omega_a)$ sind dynamische Steifigkeit und Verlustwinkel eines

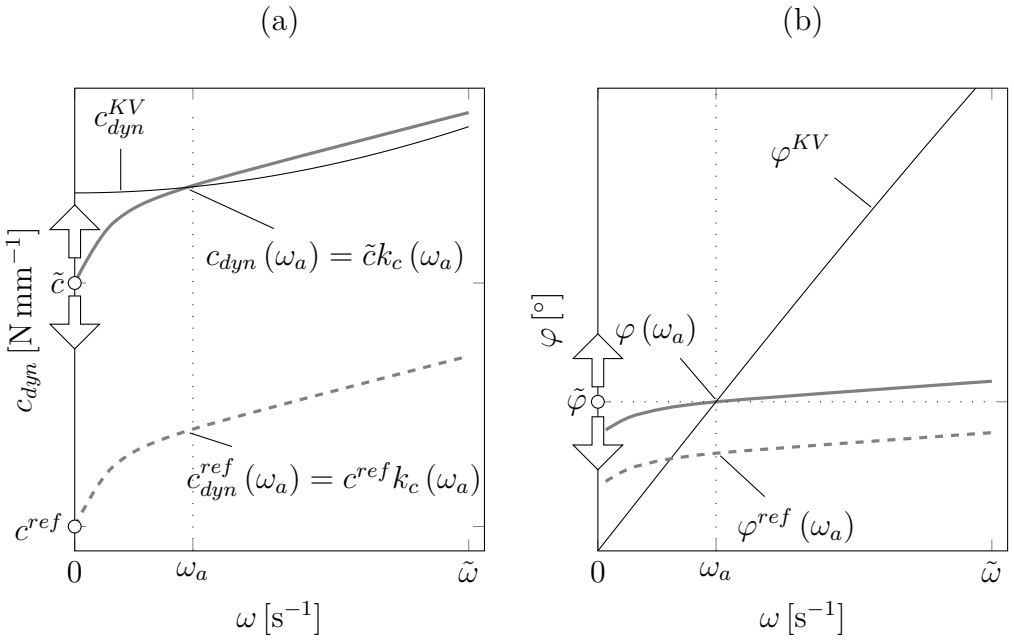

Abbildung 2.5: Verlauf der dynamischen Steifigkeit (a) und des Verlustwinkels (b) eines Referenzgummilagers (gestrichelt), eines gewünschten Lagers (grau) und des zugehörigen KV-Modells (schwarz)

vermessenen Referenzlagers entsprechend Abb. 2.5 (gestrichelt). Der charakteristische Verlauf von c_{dyn}^{ref} lässt sich dann für verschiedene Anregungsfrequenzen ω_a durch einen die dynamische Verhärtung beschreibenden Faktor

$$k_c\left(\omega_a\right)=\frac{c_{dyn}^{ref}\left(\omega_a\right)}{c^{ref}} \tag{2.27}$$

mit der statischen Steifigkeit $c^{ref}:=c_{dyn}^{ref}\left(0\right)$ ausdrücken.

Um verschiedene, von der Referenz abweichende Steifigkeiten im Modell berechnen zu können, wird nur die statische Steifigkeit des Gummilagers $\tilde{c}:=c_{dyn}\left(0\right)$ als Entwurfsvariable gewählt, während der in (2.27) bestimmte Verhärtungsfaktor $k_c\left(\omega_a\right)$ unverändert bleibt. Für eine gegebene Anregungsfrequenz ω_a kann die dynamische Steifigkeit dann mit

$$c_{dyn}\left(\omega_a\right)=\tilde{c}k_c\left(\omega_a\right) \tag{2.28}$$

berechnet werden.

Während sich die Steifigkeiten von Gummilagern im Fahrzeug bis zu einem Faktor von ungefähr 20 unterschieden, verweilen die Verlustwinkel, ausgenommen von Hydrolagern, auf einem ähnlichen Niveau. Weiterhin ist die Frequenzabhängigkeit des Verlustwinkels $\varphi^{ref}(\omega_a)$ geringer als bei der dynamischen Steifigkeit. Es wird daher angenommen, dass ein neu gewählter Verlustwinkel $\tilde{\varphi}$ unabhängig von ω_a ist und für den gesamten Frequenzbereich gilt:

$$\varphi(\omega_a) \approx \tilde{\varphi}. \tag{2.29}$$

Wenn die gewünschten Werte für $\tilde{c}$ und $\tilde{\varphi}$ gewählt sowie die Anregungsfrequenzen ω_a der zu berechnenden Fahrmanöver und der Verhärtungsfaktor $k_c(\omega_a)$ aus (2.27) bestimmt sind, können die zugehörigen Werte $c_{dyn}(\omega_a)$ und $\varphi(\omega_a)$ mit (2.28) und (2.29) berechnet werden. Diese Werte müssen dann in die zugehörige Steifigkeit c und Dämpfung d des KV-Ersatzmodells so konvertiert werden, dass die Kraft

$$F(t) = cx(t) + d\dot{x}(t) \tag{2.30}$$

für die Frequenz ω_a übereinstimmt. Um geeignete Transformationsgleichungen zu bestimmen, muss zunächst die Laplace-Transformation $F(s) = \mathcal{L}\{F(t)\}$ auf (2.30) angewandt werden, wobei $s = i\omega$ ist. Es ergibt sich dann die Übertragungsfunktion

$$G(i\omega) = \frac{F(i\omega)}{x(i\omega)} = c + id\omega. \tag{2.31}$$

Die Bestimmung der dynamischen Steifigkeit c_{dyn}^{KV} und des Verlustwinkels φ^{KV} des KV-Modells erfolgt mit

$$c_{dyn}^{KV}(\omega) = |G(i\omega)| = \sqrt{c^2 + d^2\omega^2}, \tag{2.32}$$

$$\varphi^{KV}(\omega) = \sphericalangle G(i\omega) = \arctan\left(\frac{d\omega}{c}\right). \tag{2.33}$$

Die Zusammenhänge lassen sich in einem komplexen Zeigerdiagramm, siehe Abb. 2.6, darstellen.

Bei gewählter Steifigkeit $\tilde{c}$ und Verlustwinkel $\tilde{\varphi}$ können die äquivalenten Größen c und d des KV-Modells entsprechend Abb. 2.6 und Gleichungen (2.28) und (2.29)

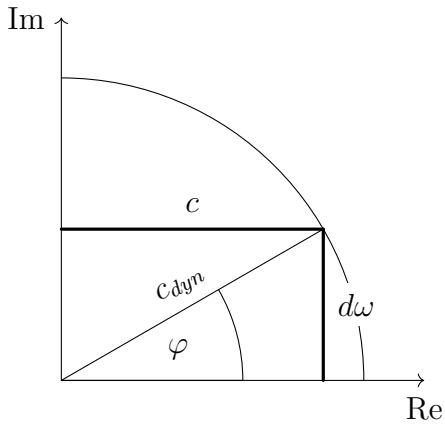

Abbildung 2.6: Repräsentation von dynamischer Steifigkeit c_{dyn}^{KV} und Verlustwinkel φ^{KV} in der komplexen Zahlenebene

berechnet werden, sodass dieses Modell die Eigenschaften eines Gummilagers bei der definierten Anregungsfrequenz ω_a bestmöglich approximiert:

$$c = c_{dyn}\left(\omega_a\right)\cos\varphi\left(\omega_a\right) = \tilde{c}k_c\left(\omega_a\right)\cos\tilde{\varphi}, \tag{2.34}$$

$$d = \frac{c_{dyn}\left(\omega_a\right)\sin\varphi\left(\omega_a\right)}{\omega_a} = \frac{\tilde{c}k_c\left(\omega_a\right)\sin\tilde{\varphi}}{\omega_a}. \tag{2.35}$$

Kapitel 3

Entwicklung neuer Fahrwerksarchitekturen

Bevor eine neue Fahrwerksarchitektur entwickelt wird, muss festgelegt werden, welche und wieviele Segmente mit auf dieser Architektur basierenden Fahrzeugmodellen zu bedienen sind. Nach Abschnitt 1.1 ist für eine Fahrzeugfamilie eines einzelnen Segments eine starre Architektur ausreichend, für mehrere Segmente sollte eine flexible Architektur verwendet werden. In der frühen Konzeptphase werden deshalb wichtige Eckdaten wie Rädergrößen, Antriebsvarianten und Grundabmessungen der Fahrzeugfamilie definiert [11].

In Abb. 3.1 sind einige Merkmale und Abmessungen von Pkw aus verschiedenen Segmenten dargestellt [38]. Die Werte der Merkmale korrelieren mit den Segmenten. Beispielsweise sind Minis und Kleinwagen deutlich kleiner und leichter als Fahrzeuge der Oberklasse. Lediglich die Fahrzeughöhe ist nicht oder sogar leicht negativ korreliert, was daran liegt, dass kurze Fahrzeuge in der Regel höher sind, um eine zufriedenstellende Innenraumgröße zu realisieren. Wichtig ist, dass je länger ein Fahrzeug ist, umso größer auch dessen Radstand und Breite ausfallen. Für die Entwicklung einer Fahrwerksarchitektur sind der Radstand und die mit der Fahrzeugbreite in Verbindung stehende Spurweite wichtige Merkmale. Im Weiteren wird deshalb die Annahme getroffen, dass diese die Hauptunterscheidungsmerkmale für Pkw verschiedener Segmente sind, siehe Abb. 3.2. Sind die Eckdaten der Fahrzeugfamilie definiert, kann die Topologie der Architektur festgelegt werden. In Bezug auf eine Fahrwerksarchitektur be-

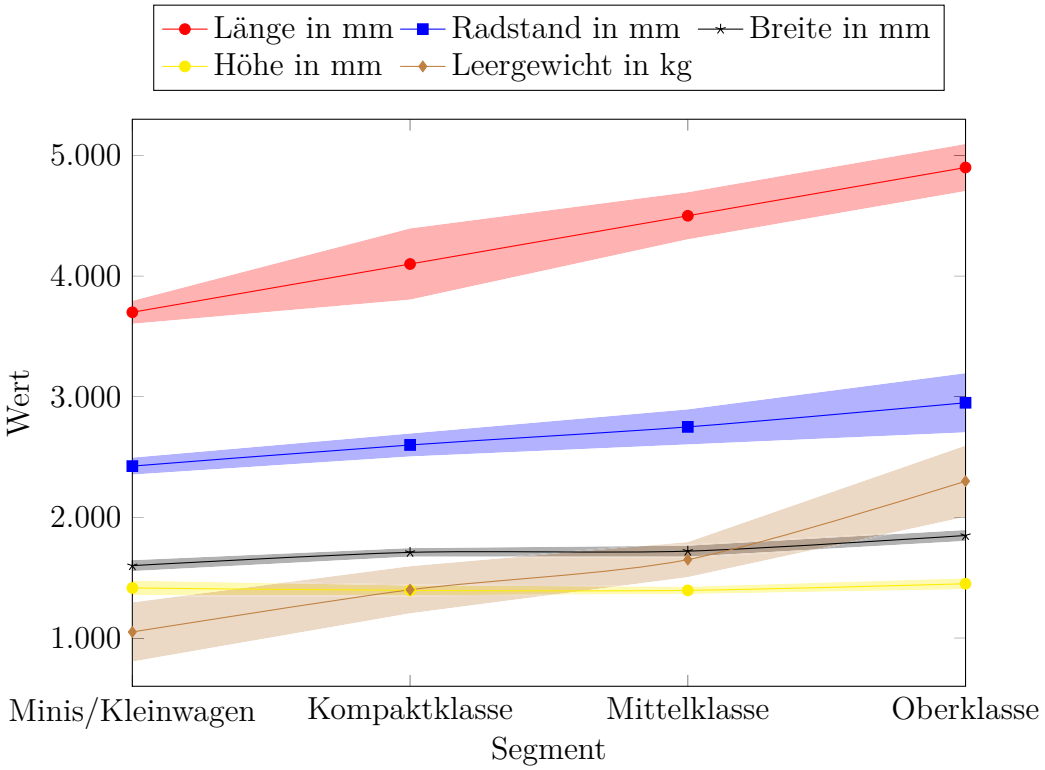

Abbildung 3.1: Übliche Wertebereiche von Merkmalen und Abmessungen von Pkw verschiedener Segmente

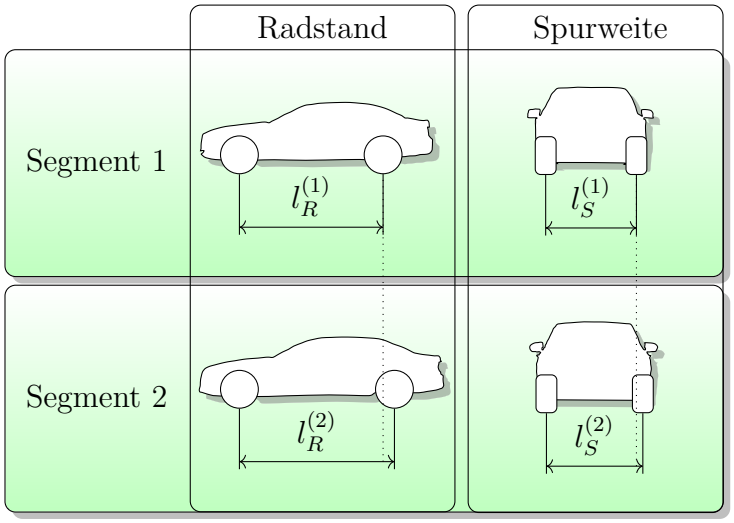

Abbildung 3.2: Radstand und Spurweite von Pkw verschiedener Segmente

deutet dies die Wahl eines geeigneten Achskonzepts, was im Folgenden genauer erläutert wird.

3.1 Entscheidungshierarchie zu Fahrzeugarchitektur, -segmenten und -derivaten

Die Hauptaufgabe einer Fahrzeugachse ist, den Reifen in jeder Fahrsituation in der gewünschten Position und Orientierung zu halten, sowie geforderte Freiheitsgrade zu gewährleisten [1]. Rad und Reifen müssen sich stets im Radhaus befinden, ohne mit der Karosserie oder anderen Fahrwerksbauteilen zu kollidieren. Um hohe Fahrsicherheit und Fahrkomfort sicherzustellen, muss das Rad ein- und ausfedern können. Damit das Fahrzeug die Fahrtrichtung ändern kann, muss das Rad um die Vertikale drehbar sein. Folglich müssen bei einer gelenkten Achse vier und bei einer ungelenkten Achse fünf von sechs möglichen translatorischen und rotatorischen Freiheitsgraden durch geeignete Bindungen gesperrt werden. Ausgehend von diesen Randbedingungen haben sich im Laufe der Jahrzehnte Achskonzepte mit jeweils unterschiedlichen Typen und Anzahlen von Gelenken und Lenkern etabliert. Eine Übersicht über gängige Achsen im Automobilbau geben Heißing et al. [38].

Nun ist es denkbar, unterschiedliche Achskonzepte in einer Fahrzeugfamilie einzusetzen, beispielsweise eine günstige einfache Variante für Fahrzeuge eines preissensiblen Segments und eine teurere aufwändige Variante für höherwertige Fahrzeuge. Allerdings implizieren unterschiedliche Achskonzepte große Anpassungen an nahezu allen Schnittstellen, zum einen aus Gründen des Bauraums, zum anderen, weil unterschiedliche Achskonzepte zumeist auch anders orientierte Gummilager mit unterschiedlichen Eigenschaften sowie unterschiedlich viele Gelenke mit abweichenden Gelenkpositionen benötigen, um optimal zu funktionieren. In Abb. 3.3 ist schematisch dargestellt, wie zwei unterschiedliche Achskonzepte unter der Annahme, dass die karosserieseitigen Anbindungspunkte des Fahrwerks die Schnittstellen der Architektur und somit vereinheitlicht sind, in zwei Derivaten verbaut werden. Die Blitze (↯) kennzeichnen hier Problemstellen, die durch falsch positionierte Anbindungen oder orientierte Gummilager entstehen. Beispielsweise wird bei einer McPherson-Achse in Abb. 3.3b aufgrund

des durch Querkraft belasteten, stärker ausgeführten Federbeins eine weiter außenliegende Anbindung an die Karosserie notwendig. Weitere Anbindungspunkte, wie die des oberen Querlenkers der Vierlenkerachse in Abb. 3.3a, werden gar nicht benötigt. Zudem wird die Belastungsrichtung des Komfortlagers, siehe unterer Blitz in Abb. 3.3b, typischerweise mit einer in Fahrzeugquerrichtung ausgerichteten Drehachse ausgeführt, während diese bei der Vierlenkerachse normal zur Lenkerrichtung orientiert ist.

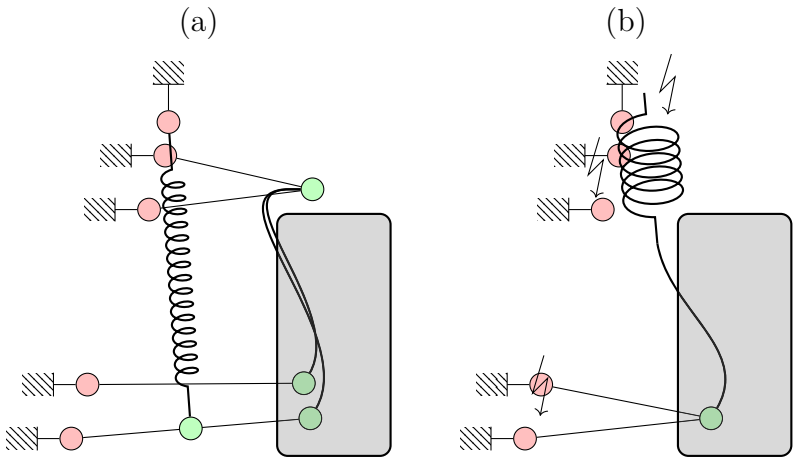

Abbildung 3.3: Verwendung von (a) Vierlenkerachse und (b) McPherson-Achse in einer Fahrwerksarchitektur mit Problemstellen (ϟ)

Zusätzlich zu den angesprochenen Problemen geht der Grundgedanke der Vereinheitlichung mit zwei komplett unterschiedlichen Achskonzepten verloren, so dass der Vorteil einer günstigeren Achse aufgrund der kostspieligen Bauteilvielfalt in Entwicklung und Produktion nicht genutzt werden kann. Hier wird deutlich, dass es sich bei einer Fahrwerksarchitektur um eine integrale Architektur handelt, es also nicht ohne größere Qualitätsverluste möglich ist, verschiedene Achskonzepte im Hinblick einer Modulstrategie auszutauschen. Folglich muss die Achstopologie a priori definiert sein, um den Anteil standardisierter Komponenten zu vergrößern.

Bei einer starren Architektur ist eine einheitliche Achstopologie Voraussetzung, da die Fahrzeugfamilie nur ein Segment bedient und die Derivate durch unterschiedliche Hüte differenziert werden. Die Architektur umfasst hierbei das

gesamte Fahrwerk, da keine segmentspezifischen Adaptionen notwendig sind. Das Ziel einer flexiblen Architektur ist, mehrere Segmente mit einer Fahrzeugfamilie zu bedienen. Das heißt, dass Fahrwerke mit unterschiedlichen Radständen und Spurweiten von der Architektur abgeleitet werden müssen. Die Frage ist, welche Komponenten und Bauteile der Achse vereinheitlicht werden können, ohne große Kompromisse beim Fahrverhalten der Fahrzeuge der entstehenden Familie eingehen zu müssen. Hierfür bieten sich verschiedene Möglichkeiten an.

Schuh et al. [71] haben einige Kombinationen von vereinheitlichten und spezifischen Elementen einer Fahrwerksarchitektur für Elektrofahrzeuge mit unterschiedlichen Spurweiten analysiert. Laut ihren Erkenntnissen ist es der einfachste Weg, die Achsschenkel oder die Radlager zu variieren und Fahrschemel und Lenker zu vereinheitlichen. Dieses Vorgehen hat allerdings den Nachteil, dass sich einige charakteristische Größen der Starrkinematik, insbesondere an der gelenkten Achse, verschlechtern. Beispiele hierfür sind der Störkrafthebelarm und der Lenkrollradius, die die Lenkreaktion bei ungleichen Radbrems- oder Antriebskräften beeinflussen und möglichst klein sein sollten [51].

Eine andere Herangehensweise ist die variable Gestaltung des Fahrschemels bzw. Achsträgers. Beispielsweise kann ein vereinheitlichtes Mittelstück mit variablen Außenteilen verbunden werden, um unterschiedliche Spurweiten zu ermöglichen [38]. Bei gleichlangen Lenkern muss bei breiten Fahrzeugen dann ein Kompromiss bezüglich der Spurweitenänderung bei Ein- und Ausfederung und der Lage des Rollzentrums eingegangen werden [86]. Zudem handelt es sich beim Fahrschemel um ein relativ großes und kostspieliges Bauteil. Eine geeignetere Lösung wäre also, den Fahrschemel zu vereinheitlichen und variable Lenker zu gestalten. Diese These wird dadurch gestützt, dass die Herstellung unterschiedlicher Lenker bei großen Stückzahlen über den Produktlebenszyklus nicht zwangsläufig zu höheren Kosten führt. Dies liegt daran, dass Schmiedewerkzeuge, die üblicherweise zur Herstellung von Lenkern im gehobenen Fahrzeugsegment eingesetzt werden, aufgrund des hohen Verschleißes mehrere Male innerhalb eines Produktlebenszyklus erneuert werden müssen. Daher ist zu vermuten, dass es vor dem Hintergrund heutiger CNC-gesteuerter Fertigung keinen großen Unterschied macht, ob ein einheitliches Werkzeug mehrmalig

erneuert wird, oder ob mehrere unterschiedliche Werkzeuge über den gesamten Lebenszyklus eingesetzt werden. Fahrschemel werden dagegen häufig aus tiefgezogenen Halbzeugen zusammengeschweißt, wobei Tiefziehwerkzeuge auch für große Stückzahlen ohne Erneuerung über den gesamten Lebenszyklus eingesetzt werden können. Folglich bietet sich ein vereinheitlichter Fahrschemel an, da so nur ein Werkzeug und eine einheitliche Schweißlinie benötigt werden.

Zusammenfassend wird eine in Abb. 3.4 schematisch dargestellte Struktur mit architektur-, segment- und derivatrelevanten Elementen für eine flexible Architektur vorgeschlagen. Die Fahrwerksarchitektur wird durch die Anbindungspunkte der Lenker an Fahrschemel und Karosserie definiert. Die achsschenkelseitigen Kinematikpunkte können, je nach Segment, variabel gewählt werden, um Größenunterschiede auszugleichen. Es ergeben sich unterschiedliche Lenker, die Schnittstellen von Fahrschemel, Karosserie und Lenkung sind allerdings vereinheitlicht, wodurch ein segmentübergreifender modularer Einsatz dieser Komponenten ermöglicht wird. Kleinere Anpassungen innerhalb eines Segments, wie beispielsweise geringfügig unterschiedliche Spurweiten zweier Derivate, können durch Anpassung von Radlager oder Einpresstiefe der Räder erreicht werden. Auf diese Weise kann ein guter Kompromiss aus Vereinheitlichung und Auslegungsfreiheit realisiert werden.

3.2 Einführung einer geeigneten Parametrik für Fahrwerksarchitekturen

Eine Fahrzeugfamilie kann, je nachdem wie viele Segmente und Derivate bedient werden, extrem variantenreich sein. Um diese Komplexität auf eine vereinfachte Struktur herunterzubrechen und parametrisch beschreiben zu können, bedarf es einer geeigneten Parametrik. Diese muss die Beschreibung der gesamten Architektur und aller relevanten Elemente berücksichtigen, um im Rahmen computerbasierter Prozesse ansteuerbar und optimierbar zu sein.

Zunächst muss festgelegt werden, wie umfangreich die Parametrik sein muss. Die grundlegende Struktur einer Architektur und der daraus entwickelten Fahrzeugfamilie ist bereits bekannt. So müssen die in Abb. 3.4 aufgezeigten

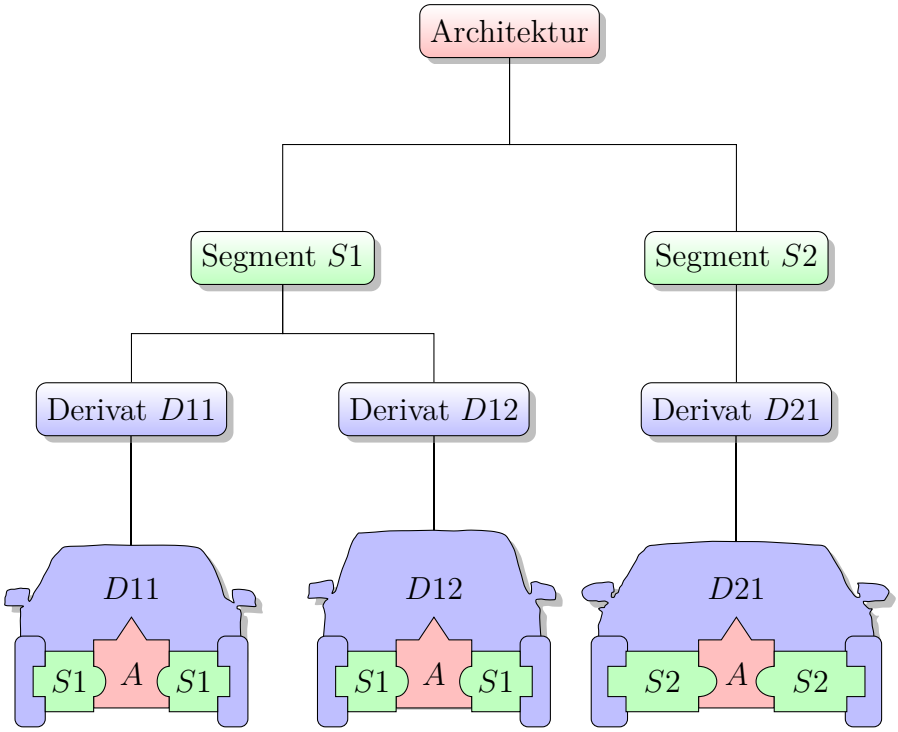

Abbildung 3.4: Hierarchische Struktur einer flexiblen Fahrwerksarchitektur A mit zwei Segmenten $S1$ und $S2$ und insgesamt drei unterschiedlichen Derivaten $D11$, $D12$ und $D21$

Elemente durch voneinander abgrenzbare Typen von Parametern beschrieben werden können. Da angenommen wird, dass die architekturrelevanten Elemente a priori bekannt sind, können diese direkt durch eine notwendige Anzahl von Parametern beschrieben werden. Diese Parameter werden einmalig definiert, da sie für alle auf dieser Architektur basierenden Fahrzeuge gelten. Der optimale Entwurf einer Architektur ist das Ziel der Entwicklung, typischerweise sind die ihr zugeordneten Parameter also Entwurfsvariablen.

Die Segmente werden, wie in der Einleitung dieses Kapitels erwähnt, bereits frühzeitig in der Entwicklung durch die Wahl von Größen wie Radstand und Spurweite definiert. Innerhalb dieser so definierten Segmente gibt es Elemente, die, den bisherigen Erläuterungen folgend, identisch für alle Derivate eines Segments sind. Die Fahrzeugfamilie enthält also Parameter, die die gleiche Funktion haben, aber in unterschiedlichen Segmenten unterschiedliche Werte annehmen können. Im Prinzip lassen sich diese Parameter als lokale Parameter bezeichnen, während architekturrelevante globale Parameter sind. Die segmentrelevanten Parameter sind weitestgehend Entwurfsvariablen, da auch sie im Rahmen der Entwicklung festgelegt werden müssen. Sind aufgrund fertigungstechnischer oder vertriebsrelevanter Anforderungen Werte wie ein fester Radstand vorgegeben, so werden diese im Rahmen der Entwicklung nicht verändert. Für eine computerbasierte Optimierung bedeutet dies, dass solche Parameter konstant sind.

Die Definition derivatbeschreibender Parameter gestaltet sich schwieriger. Dass Derivate sich üblicherweise in der Karosserieform unterscheiden, wurde bereits in Abschnitt 1.1 festgestellt. Ein fahrfertiges Fahrzeug besteht aber aus weiteren, zumeist vom Kunden wählbaren Komponenten, wie beispielsweise unterschiedlichen Motorisierungen oder Sonderausstattungen. In Abb. 3.5 ist die hierarchische Struktur einer Fahrzeugfamilie unter Berücksichtigung wichtiger wählbarer Ausstattungen abgebildet. Nun könnte nach dem gleichen Vorgehen wie bei Segmenten verfahren werden, um auch für Derivate und untergeordnete Varianten Parametertypen zu definieren. Dass aber nicht jede Variante explizit innerhalb einer Optimierung betrachtet werden kann und sollte, wird anhand folgenden Beispiels verdeutlicht.

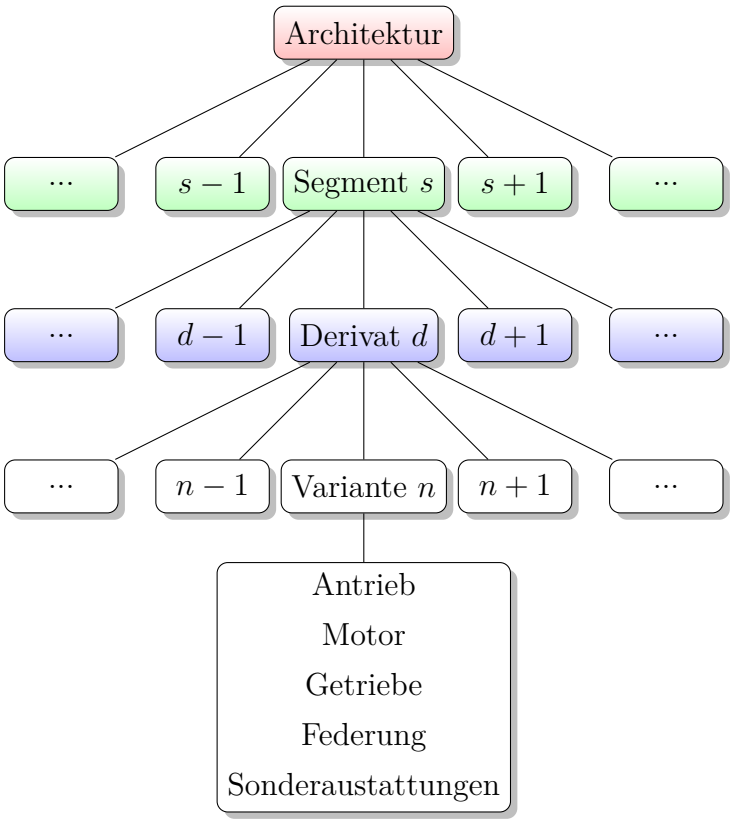

Abbildung 3.5: Hierarchische Darstellung der Vielfältigkeit von Fahrzeugmodellen einer Fahrzeugfamilie

Aus Verkaufsprospekten und Internetauftritten verschiedener Automobilhersteller lässt sich die in Abb. 3.5 dargestellte Variantenvielfalt abschätzen. Mercedes-Benz bietet beispielsweise die C-Klasse im Mittelklasse-Segment an, wobei von ihr insgesamt vier verschiedene Derivate angeboten werden, von der klassischen Limousine bis hin zum exklusiven Cabrio. Jedes dieser Derivate lässt sich mit unterschiedlichen Ausstattungen konfigurieren. Beispielhaft seien hier die wichtigsten Ausstattungen erwähnt, die das Fahrverhalten eines Pkw maßgeblich beeinflussen können. Für die klassische Limousine gibt es etwa 20 verschiedene Antriebsvarianten, die sich jeweils aus der Kombination unterschiedlicher Motoren und Getriebe ergeben, drei verschiedene Fahrwerkssysteme und/oder -abstimmungen, sieben Reifen sowie Sonderausstattungen wie ein Panoramadach und zwei unterschiedliche Tankgrößen. Alleine aus diesen Wahlmöglichkeiten ergeben sich für die vier Derivate $4 \cdot 20 \cdot 3 \cdot 7 \cdot 2 \cdot 2 = 6720$

unterschiedliche Varianten. Im Hinblick auf die in Abb. 1.3 dargestellte MRA-Fahrzeugfamilie ergeben sich unter Annahme gleicher Anzahlen von Varianten für die drei bedienten Segmente eine Gesamtzahl von $6720 \cdot 3 = 20\,160$ Varianten. Jede dieser Varianten müsste, um einen möglichst vollständigen Überblick über die Leistungsfähigkeit der Architektur zu gewinnen, berechnet werden, um einen Architekturentwurf zu bewerten. Selbst die einmalige Berechnung dieser Anzahl von Varianten ist bei Zugrundelegung eines komplexen MKS-Modells und mehreren Fahrmanövern nicht realistisch und wird auch in der Entwicklung nicht praktiziert. Vielmehr wird eine ausgewählte Variante eines Derivates als Entwicklungsfahrzeug ausgewiesen und nur besonders kritische Ausstattungskombinationen zusätzlich betrachtet.

Dieses Vorgehen soll auch für die in der Optimierung verwendete Parametrik angewendet werden. Damit gibt es derivatbeschreibende Parameter, die sich auf eine bestimmte Variante eines Derivats beziehen. Beispielsweise ist es ausreichend, die gesamte Aufbaumasse mit nur einem Parameter zu beschreiben, da sie sich aus der Summe einzelner Massen, wie beispielsweise der Masse des Panoramadachs, zusammensetzt. Zu beachten ist, dass die Derivate auf die von Architektur und Segment definierten Fahrwerksschnittstellen zurückgreifen. Überwiegend sind diese Parameter also keine Entwurfsvariablen, sondern konstante Parameter, die entweder bekannt oder abgeschätzt sind, allerdings können sie variieren und sind dann als unsichere Parameter zu betrachten.

Zusammenfassend wird für die nachfolgenden Analysen eine Parametrik definiert, die aus folgenden Typen von Variablen besteht:

Architekturvariablen *gelten einheitlich für alle Derivate aller Segmente, die auf der gleichen Architektur basieren, und sind durch ein hochgestelltes A gekennzeichnet:* $\bullet^{(A)}$.

Segmentvariablen *gelten einheitlich für alle Derivate eines Segmentes und sind durch eine hochgestellte Zählvariable s für das jeweilige Segment gekennzeichnet:* $\bullet^{(s)}, s = 1...S$.

Derivatvariablen *gelten spezifisch für ein Derivat eines Segmentes und sind durch eine hochgestellte Zählvariable d für das jeweilige Derivat und s für*

das Segment gekennzeichnet: $\bullet^{(s,d)}, d = 1...D$*, wobei* D *von* s *abhängen kann.*

Hier wird bewusst von Variablen gesprochen, da wie oben beschrieben davon ausgegangen werden muss, dass jede dieser Größen eine Entwurfsvariable, eine Konstante oder, wie im weiteren Verlauf der Arbeit diskutiert wird, eine unsicherheitsbehaftete bzw. streuende Größe sein kann. Die unterschiedlichen Typen von Variablen werden zur besseren Unterscheidbarkeit in Abbildungen in rot für Architekturvariablen, grün für Segmentvariablen und blau für Derivatvariablen dargestellt. Diese Farbcodierung dient in der gesamten Arbeit ebenfalls in schematischen Darstellungen und Abbildungen zur Unterscheidung der drei Typen.

3.3 Unterschiedliche Konzepte der Architekturoptimierung

Mithilfe der definierten Parametrik können nun Konzepte entwickelt werden, mit denen eine automatisierte Architekturoptimierung durchführbar ist. Wie in Abschnitt 1.2 beschrieben ist, gibt es bereits einige Konzepte, die in der Literatur diskutiert werden und auch hier Berücksichtigung finden. Zunächst wird aber das konventionelle Vorgehen der Entwicklungsabteilungen bei der Architekturentwicklung betrachtet.

Nach Abschnitt 2.1 gibt es Meilensteine, bei denen der vorhandene Entwicklungsstand simulativ bewertet wird. Wenn zwischen zwei Meilensteinen ein architekturrelevanter Parameter $p^{(A)}$ geändert wird, so muss jedes Fahrzeug, das von dieser Änderung betroffen ist, neu bewertet werden. Bei Änderungen von segment- oder derivatrelevanten Parametern $p^{(s)}$ bzw. $p^{(s,d)}$ ist es hingegen ausreichend, nur die bzw. das davon betroffene Derivat(e) neu zu bewerten. Die konventionelle Entwicklung folgt also einem iterativen Prozess, wobei die Charakteristika der jeweiligen Parametertypen Berücksichtigung finden. Dieses Vorgehen ähnelt dem allgemeinen Ablauf einer automatisierten Entwurfsoptimierung, wie sie nach Bestle [6] in Abb. 3.6 dargestellt ist. Aus dem konventionellen Vorgehen und dem Ablauf einer automatisierten Optimie-

rung lassen sich verschiedene in Abb. 3.7 dargestellte Konzepte ableiten, die nachfolgend erläutert und diskutiert werden.

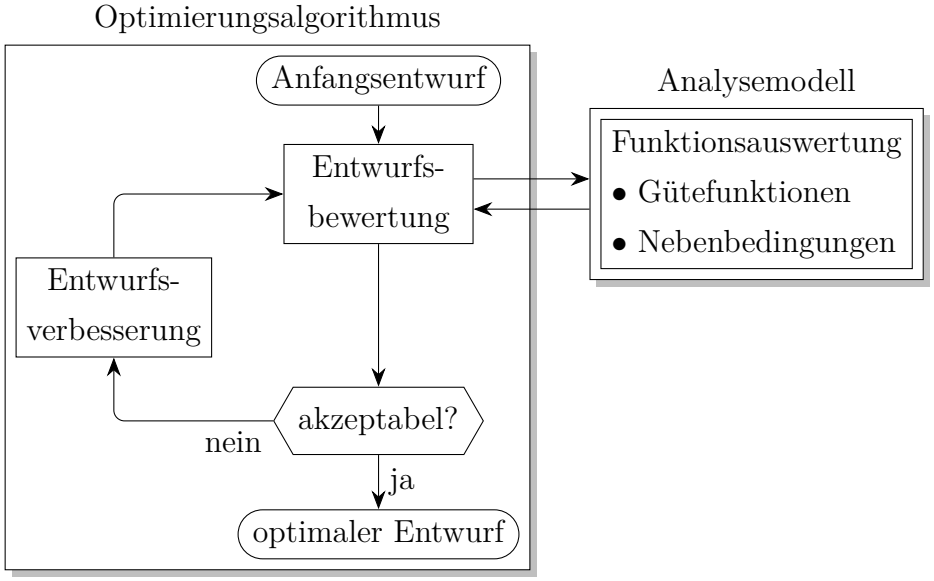

Abbildung 3.6: Allgemeiner Ablauf einer automatisierten Entwurfsoptimierung

Das erste Konzept in Abb. 3.7a wird als Architekturoptimierung mit Segmentnachoptimierungen bezeichnet. In dieser Anordnung, die auch als kaskadierte Optimierung bezeichnet wird und unter anderem bereits von Friedrich und Menzel [31] diskutiert wurde, kommen zwei stufenförmig angeordnete Optimierungsalgorithmen zum Einsatz. Die erste Stufe hat nur Zugriff auf die Entwurfsvariablen $\boldsymbol{p}^{(A)}$ der Architektur. Ein vorgeschlagener Architekturentwurf wird entsprechend Abb. 3.6 durch die Auswertung eines Analysemodells bewertet, die hier allerdings selbst eine Optimierung beinhaltet. Die kaskadierte Architekturoptimierung besteht folglich aus einem äußeren Optimierungsproblem, das allgemein als

$$\min_{\boldsymbol{p}^{(A)} \in P^{(A)}} f^{(A)}\left(\boldsymbol{p}^{(A)}\right) \text{ mit } P^{(A)} = \left\{ \boldsymbol{p}^{(A)} \in \mathbb{R}^{h^{(A)}} \middle| \boldsymbol{p}_u^{(A)} \leq \boldsymbol{p}^{(A)} \leq \boldsymbol{p}_o^{(A)} \right\}, \tag{3.1}$$

mit einem zu optimierenden Architekturkriterium $f^{(A)}$ und einem aus unteren und oberen Grenzen $\boldsymbol{p}_u^{(A)}$ und $\boldsymbol{p}_o^{(A)}$ und Dimension $h^{(A)}$ aufgespannten Entwurfsraum $P^{(A)}$ formuliert werden kann. Das Analysemodell des Architekturoptimierers ist ein Optimierer der zweiten Stufe für die Segmentkriterien

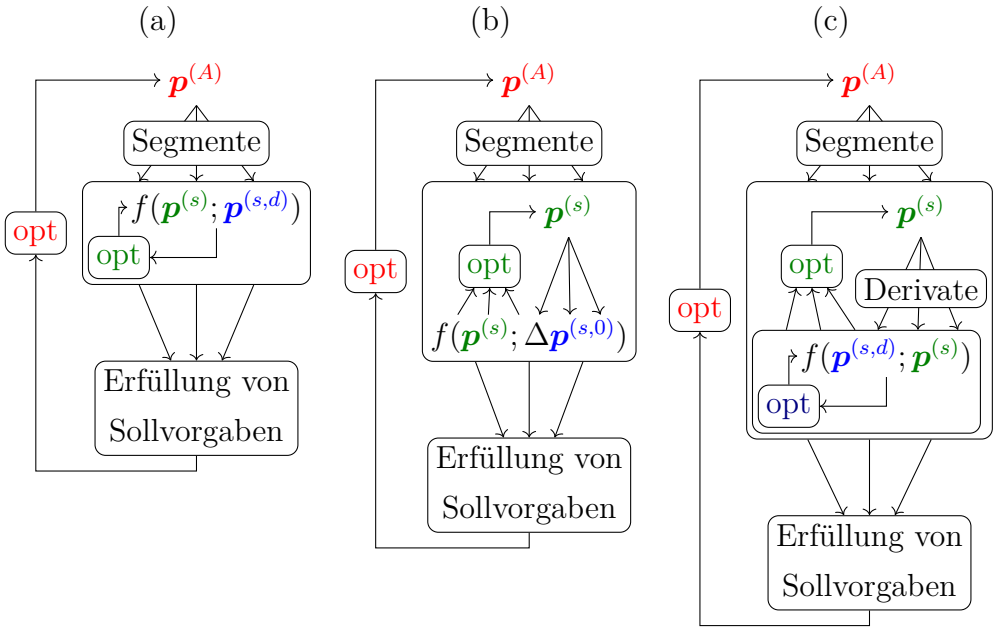

Abbildung 3.7: Konzepte der Architekturoptimierung mit (a) Segmentnachoptimierungen, (b) robusten Segmentnachoptimierungen und (c) Segment- und Derivatnachoptimierungen

$f^{(s)}$ und hat dabei nur Zugriff auf die Entwurfsvariablen $\boldsymbol{p}^{(s)}$ der Segmente s. Für jeden Architekturentwurf $\boldsymbol{p}^{(A)}$ werden also S innere Probleme

$$\min_{\boldsymbol{p}^{(s)}\in P^{(s)}} f^{(s)}\left(\boldsymbol{p}^{(s)};\boldsymbol{p}^{(A)}\right) \text{ mit } P^{(s)} = \left\{\boldsymbol{p}^{(s)} \in \mathbb{R}^{h^{(s)}} \middle| \boldsymbol{p}_l^{(s)} \leq \boldsymbol{p}^{(s)} \leq \boldsymbol{p}_u^{(s)}\right\}, \tag{3.2}$$

$s = 1...S$, gelöst. Bei dem an den Segmentoptimierer angebundenen Analysemodell werden konkrete Funktionsauswertungen, also Simulationen von Fahrzeugmodellen durchgeführt und anhand gegebener Sollvorgaben in Form von Kriterien

$$f^{(s,d)} = f\left(\boldsymbol{p}^{(s)},\boldsymbol{p}^{(A)},\boldsymbol{p}^{(s,d)}\right) \tag{3.3}$$

bewertet, die zu den oben genannten Segmentkriterien $f^{(s)}$ kombiniert werden. Hierfür werden die Derivat-beschreibenden Parameter $\boldsymbol{p}^{(s,d)}$ benötigt. Ein Derivat d entspricht dem aus den oben genannten Gründen eingeführten Auslegungsderivat, also einer vordefinierten Austattungskombination. Kurz zusammengefasst, wird für jeden Architekturentwurf versucht, die besten Segmentvariablen für alle Derivate zu bestimmen, damit jedes Segment gegebene Sollvorgaben bestmöglich erfüllt. Dieses Konzept wird in Abschnitt 4.1.3 genauer untersucht.

Das zweite Konzept in Abb. 3.7b basiert auf robusten Segmentnachoptimierungen. Die bisher als konstant definierten Derivatparameter sind hierbei streuende bzw. unsichere stochastische Variablen $\Delta\boldsymbol{p}^{(s,0)}$. In diesem Fall wird $d = 0$ gesetzt, um zu verdeutlichen, dass im Prinzip kein konkretes Derivat existiert, sondern diese ausgehend von einem *Null-Derivat* aus stochastisch erstellten Kombinationen abgeleitet werden. Dieses *Null-Derivat* kann auch als Entwicklungsderivat gesehen werden, von dem die anderen Derivate abgeleitet werden, und ist damit das einzige explizite Derivat eines Segments. Eine genauere Erläuterung dieser Herangehensweise gibt es in Abschnitt 4.1.1. Analog zu (3.1) wird im äußeren Kreislauf ein nun als Robustheitskriterum für die Architektur bezeichnetes Kriterium optimiert:

$$\min_{\boldsymbol{p}^{(A)} \in P^{(A)}} f_{RDO}^{(A)} \left(\boldsymbol{p}^{(A)} \right) \text{ mit } P^{(A)} = \left\{ \boldsymbol{p}^{(A)} \in \mathbb{R}^{h^{(A)}} \middle| \boldsymbol{p}_u^{(A)} \leq \boldsymbol{p}^{(A)} \leq \boldsymbol{p}_o^{(A)} \right\}. \tag{3.4}$$

Da es nur ein Derivat $d = 0$ pro Segment gibt, gilt $f^{(s,d)} = f^{(s,0)} = f^{(s)}$, womit das Optimierungsproblem (3.2) als robustes Problem

$$\min_{\boldsymbol{p}^{(s)} \in P^{(s)}} f^{(s,0)} \left(\boldsymbol{p}^{(s)}; \boldsymbol{p}^{(A)}, \Delta\boldsymbol{p}^{(s,0)} \right) \text{ mit } P^{(s)} = \left\{ \boldsymbol{p}^{(s)} \in \mathbb{R}^{h^{(s)}} \middle| \boldsymbol{p}_u^{(s)} \leq \boldsymbol{p}^{(s)} \leq \boldsymbol{p}_o^{(s)} \right\} \tag{3.5}$$

formuliert werden kann. Das Ziel ist hier, die Robustheit des Systems gegenüber den streuenden Variablen $\Delta\boldsymbol{p}^{(s,0)}$ zu optimieren.

Die Architekturoptimierung mit robusten Segmentnachoptimierungen kann auf verschiedene Weise interpretiert werden. Zum einen sind in der frühen Entwicklungsphase noch nicht alle Derivate bzw. deren Eigenschaften bekannt. Gegebenenfalls gibt es Derivate, deren Erscheinen noch unsicher ist. Die Architektur sollte aber robust genug sein, damit auch solche Derivate ihre Sollvorgaben hinreichend erfüllen. Der zweite Gedanke ist die oben angesprochene Vielfalt von Varianten, die für jedes Derivat in Frage kommen. Durch die Streuung der Derivatvariablen kann eine Architektur robust gegenüber dieser Variantenvielfalt ausgelegt werden, ohne dass jede Variantenkombination explizit berechnet werden oder bekannt sein muss. Zu guter Letzt ist auch eine der klassischen RDO entsprechende Nutzung möglich. Die Robustheit gegenüber Fertigungstoleranzen oder dem Nutzungsverhalten des Kunden steht

dabei im Vordergrund. Als unsicheres Nutzungsverhalten ist beispielsweise der Beladungszustand oder nachträglich aufgezogene Reifen gemeint, wodurch das Fahrzeugverhalten stark von der ursprünglichen Idealauslegung abweichen kann. Die robuste Architekturoptimierung wird in Kapitel 5 diskutiert.

Die Idee der mehrstufigen Optimierung kann weitergeführt werden, indem die Derivate in einer eigenen Stufe berücksichtigt werden. Es ergibt sich die in Abb. 3.7c dargestellte Struktur einer Architekturoptimierung mit Segment- und Derivatnachoptimierungen. Wenn Derivatnachoptimierungen Teil des Optimierungsproblems sind, so gibt es auch Entwurfsvariablen der Derivate. Unterschiedliche Derivate haben üblicherweise auf ihre Schwerpunktlage und Masse angepasste Aufbaufederdung und -dämpfung. Diese Größen können also Entwurfsvariablen der Derivate sein. Letztendlich ergibt sich dadurch ein dreistufiger Optimierungsprozess. Die Verschachtelung von Optimierungsalgorithmen ist allerdings ab einem gewissen Grad nicht mehr praktikabel, da die Anzahl notwendiger Funktionsauswertungen immens groß wird. Dieses Konzept stellt den umfassendsten Ansatz einer Architekturoptimierung dar, wird aber aufgrund des hohen Zeitaufwands in dieser Arbeit nicht weiter betrachtet.

3.4 Implementierung einer automatisierten Architekturoptimierung

Damit eine Architekturoptimierung in den laufenden Entwicklungsprozess integriert werden kann, wird ein Prozess benötigt, der die existierende Softwareinfrastruktur einer Entwicklungsabteilung berücksichtigt. Dies ist insofern vorteilhaft, da sämtliche Fahrzeugmodelle in dieser Umgebung bereitgestellt und gepflegt werden und die notwendigen Fahrmanöver und Auswertekriterien definiert sind.

Die in dieser Arbeit durchgeführten Analysen werden mit der kommerziellen Optimierungssoftware *optiSLang*® [23] durchgeführt, für eine beispielhafte Umsetzung siehe Ubben et al. [88]. Die Berechnungen der Gesamtfahrzeugmodelle findet in dem MKS-Tool *LMS Virtual.Lab*™ [76] statt. Beide Programme können allerdings nicht unmittelbar miteinander kommunizieren. Daher ist die

Entwicklung eines effizienten Prozesses zur Weitergabe von Entwurfsvariablen und Rückführung von Kriterien notwendig, um eine automatisierte Architekturoptimierung durchführen zu können. Der entwickelte Prozess wird anhand des Flussdiagramms in Abb. 3.8 kurz erläutert.

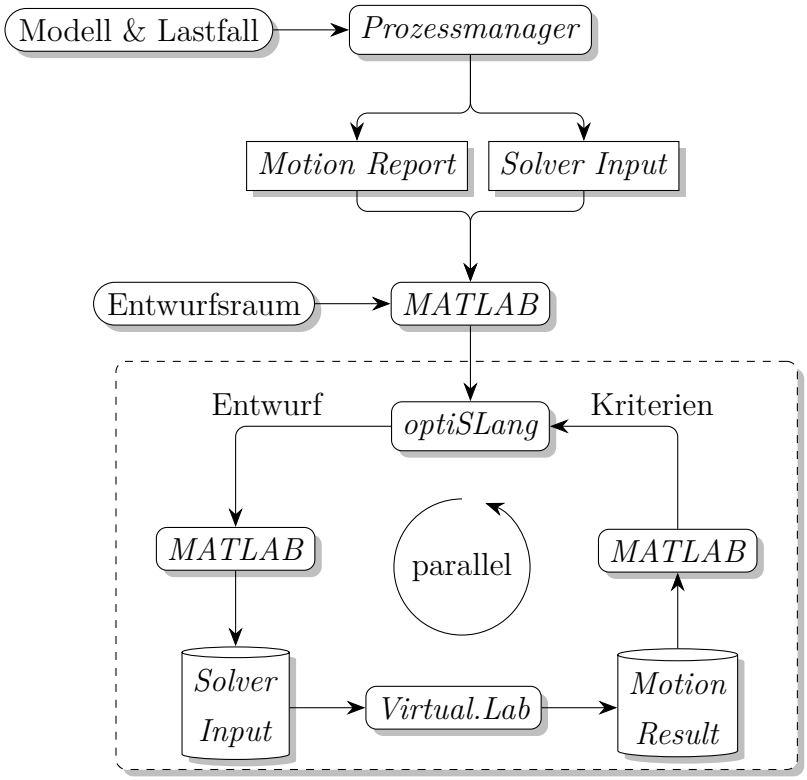

Abbildung 3.8: Flussdiagramm der Softwareinfrastruktur für die automatisierte Optimierung

Zunächst werden ein Modell und ein Lastfall definiert und in der Simulationsumgebung aufgebaut. Dies geschieht in dem sogenannten *Prozessmanager*. Prinzipiell liegen verschiedene Achstopologien und weitere Baugruppen als Module vor, die mithilfe des *Prozessmanagers* zusammengesetzt und mit den Daten des aktuellen Fahrzeugmodells versehen werden. Das nun in dem MKS-Tool geladene Modell kann in Form von Textdateien, dem sogenannten *Motion Report* und dem *Solver Input*, exportiert werden. Diese Dateien genügen, um das Modell in seiner Gesamtheit zu beschreiben. Sie beinhalten die jeweiligen Drehmatrizen und Trägheiten der Körper, Bindungselemente und weitere wichtige Informationen, wie das verwendete Reifenmodell. Diese Daten werden

automatisiert ausgelesen und als Modell-Objekte in *MATLAB*® [85] verarbeitet. Mithilfe eines gewünschten Entwurfsraums wird das Modell vorbereitet und der in Abb. 3.6 dargestellte Optimierungsprozess beginnt. Bevor näher auf den Optimierungskreislauf eingegangen wird, erfolgt eine detailliertere Beschreibung der Modellstruktur, um den Grund des Einsatzes von *MATLAB* als zusätzliche Instanz einordnen zu können.

Um besonders effizient zu sein, werden die geänderten Entwurfsvariablen direkt in die *Solver Input*-Datei eingefügt. Dies hat den Vorteil, dass nicht bei jeder Änderung einer Variablen das komplette Fahrzeugmodell aus den verschiedenen Modulen erneut zusammengesetzt und die neuen Variablenwerte eingefügt werden müssen, sondern lediglich die sich neu ergebenden Koordinaten und Drehmatrizen berechnet und an die jeweilige Stelle in der *Solver Input*-Datei eingetragen werden. Der zeitintensive Modellaufbau wird dadurch als Preprocessing vorgezogen und muss nicht bei jedem neuen Entwurf im Optimierungskreislauf durchgeführt werden. Zusätzlich können mehrere verschiedene Fahrzeugmodelle oder Lastfälle zeitgleich innerhalb einer Optimierung verwendet werden, da die notwendigen Dateien im Vorfeld erstellt wurden.

Um die Anzahl der notwendigen Berechnungs- und Manipulationsschritte zu reduzieren, wird vorab geprüft, welche Elemente von den ausgewählten Entwurfsvariablen beeinflusst werden. Nur die diese Elemente betreffenden Ortsvektoren, Drehmatrizen oder Steifigkeits- und Dämpfungsparameter werden angepasst. Bei Gummilagern werden die in Abschnitt 2.4.2 dargestellten äquivalenten Steifigkeits- und Dämpfungswerte mit (2.34) und (2.35) berechnet. Die Bestimmung der Drehmatrizen und Koordinaten der Körper und Hilfskoordinatensysteme erfolgt nach den Regeln der Lagebeschreibung eines starren Körpers und wird im Folgenden näher erläutert.

Generell wird ein Körper K_i durch Drehmatrix $\boldsymbol{S}_i$ und Ortsvektor $\boldsymbol{r}_i$ in einem intertialen Koordinatensystem O_I beschrieben, vgl. Abschnitt 2.4.1. Bei der vorliegenden Modellierung wird ein Körper durch ein sich nicht notwendig im Schwerpunkt befindendes körperfestes Non-Centroidal Body Fixed (NCBF)-Koordinatensystem definiert. Weitere, diesen Körper betreffende Punkte und

Koordinatensysteme, wie unter anderem der Schwerpunkt, werden ebenfalls in diesem NCBF-System angegeben. Die oben erwähnte *Solver Input*-Datei enthält diese Informationen über Koordinaten und Orientierungen im jeweilig zugeordneten körperfesten Koordinatensystem. Üblicherweise werden in der Fahrwerksentwicklung die Positionen von Gelenken und Körpern jedoch im Inertialsystem angegeben und müssen daher transformiert werden, um der Struktur der *Solver Input*-Datei zu entsprechen.

Um die Lage eines Körpers K_i gegenüber dem Intertialsystem $\{O_I; x_I, y_I, z_I\}$ eindeutig zu definieren, sind drei Ortsvektoren zu drei körperfesten Punkten P_i, Q_i und R_i ausreichend [64]. Der Ursprung eines körperfesten Koordinatensystems C_i wird über den Punkt P_i definiert, wodurch sich der Ortsvektor $\boldsymbol{r}_i$ zum Koordinatenursprung als Vektor $\overrightarrow{O_I P_i}$ ergibt. Das körperfeste orthogonale Dreibein $\{C_i; x_i, y_i, z_i\}$ kann nun wie folgt berechnet werden. Die z_i-Achse wird über den Vektor $\boldsymbol{r}_{P_iQ_i}$, der von P_i in Richtung Q_i zeigt, als Einheitsvektor

$$\boldsymbol{e}_{z,C_i} = \frac{\boldsymbol{r}_{P_iQ_i}}{\|\boldsymbol{r}_{P_iQ_i}\|} \tag{3.6}$$

definiert. Der Punkt R_i liegt per Definition in der $z_i x_i$-Ebene. Da die y_i-Achse senkrecht auf dieser Ebene steht, ergibt sich der nächste Einheitsvektor mit dem Kreuzprodukt

$$\boldsymbol{e}_{y,C_i} = \frac{\boldsymbol{e}_{z,C_i} \times \boldsymbol{r}_{P_iR_i}}{\|\boldsymbol{e}_{z,C_i} \times \boldsymbol{r}_{P_iR_i}\|}. \tag{3.7}$$

Da alle Achsen orthogonal zueinander stehen, kann die x_i-Achse abschließend durch

$$\boldsymbol{e}_{x,C_i} = \boldsymbol{e}_{y,C_i} \times \boldsymbol{e}_{z,C_i} \tag{3.8}$$

berechnet werden.

Nun kann ein beliebiger, im körperfesten Koordinatensystem C_i angegebener Punkt Q_i, beschrieben durch Koordinaten $\boldsymbol{r}'_{P_iQ_i}$, mit

$$\boldsymbol{r}_{Q_i} = \boldsymbol{r}_i + \boldsymbol{S}_i \boldsymbol{r}'_{P_iQ_i} \tag{3.9}$$

in das Inertialsystem transformiert werden, wobei die Drehmatrix

$$\boldsymbol{S}_i = \begin{bmatrix} \boldsymbol{e}_{x,C_i} & \boldsymbol{e}_{y,C_i} & \boldsymbol{e}_{z,C_i} \end{bmatrix} \tag{3.10}$$

aus den Einheitsvektoren (3.6) - (3.8) aufgebaut ist. Drehmatrizen von orthogonalen Koordinatensystemen genügen der Orthogonalitätsbedingung $\boldsymbol{S}\boldsymbol{S}^T = \boldsymbol{E}$ bzw. $\boldsymbol{S}^{-1} = \boldsymbol{S}^T$ [64]. Mithilfe der Umkehrtransformation kann $\boldsymbol{r}'_{P_iQ_i}$ deshalb aus (3.9) als

$$\boldsymbol{r}'_{P_iQ_i} = \boldsymbol{S}_i^T \left(\boldsymbol{r}_{Q_i} - \boldsymbol{r}_i\right) \tag{3.11}$$

in das körperfeste Koordinatensystem C_i transformiert werden. Mit gegebenen körperfesten Punkten und den daraus folgenden Transformationsmatrizen sowie (3.11) lassen sich sämtliche im Modell vorhandenen Koordinatensysteme und Punkte in Bezug auf ein Referenzmodell anpassen, um einen vom Optimierer vorgeschlagenen Entwurf berechnen und bewerten zu können.

Der eigentliche Optimierungsprozess in Abb. 3.8 wird dann von *optiSLang* gesteuert, wobei jeder zu analysierende Entwurf an die *MATLAB*-Struktur übermittelt wird. Dort werden die oben erläuterten Berechnungen durchgeführt und eine angepasste *Solver Input*-Datei exportiert sowie notwendige Berechnungsdateien bereitgestellt. Das so erstellte Modell wird dann vom MKS-Solver simuliert. Die Ergebnisse der Simulation liegen anschließend als sogenannte *Motion Result*-Datei vor, die von *MATLAB* ausgewertet wird, um die gewünschten objektiven Kriterien zu berechnen. Die Rückführung der Kriterien an *optiSLang* schließt den Kreislauf. Dieser Prozess ist in sich geschlossen und kann zur weiteren Effizienzsteigerung parallel ausgeführt werden.

Kapitel 4

Deterministische Architekturoptimierung mit Segmentnachoptimierung

Ausgehend von den bisherigen Definitionen und Konzepten werden im Folgenden verschiedene Analysen durchgeführt, um unter den gegebenen Gesichtspunkten und Anforderungen die beste Strategie zur automatisierten Architekturoptimierung zu finden. Als Basis dient das einfache und effizient zu berechnende Testproblem aus Abschnitt 2.3. Die verschiedenen Strategien werden miteinander verglichen und effizienzverbessernde Maßnahmen entwickelt.

4.1 Formulierung eines einkriteriellen Architekturoptimierungsproblems

Wie bereits in den vorangegangenen Kapiteln erwähnt wurde, ist das Ziel einer flexiblen Architektur, verschiedene Fahrzeugderivate aus verschiedenen Segmenten unter Verwendung vereinheitlichter Komponenten herstellen zu können. In den folgenden Analysen wird exemplarisch davon ausgegangen, dass eine Firma drei verschiedene Segmente mit Fahrzeugen bedienen möchte, die auf einer gemeinsamen Architektur basieren. Entsprechend der eingeführten Parametrik aus Abschnitt 3.2 gilt $s \in \{1, 2, 3\}$. Für Fahrzeuge zweier Segmente wird das in Abb. 2.2a dargestellte Federungssystem verwendet, Fahrzeuge des

verbleibenden Segments erhalten das komplexere System aus Abb. 2.2b.

Die Architektur wird nach den Erkenntnissen aus Abschnitt 3.1 ausgelegt. Der Fahrschemel und dessen Anbindung an die Karosserie und die zugehörigen Parameter m_F, c_A und d_A der Testmodelle werden also als vereinheitlicht angenommen. Die als Architekturvariable klassifizierten Parameter werden zu einem Architekturentwurfsvektor

$$\boldsymbol{p}^{(A)} = \begin{bmatrix} m_F & c_A & d_A \end{bmatrix}^T \tag{4.1}$$

zusammengefasst.

Die grundlegende Radaufhängung wird für dieses Testproblem als segmentspezifisch angenommen. Die Steifigkeiten und Dämpfungswerte der drei Segmente müssen folglich separat betrachtet werden. Es ergeben sich drei unterschiedliche Segmententwurfsvektoren

$$\boldsymbol{p}^{(1)} = \begin{bmatrix} c_F^{(1)} & d_F^{(1)} \end{bmatrix}^T, \boldsymbol{p}^{(2)} = \begin{bmatrix} c_F^{(2)} & d_F^{(2)} \end{bmatrix}^T, \boldsymbol{p}^{(3)} = \begin{bmatrix} c_F^{(3)} & d_F^{(3)} & c_{F2}^{(3)} & d_{F2}^{(3)} \end{bmatrix}^T. \tag{4.2}$$

Jedes Derivat hat einen einzigartigen Fahrzeugaufbau und spezifische Reifen. Die Derivate unterscheiden sich also durch

$$\boldsymbol{p}^{(s,d)} = \begin{bmatrix} m_A^{(s,d)} & m_R^{(s,d)} & c_R^{(s,d)} \end{bmatrix}^T. \tag{4.3}$$

Die Eigenschaften der Derivate seien vordefiniert, weshalb der Vektor $\boldsymbol{p}^{(s,d)}$ in (4.3) nur Parameter enthält, die während des gesamten Optimierungsprozesses konstant sind. Die möglichen Entwurfsräume beschränken sich folglich auf durch untere und obere Grenzen $\boldsymbol{p}_u$ und $\boldsymbol{p}_o$ beschränkte Bereiche der Architektur- und Segmentvariablen:

$$\begin{aligned} P^{(A)} &= \left\{ \boldsymbol{p}^{(A)} \in \mathbb{R}^{h^{(A)}} \middle| \boldsymbol{p}_u^{(A)} \leq \boldsymbol{p}^{(A)} \leq \boldsymbol{p}_o^{(A)} \right\}, \\ P^{(s)} &= \left\{ \boldsymbol{p}^{(s)} \in \mathbb{R}^{h^{(s)}} \middle| \boldsymbol{p}_u^{(s)} \leq \boldsymbol{p}^{(s)} \leq \boldsymbol{p}_o^{(s)} \right\}, \end{aligned} \tag{4.4}$$

wobei entsprechend Gleichung (4.1) und (4.2) $h^{(A)} = 3$, $h^{(1)} = h^{(2)} = 2$ und $h^{(3)} = 4$ sind.

Wie in Abschnitt 3.3 erwähnt, gibt es sogenannte Auslegungsderivate, die in der Entwicklung als erstes betrachtet werden. Für die nachfolgenden Analysen

wird von bereits erläuterten *Null-Derivaten* ausgegangen, um die Komplexität gering zu halten. In jedem Segment wird also nur ein Derivat betrachtet und durch $d = 0$ gekennzeichnet. Die zugehörigen Grenzen und Parameterwerte des Architekturoptimierungsproblems können Tabelle 4.1 entnommen werden.

	Parameter	Segmente $s=1$	$s=2$	$s=3$
$\boldsymbol{p}^{(s,0)}$	m_A[kg]	250	500	350
	m_R[kg]	30	50	40
	c_R[N mm^{-1}]	200	160	180
$\left[\boldsymbol{p}_u^{(s)} \quad \boldsymbol{p}_o^{(s)}\right]$	c_F[N mm^{-1}]	[14 18]	[30 34]	[21 25]
	d_F[N s mm^{-1}]	[1,2 1,8]	[1,2 1,8]	[1,2 1,8]
	c_{F2}[N mm^{-1}]	-	-	[100 1000]
	d_{F2}[N s mm^{-1}]	-	-	[0,1 3]
$\left[\boldsymbol{p}_u^{(A)} \quad \boldsymbol{p}_o^{(A)}\right]$	m_F[kg]	[20 50]		
	c_A[N mm^{-1}]	[800 2000]		
	d_A[N s mm^{-1}]	[0,1 3]		

Tabelle 4.1: Parameterwerte und -bereiche des Testproblems in Abb. 2.2

Das Entwicklungsziel ist, dass alle Derivate unabhängig von ihrer Masse und Größe ein möglichst ähnliches Fahrverhalten aufweisen. Typischerweise haben erfahrene Fahrzeugentwickler eine sehr klare Vorstellung davon, wie diese Ziele durch die Wahl geeigneter Zielwerte für Schwingungs- und Dämpfungscharakteristiken zu erreichen sind. Für das gegebene Beispiel der zu entwickelnden Architektur werden die von Aufbau und Rad dominierten Eigenfrequenzen f_A^{eig} und f_R^{eig} sowie das zum letzteren gehörende Dämpfungsmaß D_R nach Gleichung (2.17) herangezogen, deren Zuordnung über das MAC nach (2.18) sichergestellt wird. Diese sollen spezifische Zielwerte erreichen, woraus sich folgende zu minimierende, normierte Gütekriterien ergeben:

$$f_1 = \left|\frac{f_A^{eig} - 1{,}1\,\text{Hz}}{1{,}1\,\text{Hz}}\right|, \quad f_2 = \left|\frac{f_R^{eig} - 13\,\text{Hz}}{13\,\text{Hz}}\right| \text{ und } f_3 = \left|\frac{D_R - 0{,}3}{0{,}3}\right|. \tag{4.5}$$

Für jedes Segment s werden die jeweiligen $N = 3$ Gütekriterien (4.5) mit einem

Root Mean Square (RMS) Ansatz zu einem einzelnen Kriterium kombiniert:

$$f^{(s,0)} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} f_n^2}. \tag{4.6}$$

Der Mittelwert aller $S = 3$ Kriterienwerte der Segmente $f^{(s,0)}$ bildet das Architekturkriterium

$$f^{(A)} = \frac{1}{S}\sum_{s=1}^{S} f^{(s,0)}. \tag{4.7}$$

4.1.1 Lösung der Null-Architektur als Referenz

Bevor die verschiedenen Strategien der Architekturoptimierung untersucht werden, wird zunächst die sogenannte Null-Architektur optimiert. Die Null-Architektur beinhaltet keine Architekturvariablen. Es gibt also keine Architektur, sondern jedes Segment wird für sich mit der vollen Auslegungsflexibilität optimiert, ohne Kompromisse einzugehen. Das Ergebnis der Optimierung der Null-Architektur wird als utopische Lösung bezeichnet, da Vereinheitlichungen typischerweise zu einer geringeren, erreichbaren Leistungsfähigkeit der einzelnen Produkte führen. Nach Simpson [81] wird in diesem Zusammenhang auch von dem Zielkonflikt zwischen Vereinheitlichung und individueller Produktleistung gesprochen.

Aufgrund der fehlenden Architektur werden alle Entwurfsvariablen (4.1) und (4.2) zu Segment-Entwurfsvektoren

$$\boldsymbol{q}^{(s)} = \left(\boldsymbol{p}^{(A)}, \boldsymbol{p}^{(s)}\right) = \begin{cases} \begin{bmatrix} m_F^{(s)} & c_A^{(s)} & d_A^{(s)} & c_F^{(s)} & d_F^{(s)} \end{bmatrix}^T & \text{für } s = 1,2 \\ \begin{bmatrix} m_F^{(s)} & c_A^{(s)} & d_A^{(s)} & c_F^{(s)} & d_F^{(s)} & c_{F2}^{(s)} & d_{F2}^{(s)} \end{bmatrix}^T & \text{für } s = 3 \end{cases} \tag{4.8}$$

kombiniert. Basierend auf den Gütekriterien (4.6) und Entwurfsvektoren (4.8) mit zugehörigen Null-Derivat Parametern $\boldsymbol{p}^{(s,0)}$ nach Tabelle 4.1 ergeben sich drei entkoppelte Optimierungsprobleme

$$\min_{\boldsymbol{q}^{(s)} \in Q^{(s)}} f^{(s,0)}\left(\boldsymbol{q}^{(s)}; \boldsymbol{p}^{(s,0)}\right) \text{ mit } Q^{(s)} = \left\{ \boldsymbol{q}^{(s)} \in \mathbb{R}^{h^{(A)}+h^{(s)}} \middle| \boldsymbol{q}_u^{(s)} \leq \boldsymbol{q}^{(s)} \leq \boldsymbol{q}_o^{(s)} \right\}, \tag{4.9}$$

wobei $s \in \{1,2,3\}$ und der mögliche Entwurfsraum $Q^{(s)} = P^{(A)} \times P^{(s)}$ sich jeweils aus dem kartesischen Produkt der Mengen (4.4) ergibt.

Um den globalen Minimierer $\boldsymbol{q}^{(s)*}$ zu finden, wird ein in der kommerziellen Optimierungssoftware *optiSLang* enthaltener EA verwendet. Dieser Algorithmus basiert auf dem ursprünglich von Zitzler et al. [99] entwickelten Strength Pareto Evolutionary Algorithm (SPEA2). Der Algorithmus generiert maximal 90 Generationen mit jeweils 100 neuen Individuen bei einer Startpopulation von 1000 Individuen. Es wird angenommen, dass der Algorithmus nach 10 Generationen ohne Verbesserung des Gütekriteriums konvergiert ist, und die Optimierung wird abgebrochen. Für die Optimierung der Null-Architektur werden insgesamt 6900 Funktionsauswertungen für alle drei Segmente benötigt. Die Ergebnisse sind in Abb. 4.1 in Form von normierten Entwurfsvariablen und Gütekriterien dargestellt.

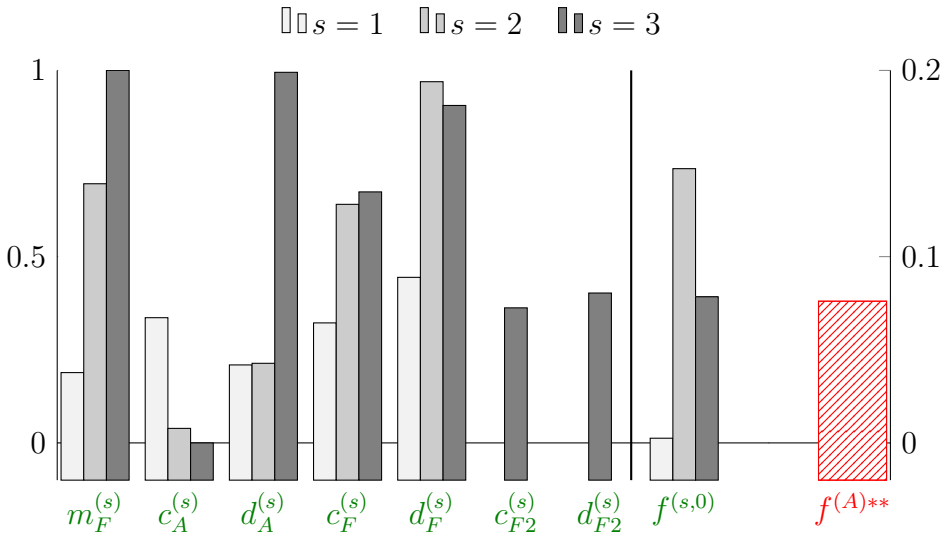

Abbildung 4.1: Optimale Ergebnisse der Null-Architekturoptimierung

Eindeutig zu erkennen ist, dass jedes Fahrzeugsegment unterschiedliche Werte für die architekturrelevanten Größen m_F, c_A und d_A bevorzugt, um die gesetzten Ziele zu erreichen. Für die folgenden Untersuchungen ist also zu erwarten, dass die Vereinheitlichung dieser Größen zu einer Verschlechterung der Fahrzeuge führt. Deshalb werden die optimalen Ergebnisse der separaten Null-Architekturoptimierungen $f^{(s,0)*}$ nach (4.7) zu einem utopischen Architekturkriterium $f^{(A)**}$ kombiniert, das in den weiteren Analysen als Referenz dient.

4.1.2 Direkte Lösung mittels einstufiger Strategie

Als nächstes wird die in Abschnitt 1.2 beschriebene einstufige Strategie angewendet, um die Fahrzeugmodelle mit vereinheitlichten Architekturvariablen zu optimieren. Die einstufige Strategie besteht, wie in Abb. 4.2 dargestellt, aus einem einzigen als Architekturoptimierung bezeichneten Optimierungskreislauf. Die Besonderheit ist, dass alle Architektur- und Segmentvariablen zu einem Entwurfsvektor $\boldsymbol{p} = \left[\boldsymbol{p}^{(A),T} \quad \boldsymbol{p}^{(1),T} \quad \dots \quad \boldsymbol{p}^{(S),T}\right]^T$ zusammengefasst werden, der abhängig von der Anzahl der Variablen $h^{(A)}$ und $h^{(s)}$, die Dimension $h = h^{(A)} + \sum_{s=1}^{S} h^{(s)}$ hat. Mit den aus Tabelle 4.1 kombinierten unteren und oberen Grenzen $\boldsymbol{p}_u$ und $\boldsymbol{p}_o$ kann das Optimierungsproblem allgemein als

$$\min_{\boldsymbol{p}\in P} f^{(A)}(\boldsymbol{p}) \;\text{ mit }\; P = \left\{\boldsymbol{p} \in \mathbb{R}^h \middle| \boldsymbol{p}_u \leq \boldsymbol{p} \leq \boldsymbol{p}_o\right\} \tag{4.10}$$

formuliert werden.

Um einen optimalen Entwurf zu finden, muss zur Entwurfsbewertung jedes Segment mit den enthaltenen Derivaten $d = 1 \dots D$ einzeln analysiert werden. Die erhaltenen Gütekriterienwerte (4.5) werden dann zu den repräsentativen Segmentkriterien $f^{(s)}$ zusammengefasst. Da für das Testproblem $d = 0$ und $f^{(s)} = f^{(s,0)}$ gilt, kann (4.6) direkt berechnet und mit (4.7) abschließend das Architekturkriterium $f^{(A)}$ gebildet werden, das vom Architekturoptimierer minimiert wird.

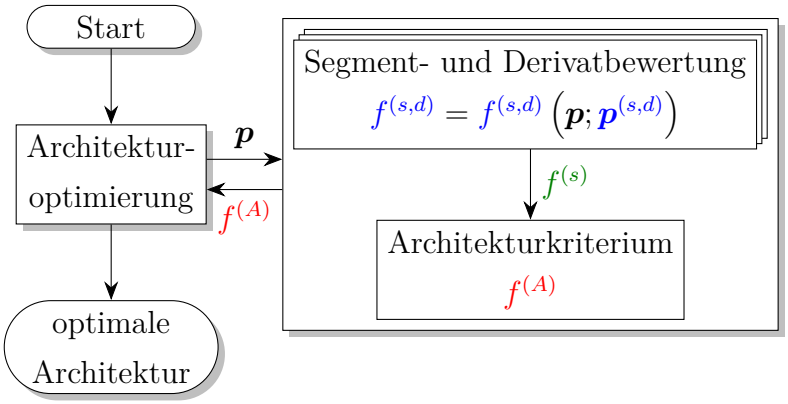

Abbildung 4.2: Ablauf der einstufigen Architekturoptimierung

In Bezug auf das Testproblem werden die Architekturvariablen in (4.1) sowie die Segmentvariablen der jeweiligen Segmente aus (4.2) verwendet. Damit ergibt

sich der alleinige Entwurfsvektor

$$\boldsymbol{p} = \begin{bmatrix} m_F & c_A & d_A & c_F^{(1)} & d_F^{(1)} & c_F^{(2)} & d_F^{(2)} & c_F^{(3)} & d_F^{(3)} & c_{F2}^{(3)} & d_{F2}^{(3)} \end{bmatrix}^T . \tag{4.11}$$

Mit den Kriterien (4.6) und (4.7) und der Dimension $h = 11$ wird das Problem (4.10) mit dem gleichen Optimierungsalgorithmus und den gleichen Einstellungen wie bei der Null-Architekturoptimierung inklusive *Null-Derivaten* $d = 0$ gelöst.

Der Algorithmus konvergiert nach insgesamt 13 200 Funktionsauswertungen mit den normierten Ergebnissen in Abb. 4.3. Der größte Unterschied zwischen Abb. 4.1 und Abb. 4.3 ist, dass die Architkturvariablen nun vereinheitlicht wurden, weshalb nur noch ein einziger Balken für die jeweiligen Parameter vorhanden ist. Die Segmentvariablen weisen deutlich andere Werte als in Abb. 4.1 auf, da jedes Segment versucht, die Verluste der Vereinheitlichung auf seine Weise bestmöglich zu kompensieren. Verglichen mit der utopischen Lösung $f^{(A)**}$ verschlechtern sich die durch das Architekturkriterium $f^{(A)}$ bewerteten Fahrzeuge um etwa 7,92 %. Dieser Verlust ist aber unter Umständen nicht nur auf die Vereinheitlichung zurückzuführen, weswegen sich die Frage stellt, inwiefern mithilfe anderer Optimierungsstrategien bessere Lösungen erreicht werden können. Deshalb wird im Folgenden eine kaskadierte Strategie untersucht.

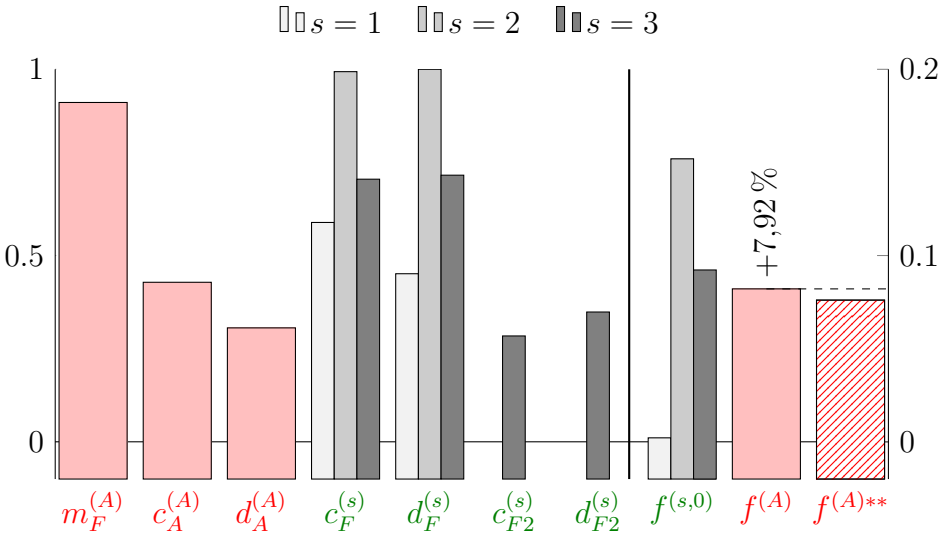

Abbildung 4.3: Optimale Ergebnisse der einstufigen Architekturoptimierung

4.1.3 Direkte Lösung mittels kaskadierter Strategie

Neben der bereits untersuchten einstufigen Strategie gibt es die in Abschnitt 1.2 diskutierte zweistufige Strategie. Trotz geringerer Dimension schneiden sequentiell durchgeführte zweistufige Strategien laut Simpson [77] generell schlechter ab, als die oben betrachtete einstufige Strategie. Im Folgenden wird deshalb das in Abschnitt 3.3 vorgestellte Konzept einer kaskadierten Strategie aus Abb. 3.7a untersucht. Hierbei gibt es ebenfalls zwei Stufen, diese werden aber nicht einmalig sequentiell, sondern ineinander verschachtelt durchgeführt. In Abb. 4.4 ist dieses Konzept als konkreter Prozess dargestellt, der, wie in Abschnitt 3.3 beschrieben, aus einem äußeren Optimierungskreislauf besteht, der als Architekturoptimierer fungiert, und mehreren inneren, die die jeweiligen Segmente optimieren.

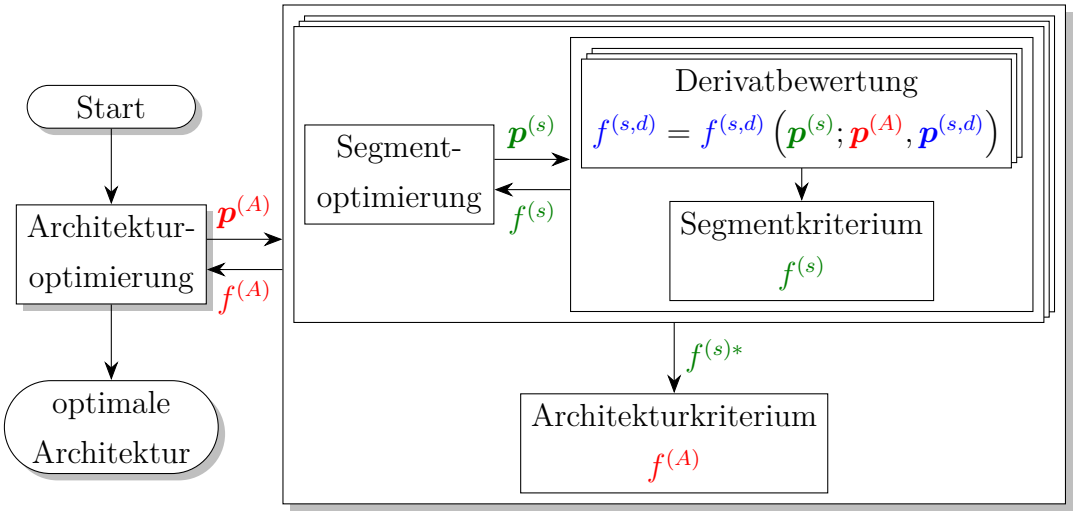

Abbildung 4.4: Ablauf einer kaskadierten Architekturoptimierung

In den beiden Kreisläufen werden die Optimierungsprobleme (3.1) und (3.2) gelöst, wobei der Architekturoptimierer das gleiche Gütekriterium (4.7) wie in der einstufigen Strategie (4.10) minimiert, allerdings nur die $h^{(A)}$ Entwurfsvariablen des Architekturentwurfsvektors (4.1) in den Grenzen $\boldsymbol{p}_u^{(A)}$ und $\boldsymbol{p}_o^{(A)}$ betrachtet. Die optimalen $h^{(s)}$ Segmentvariablen (4.2) werden dann in den inneren Kreisläufen durch das Optimieren der segmentspezifischen Kriterien (4.6) für den aktuellen Architekturentwurf $\boldsymbol{p}^{(A)}$ und die konstanten Parameter der Derivate (4.3) bestimmt. Wie bei der bereits untersuchten einstufigen Strategie werden

für jeden Segmententwurf alle Derivate D des Segments berechnet und zu dem kombinierten Kriterium $f^{(s)}$ zusammengefasst. Da es sich bei den inneren Kreisläufen nun um Optimierungen handelt, werden optimale Gütekriterienwerte $f^{(s)*}$ der Segmente zu $f^{(A)}$ verrechnet und dem äußeren Kreislauf zurückgeführt.

Für die nachfolgende Analyse wird wieder nur ein Auslegungsderivat für jedes Segment, also $d = 0$, betrachtet. Wie in den vorangegangenen Analysen kann das Segmentkriterium dann direkt durch $f^{(s)} = f^{(s,0)}$ ersetzt werden. Für sämtliche Optimierungen wird derselbe Algorithmus mit den gleichen Einstellungen wie in Abschnitt 4.1.1 und 4.1.2 verwendet. Die Architekturoptimierung konvergiert nach insgesamt $3{,}34 \cdot 10^7$ Funktionsauswertungen und resultiert in den in Abb. 4.5 dargestellten Ergebnissen.

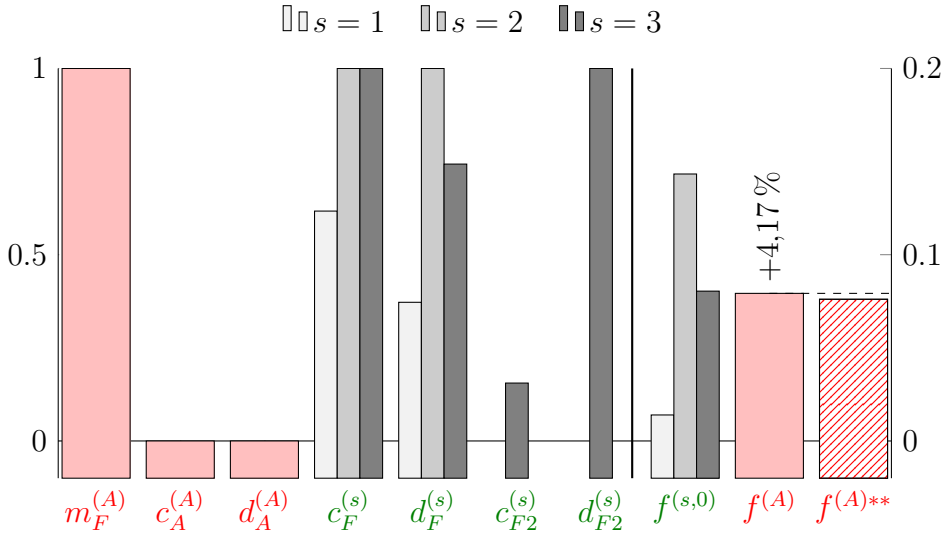

Abbildung 4.5: Optimale Ergebnisse der kaskadierten Architekturoptimierung

Verglichen mit der utopischen Lösung $f^{(A)**}$ schneidet die mit der kaskadierten Strategie optimierte Architektur um 4,17 % schlechter ab, ist aber deutlich besser als das Ergebnis der einstufigen Strategie in Abb. 4.3. Auffällig ist, dass sich nicht nur die Werte der Architekturvariablen, sondern auch die der Segmentvariablen im Vergleich zu den Werten der optimalen Lösung der einstufigen Strategie unterscheiden. Wie vermutet, hängt der Leistungsverlust der einzelnen Fahrzeuge also nicht nur von der Vereinheitlichung, sondern auch von der eingesetzten Architekturoptimierungsstrategie ab.

4.1.4 Zusammenfassung und Diskussion der bisherigen Ergebnisse

In diesem Kapitel wurden verschiedene Strategien zur Architekturoptimierung anhand eines anschaulichen Architekturoptimierungsproblems analysiert. Wie erwartet, wirkt sich die Vereinheitlichung negativ auf die Leistungsfähigkeit der einzelnen Fahrzeuge aus. Entsprechend schneiden alle Strategien im Vergleich zur utopischen Lösung der Null-Architekturoptimierung ohne vereinheitlichte Architekturvariablen schlechter ab. Zusätzlich haben die vorangegangenen Analysen aber auch gezeigt, dass die Wahl der Optimierungsstrategie einen maßgeblichen Einfluss auf das erreichte Ergebnis hat. Die wichtigsten Größen der Ergebnisse aller Strategien sind daher für einen direkten Vergleich in Tabelle 4.2 zusammengefasst.

Strategie	# direkte Funktionsauswertungen	endgültiger $f^{(A)}$ Wert	Verlust zur utopischen Lösung
einstufig	13 200	0,0821	+7,92 %
kaskadiert	$33{,}4 \cdot 10^7$	0,0792	+4,17 %

Tabelle 4.2: Vergleich der Ergebnisse der einkriteriellen Architekturoptimierungen

Die einstufige Strategie weist die größte Abweichung zur utopischen Lösung auf. Diese resultiert aus dem ineffizienteren Konvergenzverhalten der Optimierung aufgrund der höchsten Dimension des Entwurfsraums. Für die kaskadierte Strategie sprechen der geringere endgültige Wert von $f^{(A)}$ sowie der daraus resultierende geringere Verlust zur utopischen Lösung $f^{(A)**}$. Dem steht der Nachteil gegenüber, deutlich mehr Funktionsauswertungen zu benötigen. Die kaskadierte Strategie benötigt verglichen mit der einstufigen Strategie etwa 2500 Mal so viele Funktionsauswertungen.

Das Konzept der Architekturoptimierung soll zur Optimierung komplexer, in Abschnitt 2.4 erläuterter MKS-Modelle eingesetzt werden. Die Effizienz des Ansatzes ist also von besonderer Wichtigkeit, da eine Anwendung mit mehr

als 10 000 Funktionsauswertungen in der Fahrzeugentwicklung unter Berücksichtigung der zur Verfügung stehenden Zeit nicht mehr vertretbar ist. Dieser Aspekt legt die Verwendung der effizienteren einstufigen Strategie nahe. Allerdings muss davon ausgegangen werden, dass bei der Verwendung komplexerer Modelle auch eine größere Anzahl von Entwurfsvariablen optimiert wird, wodurch sich die Problematik der höheren Dimension des Entwurfsraums weiter verschärft. Wie anfänglich erwähnt ist außerdem ein weiteres Ziel der Arbeit, eine mehrkriterielle Architekturoptimierung zu ermöglichen. Der Raum des Optimierungsproblems spannt sich dann über sämtliche Entwurfsvariablen von Architektur und Segmenten sowie deren Kriterien auf.

Es stellt sich also die Frage, ob sich die Vorteile der geringeren Dimension der Segmentoptimierungen der kaskadierten Strategie halten lassen können, während die Gesamtzahl der zeitaufwändigen Funktionsauswertungen reduziert wird. Im folgenden Abschnitt werden deshalb ein mehrkriterieller Ansatz und verschiedene, effizienzsteigernde Maßnahmen auf Basis der kaskadierten Strategie analysiert. Für weitere, auf einkriterielle Probleme bezogene Analysen wird auf den Beitrag von Ubben et al. [89] verwiesen. Die einstufige Strategie wird hingegen aufgrund der genannten Punkte verworfen und im weiteren Verlauf der Arbeit nicht weiter betrachtet.

4.2 Formulierung eines mehrkriteriellen Architekturoptimierungsproblems

Technische Systeme lassen sich zumeist nicht durch ein einzelnes Gütekriterium beschreiben, denn ein Entwicklungsingenieur muss typischerweise ein System entwickeln, das im Allgemeinen mehreren Zielen genügt. Diese Ziele sind in der Regel gegensätzlich, so dass die skalare Optimierung den typischen menschlichen Entscheidungsprozess unterbindet. Durch die bisherige Skalarisierung des Mehrkriterienproblems in Abschnitt 4.1 wird der Lösungsraum a priori eingeschränkt, da im Allgemeinen nur ein einziger optimaler Kompromiss berechnet wird. Das Vektorkriterium $\boldsymbol{f}$ einer mehrkriteriellen Optimierung erlaubt dagegen die unbeschränkte Suche nach optimalen Kompromissen entlang der nach Vilfredo Pareto [57] benannten Pareto-Front. Der Anwender kann dann

einen optimalen Kompromiss anhand seiner Vorlieben oder Vorgaben auswählen [7].

In den folgenden Analysen werden die Segmentoptimierungen mehrkriteriell betrachtet. Die Idee ist, dass eine gemeinsame Architektur ausgelegt wird, die endgültigen Ausprägungen der einzelnen Segmente aber anhand der gefundenen Kompromisslösungen frei gewählt werden können. Die Entscheidung, ob eine maximale Diversität oder ein möglichst einheitliches Verhalten für die Fahrzeuge gewählt wird, kann so auch nach vollendeter Optimierung noch getroffen werden. Auf diese Weise ist es möglich, kurzfristig neue Segmente von der Architektur abzuleiten, die bisher nicht im Fokus der Entwicklung standen.

Die Entwurfspunkte auf einer Pareto-Front werden nach Bestle [6] durch die Edgeworth-Pareto-Optimalität definiert. Edgeworth-Pareto-optimal wird ein Entwurf $\boldsymbol{p}_P \in P$ bezeichnet, wenn es keinen zulässigen anderen Entwurf $\boldsymbol{p} \in P$ gibt, dessen Vektorkriterium $\boldsymbol{f}(\boldsymbol{p}) < \boldsymbol{f}(\boldsymbol{p}_P)$, d.h. $f_i(\boldsymbol{p}) \leq f_i(\boldsymbol{p}_P)\, \forall i \wedge \boldsymbol{f}(\boldsymbol{p}) \neq \boldsymbol{f}(\boldsymbol{p}_P)$, erfüllt. Das bedeutet, dass ein Entwurf der Pareto-Front in mindestens einem Kriterium besser, aber auch in mindestens einem schlechter ist als ein anderer Pareto-optimaler Entwurf oder diese gleich gut sind. In diesem Zusammenhang wird auch von dominierten und nicht dominierten Entwürfen gesprochen. Alle nicht dominierten Entwürfe bilden die Menge Pareto-optimaler Entwürfe. Die Menge dieser Entwürfe wird im Entwurfsraum durch

$$\mathcal{N} := \left\{ \boldsymbol{p}_P \in P \middle| \nexists \boldsymbol{p} \in P : \boldsymbol{f}(\boldsymbol{p}) < \boldsymbol{f}(\boldsymbol{p}_P) \right\} \rightarrow F_P = \boldsymbol{f}(\mathcal{N}) \tag{4.12}$$

beschrieben, wobei F_P die Menge der Pareto-optimalen Entwürfe im Kriterienraum ist.

Eine Mehrzieloptimierung liefert z.B. K nicht dominierte Entwürfe $\boldsymbol{p}_{P,k}$ sowie $\bar{K}$ dominierte Entwürfe $\boldsymbol{p}_k$:

$$\mathcal{N} = \{\boldsymbol{p}_{P,k}, k = 1 \ldots K\}, \quad \mathcal{D} = \left\{\boldsymbol{p}_k, k = 1 \ldots \bar{K}\right\}. \tag{4.13}$$

Das Ziel einer Optimierung ist nun, eine möglichst große Menge $\mathcal{N}$ zu finden, wobei die enthaltenen Entwürfe weit voneinander entfernt liegen sollten, um den größtmöglichen Bereich der tatsächlichen Pareto-Front abzudecken. Üblicherweise ist es insbesondere bei mehrdimensionalen Kriterienräumen äußert

aufwändig, die tatsächliche Pareto-Front zu bestimmen. Es muss deshalb immer von einer Approximation der Front gesprochen werden. Dabei handelt es sich stets um einen Kompromiss zwischen dem Grad der Annäherung an die tatsächliche Front und der dafür aufgebrachten Anzahl von Funktionsauswertungen.

Das bisher betrachtete skalare Segmentoptimierungsproblem (3.2) wird nun durch das Mehrzielproblem

$$\min_{\boldsymbol{p}^{(s)} \in P^{(s)}} \boldsymbol{f}^{(s,0)} \left(\boldsymbol{p}^{(s)}; \boldsymbol{p}^{(A)}, \boldsymbol{p}^{(s,0)} \right) \text{ mit } P^{(s)} = \left\{ \boldsymbol{p}^{(s)} \in \mathbb{R}^{h^{(s)}} \middle| \boldsymbol{p}_u^{(s)} \leq \boldsymbol{p}^{(s)} \leq \boldsymbol{p}_o^{(s)} \right\} \tag{4.14}$$

ersetzt. Die bisher betrachteten Entwurfsvariablen (4.1) und (4.2) und deren Entwurfsräume (4.4) bleiben unverändert. Die einzelnen Segmentkriterien (4.6) werden nun durch ein entsprechendes Vektorkriterium

$$\boldsymbol{f}^{(s,0)} = \begin{bmatrix} f_1^2 & f_2^2 & f_3^2 \end{bmatrix}^T \tag{4.15}$$

ersetzt. Um Entwürfe, die weit vom Zielbereich entfernt liegen, stärker zu bestrafen, werden die Kriterien (4.5) jeweils quadriert betrachtet.

Da die Segmente nun mehrkriteriell optimiert werden, kann das kombinierte Architekturkriterium nicht nach (4.7) gebildet werden. Es bedarf eines geeigneten Konzeptes, um die sich ergebenden Pareto-Fronten der Segmentoptimierungen mit einem skalaren Wert zu beschreiben, womit sich der nächste Abschnitt befasst.

4.2.1 Konzept zur Skalarisierung von Pareto-Fronten

In der Literatur werden sogenannte Metriken für die Beschreibung oder Bewertung von Pareto-Fronten eingesetzt. Typischerweise dienen sie dem Vergleich verschiedener multikriterieller Optimierungsalgorithmen, indem die gefundenen Pareto-Fronten durch Metriken bewertet werden und damit die Leistungsfähigkeit des Algorithmus bemessen wird.

In der Forschung haben sich verschiedene Metriken etabliert, wobei die von Zitzler und Thiele eingeführte S-Metrik [100] zu den bekanntesten gehört und als besonders gerechtes Maß gilt [98, 20]. Die S-Metrik wird aber nicht nur für

den Vergleich von bereits optimierten Pareto-Fronten eingesetzt, Emmerich et al. [25] setzen die S-Metrik beispielsweise als Selektionskriterium für Entwürfe in einem Optimierungsalgorithmus ein, um Pareto-optimale Lösungen zu finden. Aufgrund der weiten Verbreitung und anerkannten Qualität der S-Metrik wird diese hier zur Bewertung der Pareto-Fronten der Segmentoptimierungen herangezogen.

Die S-Metrik ist ein Maß für das dominierte Hypervolumen. Ähnlich wie ein Pareto-optimaler Entwurf einen nicht Pareto-optimalen dominieren kann, beschreibt die S-Metrik das Hypervolumen, das von einer Pareto-Front dominiert wird. In Abb. 4.6 sind zwei zweidimensionale, aus den Mengen $F_{P,1}^{(s)}$ und $F_{P,2}^{(s)}$ bestehende Pareto-Fronten eines Segments s und die zugehörigen dominierten Flächen bzw. S-Metriken $\mathcal{S}\left(F_{P,1}^{(s)}\right)$ und $\mathcal{S}\left(F_{P,2}^{(s)}\right)$ in Bezug auf einen Referenzpunkt $\boldsymbol{f}_{ref}$ dargestellt. Je größer diese dominierte Fläche bzw. ein dominiertes Hypervolumen ist, umso näher befindet sich die Pareto-Front am absoluten Optimum aller Kriterien und bzw. oder umso stärker ist die Front besetzt.

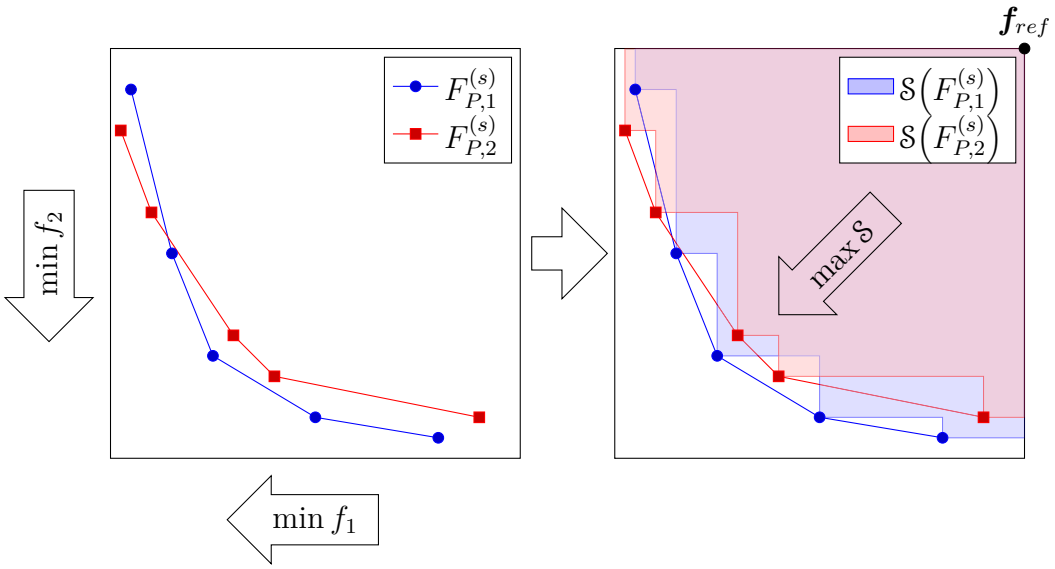

Abbildung 4.6: Bewertung von Pareto-Fronten bei mehrkriterieller Architekturoptimierung

Nach Coello Coello et al. [20] kann das Hypervolumen $\mathcal{S}$ mathematisch mit dem Lebesgue-Maß Λ der Vereinigung der Hyperwürfel a_k, die durch die nicht

dominierten Punkte $\boldsymbol{f}_{P,k}$ und einem Referenzpunkt $\boldsymbol{f}_{ref}$ aufgespannt sind, berechnet werden:

$$\mathcal{S}\left(F_P\right) := \Lambda\Big(\Big\{\bigcup_k a_k \Big| \boldsymbol{f}_{P,k} \in F_P\Big\}\Big) = \Lambda\Big(\Big\{\bigcup_{\boldsymbol{f}_P \in F_P} \boldsymbol{f} \Big| \boldsymbol{f}_P \prec \boldsymbol{f} \prec \boldsymbol{f}_{ref}\Big\}\Big). \tag{4.16}$$

Die Berechnung des Hypervolumens ist sehr zeitintensiv. Die effiziente Berechnung des Volumens hat deshalb zu einem großen Forschungsinteresse geführt, das in der Entwicklung verschiedener Algorithmen resultierte. In dieser Arbeit wird ein sogenannter *Dimension-Sweep* Algorithmus von Fonseca et al. [30] verwendet, dessen grundlegender Ablauf in Abschnitt A dargestellt ist.

Für die Bildung des Architekturkriteriums $f^{(A)}$ werden die Pareto-optimalen Entwürfe $\boldsymbol{p}_P^{(s)}$ der Segmentoptimierungen des Vektorkriteriums (4.15) benötigt:

$$\boldsymbol{p}_P^{(s)} = \underset{\boldsymbol{p}^{(s)} \in P^{(s)}}{\arg\min}\, \boldsymbol{f}^{(s,0)}\left(\boldsymbol{p}^{(s)}; \boldsymbol{p}^{(A)}, \boldsymbol{p}^{(s,0)}\right). \tag{4.17}$$

Für jeden Architekturentwurf $\boldsymbol{p}^{(A)}$ ergibt sich eine Anzahl $K^{(s)}$ Pareto-optimaler Segmententwürfe $\boldsymbol{p}_P^{(s)}$, die zur segmentabhängigen Menge Pareto-optimaler Entwürfe

$$\mathcal{N}^{(s)} := \left\{ \boldsymbol{q}_{P,k}^{(s)} = \begin{bmatrix} \boldsymbol{p}^{(A)} \\ \boldsymbol{p}_{P,k}^{(s)} \end{bmatrix}, k = 1 \ldots K^{(s)} \right\} \tag{4.18}$$

zusammengefasst werden. Um die S-Metrik zu berechnen, werden die Entwürfe jedes Segments im Kriterienraum benötigt. Mit den zugehörigen Derivatvariablen $\boldsymbol{p}^{(s,0)}$ kann mit (4.15) die Menge der Pareto-optimalen Entwürfe des aktuellen Architekturentwurfs im Kriterienraum berechnet werden:

$$F_P^{(s)} := \boldsymbol{f}^{(s,0)}\left(\mathcal{N}^{(s)}; \boldsymbol{p}^{(s,0)}\right). \tag{4.19}$$

Die S-Metriken $\mathcal{S}^{(s)}$ der Segmente eines jeden Architekturentwurfes werden mit den Pareto-Mengen $F_P^{(s)}$ und dem Referenzpunkt $\boldsymbol{f}_{ref} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$ nach (4.16) gebildet:

$$\mathcal{S}^{(s)} = \mathcal{S}\left(F_P^{(s)}\right). \tag{4.20}$$

Das Architekturkriterium für Mehrzielprobleme $f_{\mathcal{S}}^{(A)}$ wird schließlich wie bei der einkriteriellen Optimierung analog zu (4.7) als Mittelwert der Hypervolumina der Segmente $\mathcal{S}^{(s)}$ gebildet:

$$f_{\mathcal{S}}^{(A)} = \frac{1}{S} \sum_{s=1}^{S} \mathcal{S}^{(s)}. \tag{4.21}$$

Abschließend ergeben sich die im Folgenden zu analysierenden Optimierungsprobleme der Segmente nach (4.14) und das der Architektur als

$$\max_{\boldsymbol{p}^{(A)} \in P^{(A)}} f_{\mathrm{S}}^{(A)} \left(\boldsymbol{p}^{(A)} \right) \text{ mit } P^{(A)} = \left\{ \boldsymbol{p}^{(A)} \in \mathbb{R}^{h^{(A)}} \middle| \boldsymbol{p}_u^{(A)} \leq \boldsymbol{p}^{(A)} \leq \boldsymbol{p}_o^{(A)} \right\}. \quad (4.22)$$

4.2.2 Direkte Lösung mithilfe der S-Metrik

Der Fokus der folgenden Kapitel liegt auf der Einführung effizienzsteigernder Maßnahmen für die Mehrzieloptimierung von Architekturen. Für Vergleichszwecke wird deshalb zunächst eine direkte Optimierung ohne zusätzliche Maßnahmen durchgeführt. Die Strategie gleicht der in Abschnitt 4.1.3 analysierten, wobei nun die Probleme (4.14) und (4.22) gelöst werden.

Für die inneren und äußeren Optimierungskreisläufe wird wieder der SPEA2 Algorithmus verwendet. Für die Architektur werden maximal 90 Generationen mit jeweils 100 neuen Individuen bei einer Startpopulation von 1000 Individuen generiert. Nach spätestens 10 Generationen ohne Verbesserung wird der Algorithmus als konvergiert angenommen. Die Segmentoptimierungen laufen ähnlich ab, der einzige Unterschied ist ein Archiv mit einer Kapazität von 20 Individuen. Individuen in diesem Archiv repräsentieren die besten Entwürfe, die bisher gefunden wurden, und werden stets in die nächste Generation übernommen. Üblicherweise handelt es sich hierbei um die oder einen Teil der Pareto-optimalen Entwürfe. Ein Entwurf wird erst aus dem Archiv entfernt, wenn ein Besserer gefunden wurde.

Für die direkte mehrkriterielle kaskadierte Architekturoptimierung werden insgesamt $5{,}34 \cdot 10^6$ Funktionsauswertungen benötigt. Die zugehörigen Ergebnisse sind in Abb. 4.7 dargestellt. Zu sehen sind die Pareto-Fronten der drei Segmente. Das Kriterium f_1 bildet die dritte Dimension und wird durch eine Farbskala repräsentiert. In der unteren rechten Ecke sind die Werte der normierten Architekturvariablen (4.1) sowie die Hypervolumina (4.20) und der Gütefunktionswert (4.21) des besten Architekturentwurfs dargestellt. Bei der Betrachtung der Werte der Architekturvariablen fällt auf, dass dieselben Werte wie bei der einkriteriellen Architekturoptimierung in Abb. 4.5 gefunden werden. Allerdings bietet der mehrkriterielle Ansatz nun die Möglichkeit, die

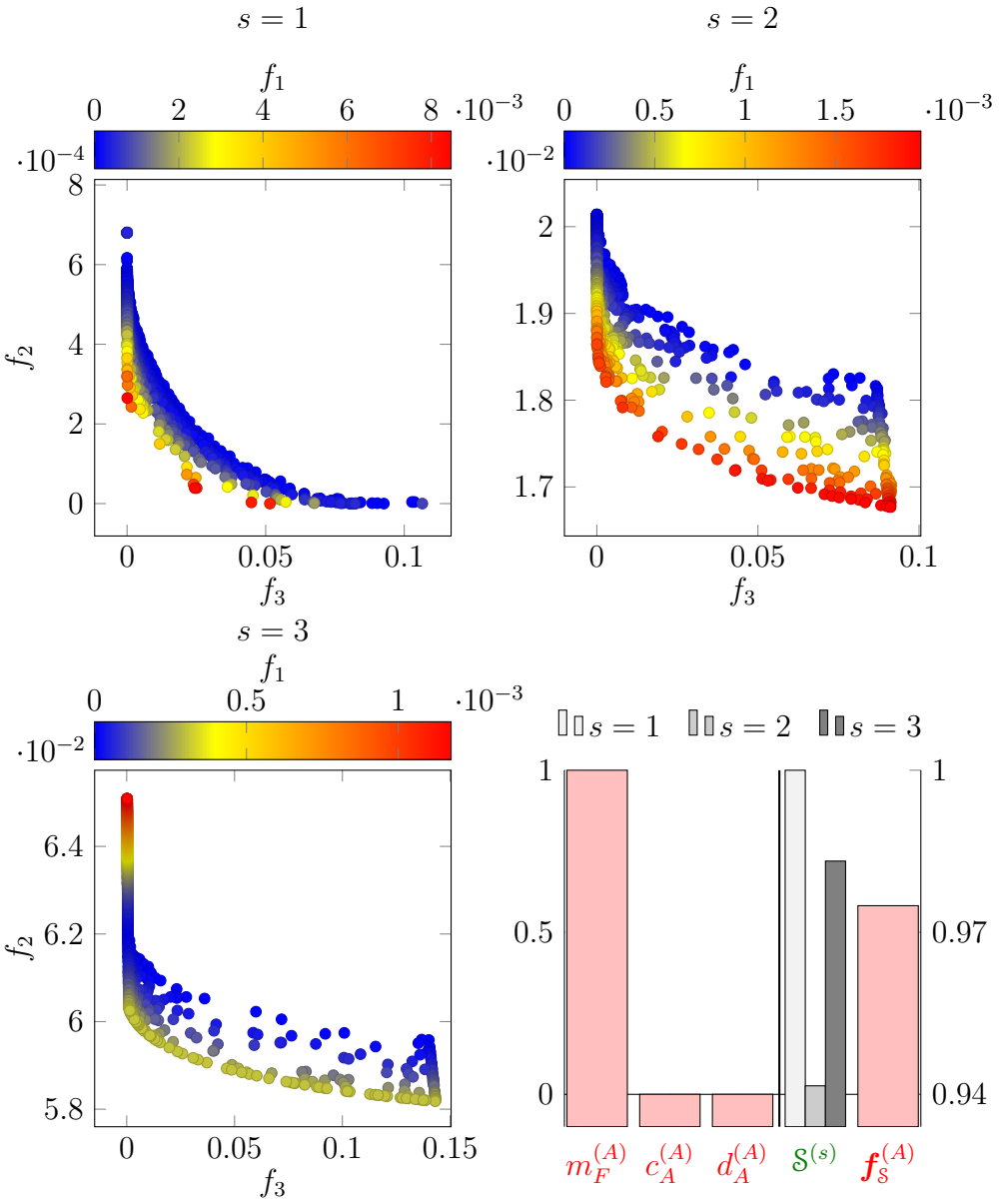

Abbildung 4.7: Optimale Ergebnisse der direkten kaskadierten Strategie

Segmente frei aus den Entwürfen der Pareto-Fronten auszuwählen, ohne die Architektur zu verschlechtern. Allerdings ist auch hier die Zahl der Entwurfsauswertungen zu hoch. Wie bereits in Abschnitt 4.1.4 erwähnt, muss diese Anzahl reduziert werden, um die Architekturoptimierung in der Entwicklung einsetzbar zu machen. Hierfür bieten sich verschiedene Strategien an, die im Folgenden untersucht werden.

4.3 Verbesserte Strategien zur Lösung mehrkriterieller Architekturprobleme

Eine verbreitete Strategie für die Reduzierung des Optimierungsaufwands ist der Einsatz sogenannter Response Surfaces (RS), die auch als Antwortflächen, Surrogates oder Metamodelle bezeichnet werden. Diese werden eingesetzt, um Gütefunktionen mithilfe von analytischen Ersatzfunktionen, die deutlich schneller berechnet werden können, zu approximieren. Die RS werden aus einer Menge von Entwürfen gebildet, die zuvor mit direkten Funktionsauswertungen berechnet wurden. Diese Entwürfe werden als Stützstellen bezeichnet.

Wenn RS eingesetzt werden, sollte sichergestellt sein, dass sie eine ausreichende Approximationsqualität aufweisen, um vertrauenswürdige Optimierungsergebnisse zu erzielen [53]. Dafür bietet sich eine Adaptive Response Surface Modeling (ARSM) Strategie an. Das Ziel dieser Art von Verfahren ist, die Approximationsqualität im Laufe der Optimierung zu verbessern. Park et al. [58] erreichen diese Verbesserung beispielsweise so, dass der optimale Entwurf einer Optimierung auf den Antwortflächen mit dem aufwändigen Modell nachgerechnet und zur Stützstellenmenge hinzugefügt wird. Die Optimierung auf den Antwortflächen und das darauffolgende Hinzufügen des besten Entwurfes zur Stützstellenmenge wird anschließend so oft wiederholt, bis ein vordefiniertes Konvergenzkriterium erfüllt oder eine maximale Anzahl von Funktionsauswertungen erreicht werden. In der vorliegenden Arbeit wird eine auf mehrkriterielle Optimierungen angepasste Strategie verwendet [87]. Der Kern dieser Strategie liegt in der Bestimmung neuer Stützstellen, die nun genauer betrachtet wird.

Sei

$$I_i = \{\boldsymbol{p}_j, j = 1 \dots J_i\} \tag{4.23}$$

eine Stützstellenmenge der aktuellen Iteration i mit den daraus ermittelten Mengen der Pareto-optimalen Entwürfe $\mathcal{N}$ und dominierten Entwürfe $\mathcal{D}$ nach (4.13) eines mehrkriteriellen Optimierungsprozesses. Nun gilt es, die Stützstellenmenge I_i und die darauf basierenden Antwortflächen gezielt in aussichtsreichen Regionen zu verbessern. Hierfür wird der in Abb. 4.8 dargestellte Prozess verwendet.

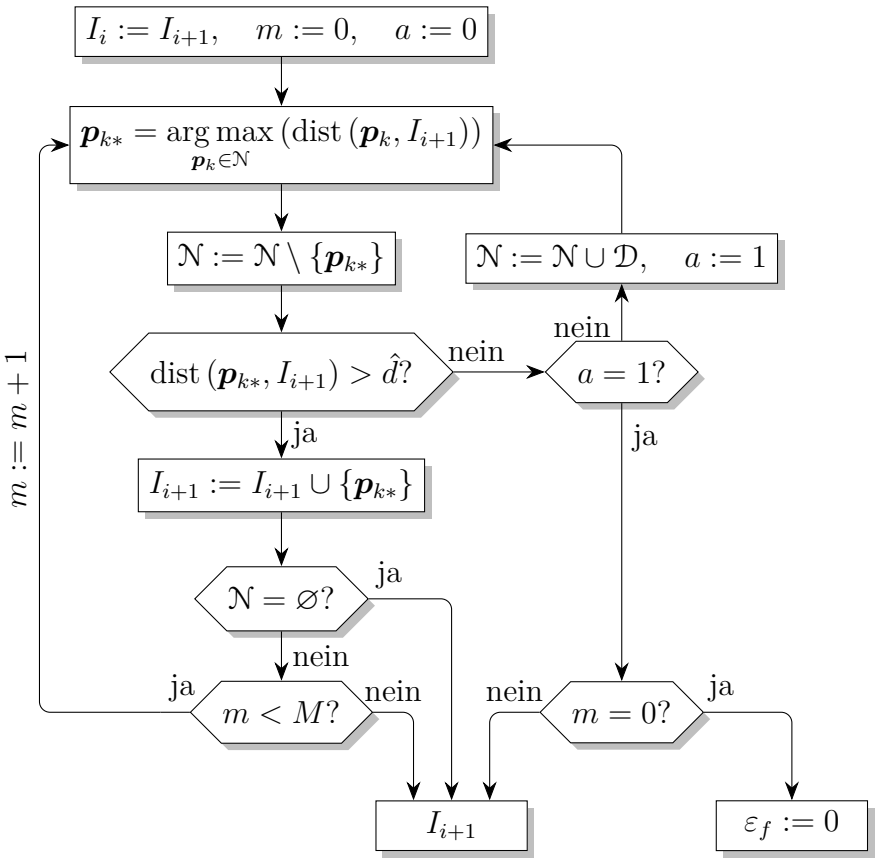

Abbildung 4.8: Bestimmung neuer Stützstellen

Zu Beginn werden die minimalen Abstände zwischen nicht dominierten Entwürfen $\boldsymbol{p}_k \in \mathcal{N}$ und bereits vorhandenen Stützstellen der aktuellen Menge $I_{i+1} := I_i$ berechnet:

$$d\,(\boldsymbol{p}_k) := \mathrm{dist}\,(\boldsymbol{p}_k, I_{i+1}) = \min_{\boldsymbol{p}_j \in I_{i+1}} \|\boldsymbol{p}_k - \boldsymbol{p}_j\|\,. \tag{4.24}$$

Der Entwurf $\boldsymbol{p}_{k*}$ mit dem größten Abstand $d\,(\boldsymbol{p}_{k*})$ wird als neue potenzielle

Stützstelle gewählt und aus der Menge nicht dominierter Entwürfe $\mathcal{N}$ entfernt. Um zu verhindern, dass neue Stützstellen gewählt werden, die zu nah an bereits vorhandenen liegen oder gar identisch sind, wird eine charakteristische Distanz $\hat{d}$ eingeführt. Diese Distanz wird nach

$$\hat{d}\left(J_i\right) := \frac{a_{\hat{d}}}{\sqrt[h-1]{J_i} - 1} \tag{4.25}$$

berechnet, wobei $J_i = |I_i|$ die Anzahl der Stützstellen, h die Dimension des Entwurfsraums und $a_{\hat{d}} > 0$ ein frei wählbarer Faktor zur Beschreibung der Ausdehnung einer Pareto-Front ist.

Die Gleichung (4.25) entsteht aus der Annahme, dass normierte Entwurfsvariablen $p_i \in [0, 1]$ verwendet werden und ein Einheitshyperwürfel im h-dimensionalen Entwurfsraum gleichmäßig mit $J_i = N^h$ Punkten gefüllt ist, wobei $N \geq 2$ ist. In diesem Fall ist der minimale Abstand zwischen benachbarten Punkten $\hat{d} = 1/\left(N - 1\right) = 1/\left(\sqrt[h]{J_i} - 1\right)$. Dieser Zusammenhang ist für $N = 3$ Faktorstufen und $h = 3$ Dimensionen in Abb. 4.9 dargestellt. Auf lange Sicht sammeln sich die Stützstellen der Menge I_i auf der Pareto-Front, wobei angenommen wird, dass diese eine Dimension kleiner als der Entwurfsraum ist. Somit kann h durch $(h - 1)$ substituiert werden, woraus die Gleichung (4.25) resultiert.

Ist die Distanz $d\left(\boldsymbol{p}_{k*}\right)$ einer potentiellen Stützstelle nun größer als $\hat{d}$, dann wird dieser Entwurf $\boldsymbol{p}_{k*}$ zur Menge der Stützstellen I_{i+1} hinzugefügt und der Prozess wiederholt, bis eine vordefinierte Anzahl M neuer Stützstellen gefunden wurde. Für den Fall, dass keine weiteren potenziellen Stützstellen vorhanden sind, also $\mathcal{N} = \varnothing$ ist, werden die m bis dahin gefunden Stützstellen zur RS Verbesserung herangezogen. Wenn die Distanz $d\left(\boldsymbol{p}_{k*}\right) < \hat{d}$ ist, wird die Menge Pareto-optimaler Entwürfe $\mathcal{N}$ um die Menge dominierter Entwürfe $\mathcal{D}$ erweitert, um den Algorithmus zu einer globalen Verbesserung der Antwortflächen zu zwingen. Wurde die Menge $\mathcal{N}$ bereits erweitert, ist also der Schalter $a = 1$, werden ebenfalls die m bis dahin gefunden Stützstellen gewählt. Werden überhaupt keine neuen Stützstellen gefunden, ist also $m = 0$, dann wird eine als Konvergenzkriterium verwendete Fehlerrate ε_f zu Null gesetzt, deren Verwendung später genauer erläutert wird.

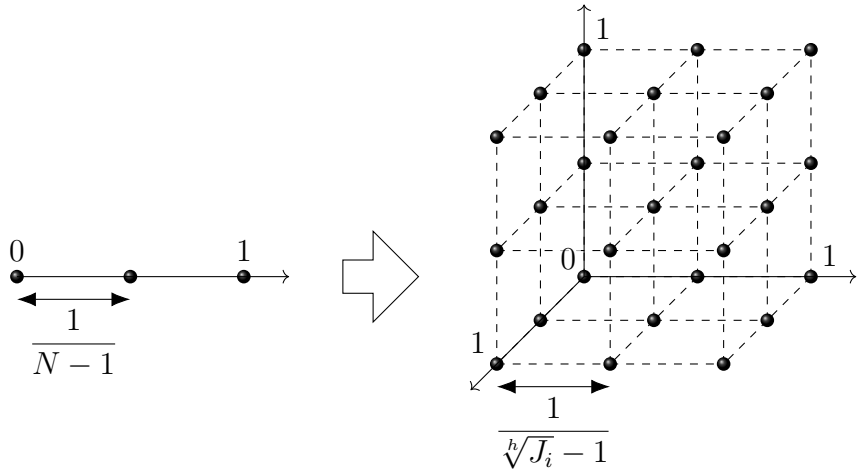

Abbildung 4.9: Beispielhafte Darstellung der charakteristischen Distanz $\hat{d}$ anhand eines vollfaktoriellen Versuchsplans mit 3 Dimensionen und 3 Faktorstufen

Auf Basis dieser Strategie zur Wahl neuer Stützstellen werden nun zwei verbesserte mehrkriterielle Strategien untersucht. Die erste kombiniert die kaskadierte Strategie mit einem adaptiven Antwortflächen-Verfahren (ARSM), die zweite kommt ursprünglich aus dem Feld der multidisziplinären Optimierung und wird als Concurrent Subspace Optimization (CSSO) bezeichnet.

4.3.1 Lösung mit adaptiven Antwortflächen für unterschiedliche Segmente

Der zunächst untersuchte ARSM Prozess ist in Abb. 4.10 dargestellt und wird auf das bereits in Abschnitt 4.2.2 betrachtete mehrkriterielle Architekturproblem angewendet. Bevor auf die Ergebnisse der Strategie eingegangen wird, folgt im vorderen Teil des Abschnitts eine ausführliche Erläuterung ihres Ablaufs.

Für die initiale Iteration $i := 0$ werden zunächst individuelle Mengen $I_0^{(s)}$ initialer Stützstellen generiert, wobei $\boldsymbol{q}^{(s)}$ nach (4.8) über den gesamten Entwurfsraum jedes Segmentes s inklusive der Architekturvariablen verteilt wird:

$$I_0^{(s)} = \left\{ \boldsymbol{q}_j^{(s)} = \begin{bmatrix} \boldsymbol{p}_j^{(A,s)} \\ \boldsymbol{p}_j^{(s)} \end{bmatrix}, j = 1 \dots J_0^{(s)} \right\}. \tag{4.26}$$

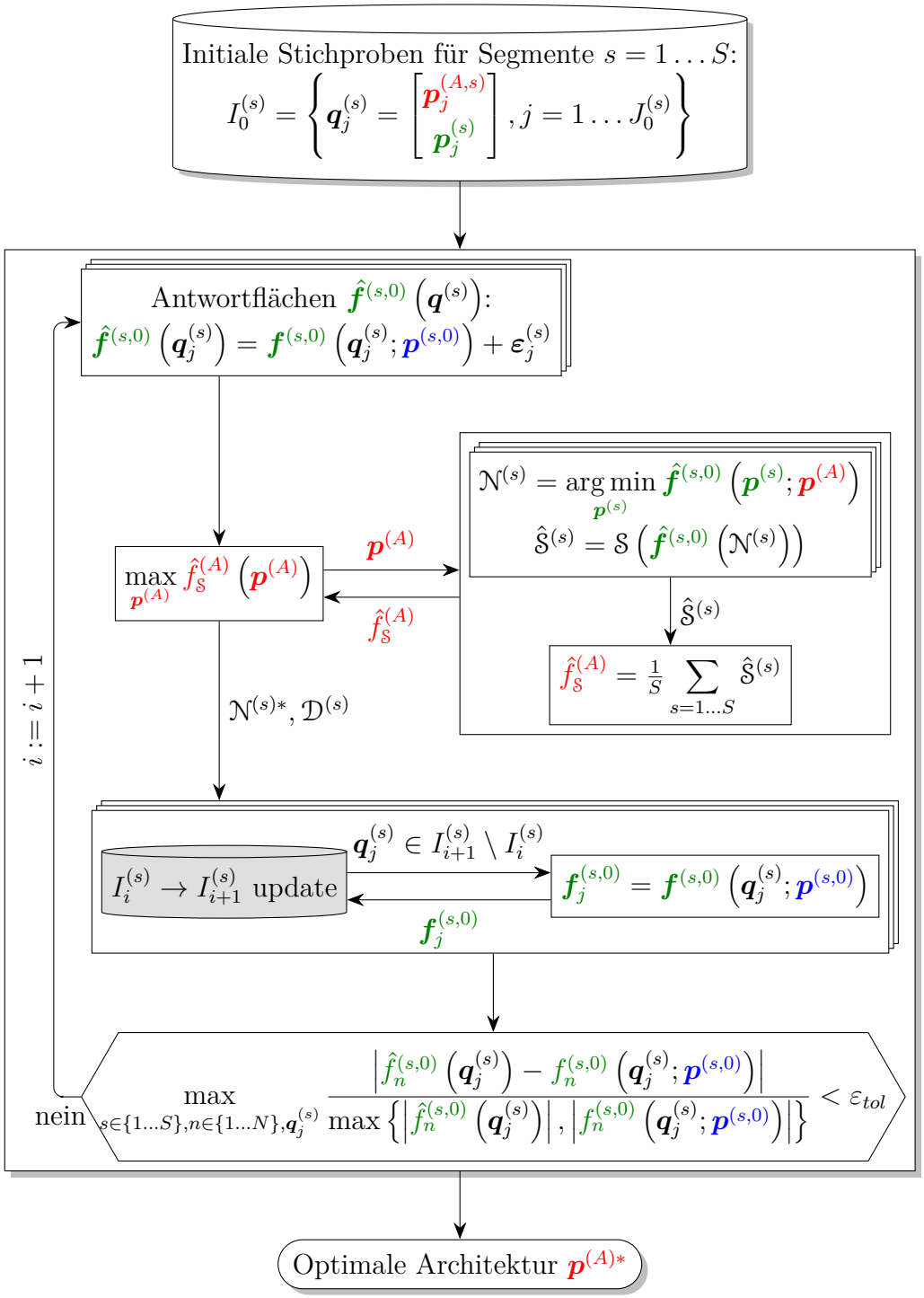

Abbildung 4.10: Ablauf der ARSM-basierten Architekturoptimierung

Um diese Menge und den zugehörigen Entwurfsraum möglichst gleichmäßig mit Stützstellen zu besetzen, bieten sich verschiedene Verfahren aus dem Gebiet der DoE an. Da eine möglichst unkorrelierte Stichprobe ohne Anhäufung gewünscht und die Anzahl der Stützstellen sehr gering ist, wird ein Advanced Latin Hypercube Sampling (ALHS) Verfahren von Huntington et al. [39] eingesetzt. Kurz gesagt minimiert dieses Verfahren die Korrelation der Stichprobe, indem zwei Werte einer Variablen getauscht und der daraus resultierende Grad der Verringerung der Korrelation gemessen wird. Wird keine weitere Verbesserung erreicht oder ist eine vorgegebene maximale Korrelation unterschritten, endet der Algorithmus.

Im nächsten Schritt werden Antwortflächen $\hat{\boldsymbol{f}}^{(s,0)}\left(\boldsymbol{q}^{(s)}\right)$ aus der aktuellen Menge von Stützstellen $\boldsymbol{q}_j^{(s)} \in I_i^{(s)}$ und zugehörigen Funktionswerten $\boldsymbol{f}_j^{(s,0)} = \boldsymbol{f}^{(s,0)}\left(\boldsymbol{q}_j^{(s)}; \boldsymbol{p}^{(s,0)}\right)$ nach (4.15) gebildet:

$$\hat{\boldsymbol{f}}^{(s,0)}\left(\boldsymbol{q}^{(s)}\right) = \boldsymbol{f}^{(s,0)}\left(\boldsymbol{q}^{(s)}; \boldsymbol{p}^{(s,0)}\right) + \boldsymbol{\varepsilon}^{(s)}. \tag{4.27}$$

Für die Approximation der Antwortflächen wird das sogenannte Metamodel of Optimal Prognosis (MOP) verwendet [94]. Kurz gesagt ist das MOP ein automatisierter Ansatz, der das beste Antwortflächenverfahren für eine gegebene Datenmenge in Bezug auf eine spezifische Validierungsmethode findet. In *optiSLang* implementierte Antwortflächenverfahren sind lineare oder quadratische Polynome der kleinsten Fehlerquadrate, Moving Least Squares und das herkömmliche Kriging [24]. Eine detailliertere Vorstellung und Analyse dieser und weiterer Verfahren gibt es von Roos et al. [65] und in gängigen Standardwerken zur Thematik. Als Validierungsmethode wird in dieser Arbeit die k-fold Kreuzvalidierung verwendet [75].

Nach der Generierung der Antwortflächen $\hat{\boldsymbol{f}}^{(s,0)}$ wird eine kaskadierte Optimierung angewendet. Der Unterschied zu den bisherigen Anwendungen ist nun, dass die Optimierungen auf Antwortflächen ablaufen, womit sich das Architekturoptimierungsproblem

$$\max_{\boldsymbol{p}^{(A)} \in P^{(A)}} \hat{f}_{\mathbb{S}}^{(A)}\left(\boldsymbol{p}^{(A)}\right) \text{ mit } P^{(A)} = \left\{\boldsymbol{p}^{(A)} \in \mathbb{R}^{h^{(A)}} \middle| \boldsymbol{p}_u^{(A)} \leq \boldsymbol{p}^{(A)} \leq \boldsymbol{p}_o^{(A)}\right\} \tag{4.28}$$

und die Segmentoptimierungsprobleme

$$\min_{\boldsymbol{p}^{(s)} \in P^{(s)}} \hat{\boldsymbol{f}}^{(s,0)} \left(\boldsymbol{p}^{(s)}; \boldsymbol{p}^{(A)} \right) \text{ mit } P^{(s)} = \left\{ \boldsymbol{p}^{(s)} \in \mathbb{R}^{h^{(s)}} \middle| \boldsymbol{p}_u^{(s)} \leq \boldsymbol{p}^{(s)} \leq \boldsymbol{p}_o^{(s)} \right\} \quad (4.29)$$

ergeben. Ergebnis der Segmentoptimierungen sind die Pareto-optimale Entwürfe enthaltenden Mengen $\mathcal{N}^{(s)}$. Diese werden herangezogen, um die zugehörigen S-Metriken $\hat{\mathcal{S}}^{(s)}$ nach (4.19) und (4.20) mit

$$\hat{\mathcal{S}}^{(s)} = \mathcal{S} \left(\hat{\boldsymbol{f}}^{(s,0)} \left(\mathcal{N}^{(s)} \right) \right) \quad (4.30)$$

zu berechnen, die anschließend nach (4.21) zum Architekturkriterium $\hat{f}_{\mathcal{S}}^{(A)}$ zusammengefasst und nach (4.28) optimiert werden. Nach Beendigung der Architekturoptimierung ergeben sich die Mengen Pareto-optimaler Entwürfe für den besten Architekturentwurf $\boldsymbol{p}^{(A)*}$:

$$\mathcal{N}^{(s)*} := \left\{ \boldsymbol{q}_{P,k}^{(s)*} = \begin{bmatrix} \boldsymbol{p}^{(A)*} \\ \boldsymbol{p}_{P,k}^{(s)} \end{bmatrix}, k = 1 \ldots K^{(s)*} \right\}. \quad (4.31)$$

Für alle anderen $K^{(A)}$ berechneten, aber nicht optimalen Architekturentwürfe $\boldsymbol{p}^{(A)}$ werden aus den Pareto-Punkten der Segmente Punkte $\boldsymbol{q}_P^{(s)} \in \mathcal{D}^{(s)}$ mit

$$\mathcal{D}^{(s)} := \bigcup_{k=1}^{K^{(A)}} \mathcal{N}_k^{(s)} \quad (4.32)$$

gebildet. Die Menge $\mathcal{D}^{(s)}$ enthält hierbei nicht zwangsläufig nur dominierte Entwürfe, da eine Pareto-Front mit einem kleinen Hypervolumen Entwürfe enthalten kann, die Entwürfe einer anderen Pareto-Front mit größerem Hypervolumen dominieren. Da die Pareto-Front des Architekturentwurfs mit dem größten mittleren Hypervolumen aber im Fokus des Interesses steht, werden die Entwürfe der Pareto-Fronten mit kleinerem Hypervolumen zu den Mengen $\mathcal{D}^{(s)}$ hinzugefügt. In diesem Zusammenhang wird im Weiteren von Metrik-dominierten Pareto-Fronten und Entwürfen gesprochen.

Mit den Mengen $\mathcal{N}^{(s)*}$ und $\mathcal{D}^{(s)}$ wird dann der Prozess in Abb. 4.8 für jedes Segment s separat bedient, um neue Stützstellen $\boldsymbol{q}^{(s)}$ zu bestimmen. Dieser Prozess ist vereinfacht als grau hinterlegter Block in Abb. 4.10 dargestellt. Die neu gefundenen Stützstellen $\boldsymbol{q}_j^{(s)} \in I_{i+1}^{(s)} \setminus I_i^{(s)}$ werden anschließend mit $\boldsymbol{f}_j^{(s,0)} = \boldsymbol{f}^{(s,0)} \left(\boldsymbol{q}_j^{(s)}; \boldsymbol{p}^{(s,0)} \right)$ direkt berechnet, um die zugehörigen Funktionswerte

zu erhalten.

Zur Prüfung der Konvergenz werden die Approximationsgüten der neuen Stützstellen im Kriterienraum bewertet. Dafür wird die maximale relative Differenz zwischen den von Antwortflächen erhaltenen Funktionswerten $\hat{\boldsymbol{f}}_j^{(s,0)}$ und den original nachgerechneten $\boldsymbol{f}_j^{(s,0)}$ bestimmt:

$$\max_{s\in\{1\ldots S\},n\in\{1\ldots N\},\boldsymbol{q}_j^{(s)}} \frac{\left|\hat{f}_n^{(s,0)}\left(\boldsymbol{q}_j^{(s)}\right) - f_n^{(s,0)}\left(\boldsymbol{q}_j^{(s)};\boldsymbol{p}^{(s,0)}\right)\right|}{\max\left\{\left|\hat{f}_n^{(s,0)}\left(\boldsymbol{q}_j^{(s)}\right)\right|,\left|f_n^{(s,0)}\left(\boldsymbol{q}_j^{(s)};\boldsymbol{p}^{(s,0)}\right)\right|\right\}} < \varepsilon_{tol}. \tag{4.33}$$

Der Algorithmus wird als konvergiert angenommen, wenn die Fehlertoleranz ε_{tol} unterschritten wird. Aus Sicherheitsgründen endet der Algorithmus erst nach einer vordefinierte Anzahl von zusätzlichen Iterationen, schließlich könnten weit von der echten Pareto-Front entfernte Stützstellen in einem Bereich höchster Approximationsgüte liegen und der Prozess somit fälschlicherweise enden. Erst wiederholt gute Approximationsgüten sprechen für vertrauenswürdige Ergebnisse. Für den Fall, dass keine neuen Stützstellen gefunden werden, wird die Fehlerrate ε_f aus Abb. 4.8 herangezogen. Das Konvergenzkriterium (4.33) ist dann in jedem Fall erfüllt, da $\varepsilon_f < \varepsilon_{tol}$. Wird ε_{tol} hingegen nie unterschritten, endet der Algorithmus nach einer definierten maximalen Anzahl von Iterationen.

Die einzelnen Optimierer erhalten die gleichen Einstellungen wie bei der vorangegangenen direkten Optimierung ohne Antwortflächen in Abschnitt 4.2.2. Zu Beginn werden $J_0^{(s)} = 30$ initiale Stützstellen generiert. Der Algorithmus endet, wenn nach 10 Iterationen keine Verbesserung der Approximationsqualität oder eine Gesamtzahl von 50 Iterationen erreicht wird. In jeder Iteration wird eine maximale Anzahl von $M = 20$ neuen Stützstellen zur Stützstellenmenge $I_i^{(s)}$ hinzugefügt. Die Fehlertoleranz beträgt $\varepsilon_{tol} = 5\,\%$.

Die Optimierung benötigt $i = 29$ Iterationen und 1656 Funktionsauswertungen und resultiert in den Ergebnissen in Abb. 4.11. Im direkten Vergleich zu den Ergebnissen der direkten Optimierung in Abb. 4.7 fällt auf, dass zum einen die gleiche Architektur gefunden wird, also sämtliche Architekturvariablen den gleichen Wert aufweisen, und sich zum anderen die gefundenen Pareto-Fronten sehr ähneln. Diese werden maßgeblich durch die Architektur bestimmt, weshalb bei gleichen Werten der Architekturvariablen lediglich die Segmentvariablen

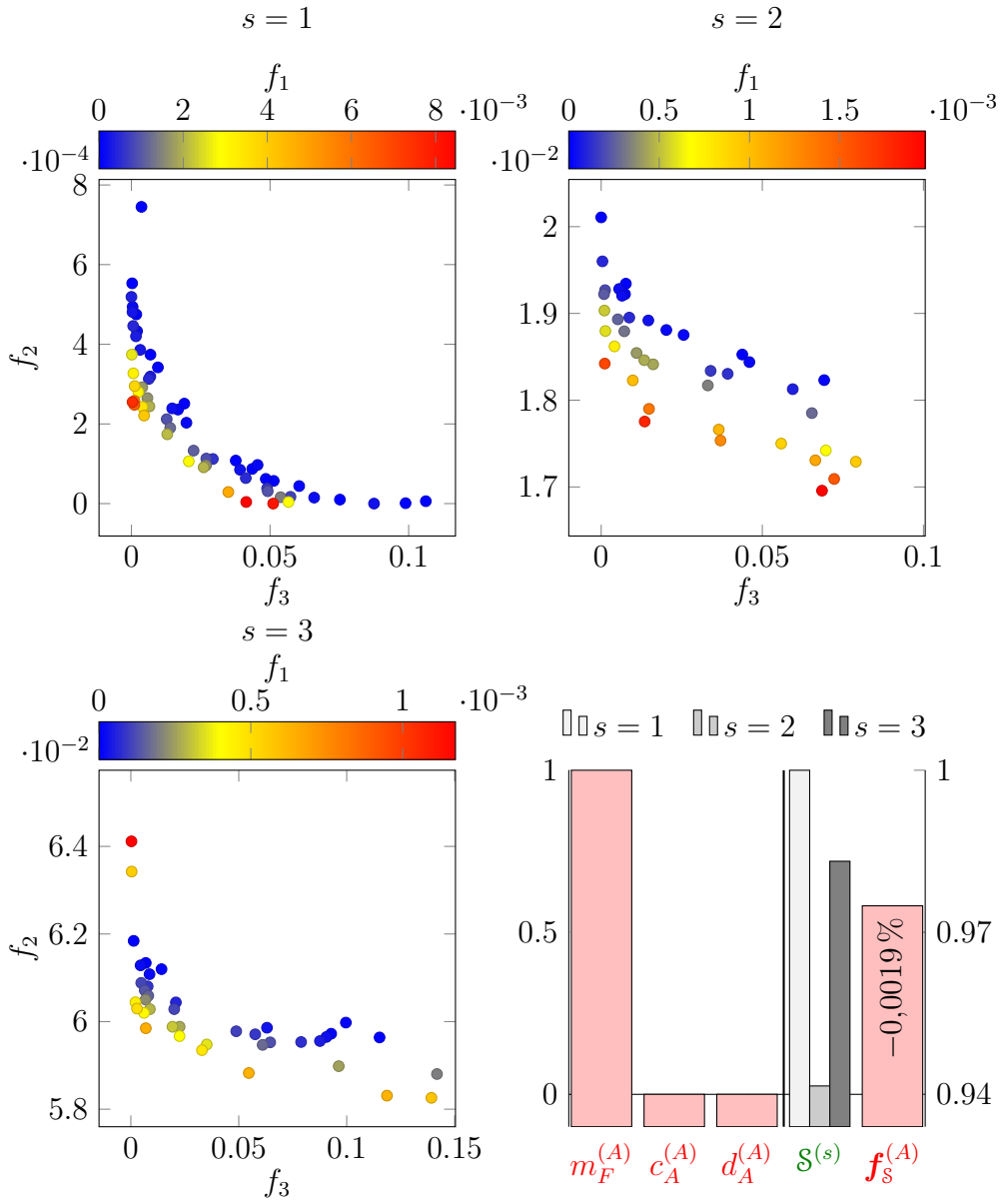

Abbildung 4.11: Optimale Ergebnisse der ARSM Strategie

blieben, um die Diversität und Besetzung der Fronten auszubilden. Der große Unterschied liegt nun darin, dass die Pareto-Fronten der ARSM-Strategie deutlich dünner besetzt sind, da in jeder Iteration nur maximal $M = 20$ Pareto-optimale Stützstellen direkt berechnet werden, um die Anzahl aufwändiger Funktionsauswertungen gering zu halten. Während bei der direkten Strategie mehrere tausend Pareto-optimale Entwürfe gefunden werden, sind es bei der effizienteren Strategie nur insgesamt 120. Das bedeutet, dass in 6 Iterationen die gleiche Architektur gefunden wird und in jeder die dargestellten Pareto-Fronten dichter besetzt werden.

Zusammenfassend kann festgehalten werden, dass die neue Strategie deutlich weniger direkte Funktionsauswertungen benötigt und dabei die gleiche Architektur findet. Im Vergleich zur direkten Strategie verliert das zu maximierende Architekturkriterium $f_S^{(A)}$ lediglich $-0{,}0019\,\%$ an Qualität. Dieser Verlust ist dadurch zu erklären, dass die Pareto-Fronten weniger dicht besetzt sind und die Hypervolumina damit marginal kleiner ausfallen. Im Rahmen einer Entwicklung können die Fronten der jeweiligen Segmente auf Basis der gefundenen Architektur jedoch nachträglich gezielt verbessert werden, falls eine dichtere Besetzung gewünscht ist.

4.3.2 Lösung mit Subraumoptimierungen für unterschiedliche Segmente

Die als Concurrent Subspace Optimization (CSSO) bezeichnete Strategie wird üblicherweise in der Multidisciplinary Optimization (MDO) eingesetzt, bei der verschiedene Ingenieursdisziplinen in einen Optimierungsprozess integriert werden [73]. Die Grundidee der CSSO ist es, optimale Lösungen für komplexe, hoch integrierte und aus voneinander abhängigen Teilproblemen bestehende Optimierungsprobleme zu finden. Die Kopplung resultiert dabei aus gemeinsamen und spezifischen Variablen und gekoppelten Kriterien. Die Probleme der vorliegenden Arbeit beziehen sich zwar nur auf Optimierungen mit lediglich einer Disziplin, dennoch ist die Charakteristik der Architekturoptimierungsprobleme mit vereinheitlichten und spezifischen Entwurfsvariablen, unterschiedlichen Berechnungsmodellen und teilweise gekoppelten Kriterien ähnlich einem MDO-

Problem.

Eine CSSO besteht aus drei wesentlichen Teilen: Zwei davon, die Antwortflächenmodellierung und die kaskadierte Optimierung, sind vergleichbar mit denen der bereits vorgestellten ARSM-Strategie. Lediglich ein dritter Teil muss hinzugefügt werden, um einen auf CSSO basierenden Prozess zu erhalten. Hierbei handelt es sich um die sogenannten Subraumoptimierungen (SSO), die im Rahmen der Vorstellung des in Abb. 4.12 dargestellten Prozesses genauer erläutert werden.

Die generelle Aufgabe der Subraumoptimierungen liegt vornehmlich darin, neue Stützstellen zur Verbesserung der Antwortflächen zu generieren. Am Anfang des Prozesses werden für jedes Segment analog zur ARSM-Strategie in Abschnitt 4.3.1 eine initiale Stützstellenmenge (4.26) generiert und die enthaltenen Entwürfe direkt berechnet. Anschließend werden die aus Darstellungsgründen vereinfacht als graue Box dargestellten Subraumoptimierungen durchgeführt, deren Ablauf in Abb. 4.13 abgebildet ist.

Zunächst werden für die Subraumoptimierungen Antwortflächen (4.27) gebildet. Eine Subraumoptimierung betrachtet dann ein einzelnes Segment, das vollen Zugriff auf die zugehörigen Segment- und Architekturvariablen hat. Für jedes Segment s wird das Optimierungsproblem

$$\min_{\boldsymbol{q}^{(s)} \in Q^{(s)}} \hat{\boldsymbol{f}}_{\mathrm{S}}^{(A,s)} \left(\boldsymbol{q}^{(s)} \right) \text{ mit } Q^{(s)} = \left\{ \boldsymbol{q}^{(s)} \in \mathbb{R}^{h^{(A)}+h^{(s)}} \middle| \boldsymbol{q}_u^{(s)} \leq \boldsymbol{q}^{(s)} \leq \boldsymbol{q}_o^{(s)} \right\} \quad (4.34)$$

mit dem Vektorkriterium (4.36) gelöst, wobei $\boldsymbol{q}^{(s)}$ nach (4.8) für jedes Segment separat variierbare Architekturvariablen enthält, also volle Entwurfsflexibilität gegeben ist.

Für jeden Entwurf $\boldsymbol{q}^{(s)}$ wird das jeweilige Segment s mit $\hat{\boldsymbol{f}}^{(s,0)} = \hat{\boldsymbol{f}}^{(s,0)} \left(\boldsymbol{q}^{(s)} \right)$ auf der entsprechenden Antwortfläche berechnet. Vom Prinzip her entspricht dies der in Abschnitt 4.1.1 behandelten Null-Architekturoptimierung mit voller Entwurfsflexibilität. Nach Abb. 4.1 ist es so aber nahezu unmöglich, eine einheitliche Architektur zu finden. Um also nicht für das einzelne Segment optimale und für die gesamte Architektur ungeeignete Stützstellen zu generieren,

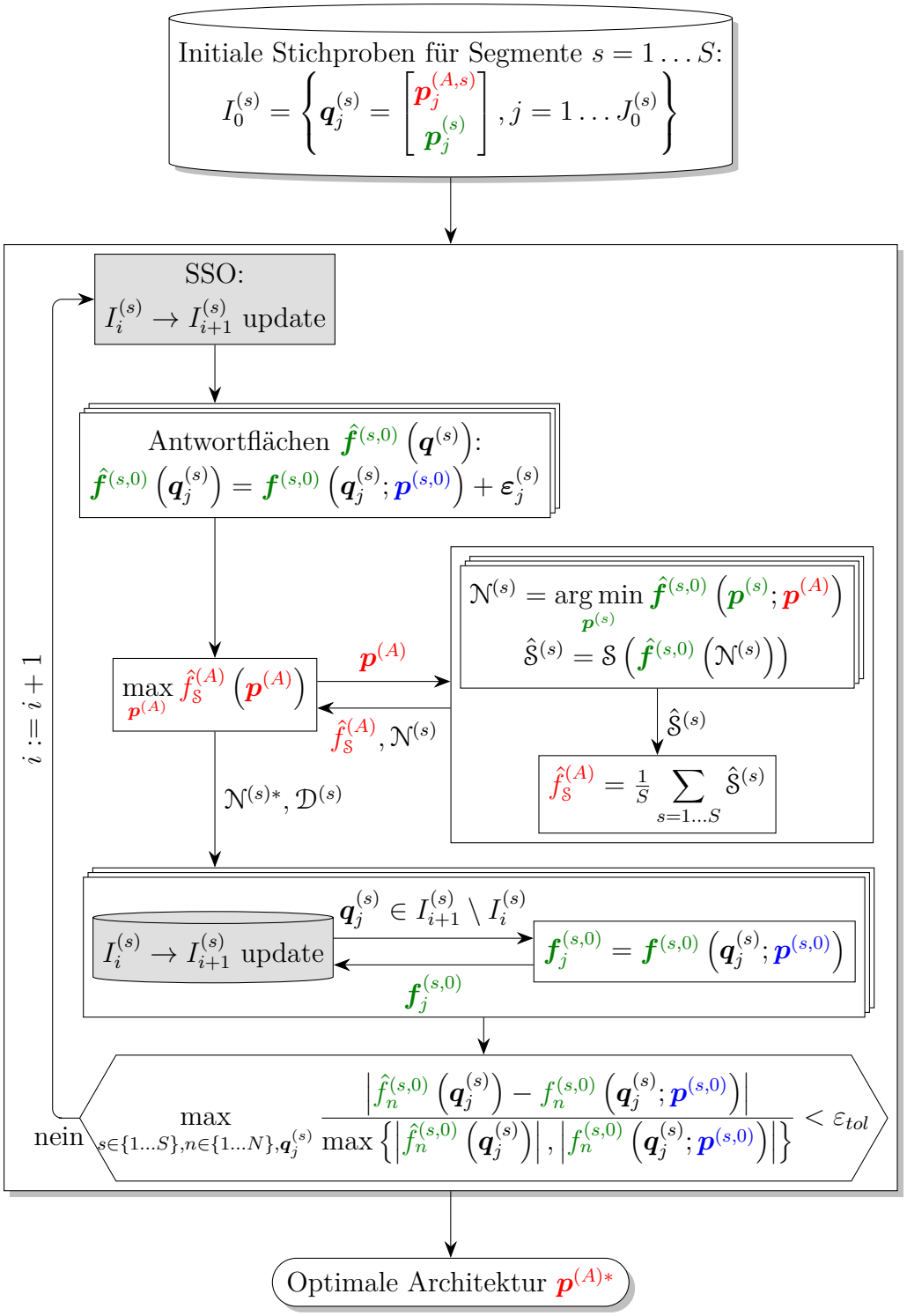

Abbildung 4.12: Ablauf der auf Subraumoptimierungen basierenden Architekturoptimierung

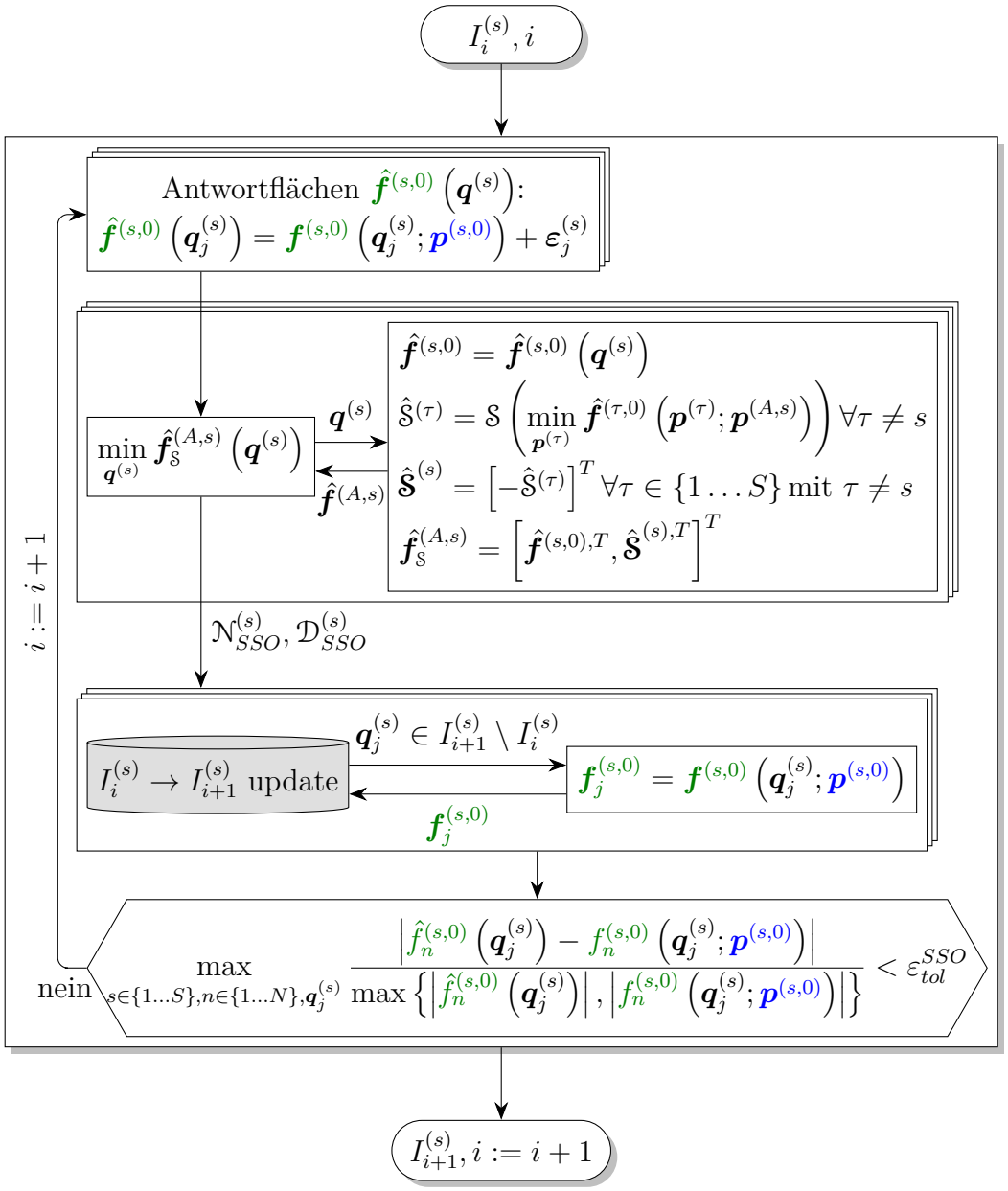

Abbildung 4.13: Ablauf der Subraumoptimierungen

werden die anderen Segmente $\tau \in \{1 \dots S\}$ mit $\tau \neq s$ für den aktuellen in $\boldsymbol{q}^{(s)}$ enthaltenen Architekturentwurf $\boldsymbol{p}^{(A,s)}$ optimiert:

$$\min_{\boldsymbol{p}^{(\tau)} \in P^{(\tau)}} \hat{\boldsymbol{f}}^{(\tau,0)} \left(\boldsymbol{p}^{(\tau)}; \boldsymbol{p}^{(A,s)} \right) \text{mit } P^{(\tau)} = \left\{ \boldsymbol{p}^{(\tau)} \in \mathbb{R}^{h^{(\tau)}} \middle| \boldsymbol{p}_u^{(\tau)} \leq \boldsymbol{p}^{(\tau)} \leq \boldsymbol{p}_o^{(\tau)} \right\}. \tag{4.35}$$

Die Bezeichnung $\boldsymbol{p}^{(A,s)}$ ist hierbei bewusst gewählt, da es sich im Rahmen des gesamten Prozesses um Architekturvariablen handelt, diese aber in den Subraumoptimierungen wie Segmentvariablen behandelt werden. Mit (4.30) werden dann die S-Metriken $\hat{\mathbb{S}}^{(\tau)}$ auf Basis der Menge gefundener Pareto-optimaler Entwürfe berechnet. Schlussendlich wird das in (4.34) optimierte Kriterium der Subraumoptimierung mit

$$\hat{\boldsymbol{f}}_{\mathbb{S}}^{(A,s)} = \begin{bmatrix} \hat{\boldsymbol{f}}^{(s,0)} \\ \hat{\boldsymbol{\mathbb{S}}}^{(s)} \end{bmatrix} \tag{4.36}$$

gebildet. Der Vektor $\hat{\boldsymbol{\mathbb{S}}}^{(s)}$ enthält alle S-Metriken $\hat{\mathbb{S}}^{(\tau)}$ entsprechend

$$\hat{\boldsymbol{\mathbb{S}}}^{(s)} = \left[-\hat{\mathbb{S}}^{(\tau)} \right]^T \forall \tau \in \{1 \dots S\} \text{ mit } \tau \neq s, \tag{4.37}$$

wobei diese negativ behandelt werden, da sie nach Abb. 4.6 nicht minimiert, sondern maximiert werden sollen. Die Subraumoptimierungen (4.34) werden für jedes Segment s parallel ausgeführt, weshalb die Strategie auch *Concurrent Subspace Optimization* heißt.

Ergebnis dieser Subraumoptimierungen sind Mengen $\mathcal{N}_{SSO}^{(s)}$ mit gefundenen Pareto-optimalen und $\mathcal{D}_{SSO}^{(s)}$ mit den restlichen Metrik-dominierten Entwürfen. Für jedes Segment s werden nun analog zur ARSM-Strategie nach Abb. 4.8 neue Stützstellen bestimmt, direkt berechnet und zur Menge vorhandener Stützstellen $I_i^{(s)}$ hinzugefügt. Dieser Prozess wird so lange durchgeführt, bis das Konvergenzkriterium (4.33) zum wiederholten Male erfüllt oder eine maximale Anzahl von Iterationen erreicht ist. Die Maximalanzahl der Iterationen ist hierbei geringer und die Fehlertoleranz ε_{tol}^{SSO} höher gewählt als bei der ARSM-Strategie und dem übergeordneten Prozess der CSSO-Strategie, da der Fokus nur auf der Verbesserung der Antwortflächen liegt und der größere Teil des Aufwands in die Suche der optimalen Architektur investiert werden sollte.

Kurz zusammengefasst ist die Aufgabe der Subraumoptimierungen die gezielte Verbesserung der Antwortflächen. Hierbei wird nicht wie bei der ARSM-Strategie versucht, die Antwortflächen im Bereich der vermuteten optimalen Architektur, sondern entlang der Pareto-Front aller möglichen Architekturen zu verbessern. Die Wahrscheinlichkeit, in einem lokalen Architekturminimum zu enden, soll hierdurch im Vergleich zur ARSM-Strategie verringert werden. Da jedes Segment s die Architekturvariablen $\boldsymbol{p}^{(A,s)}$ für sich selbst optimiert, ist das Ziel einer einheitlichen Architektur allerdings nahezu unmöglich zu erreichen. Es wird deshalb ein letzter Koordinationsschritt notwendig, der entsprechend Abb. 4.12 nach den Subraumoptimierungen durchgeführt wird.

Zunächst werden die verbesserten Stützstellenmengen der Subraumoptimierungen $I_i^{(s)}$ zur Bildung von aktuellen Antwortflächen $\hat{\boldsymbol{f}}^{(s,0)}\left(\boldsymbol{q}^{(s)}\right)$ herangezogen. Auf diesen Antwortflächen wird dann analog zur ARSM-Strategie eine kaskadierte Architekturoptimierung (4.28) durchgeführt. Wie bereits erwähnt, ist dieser Koordinationsschritt notwendig, um den Algorithmus zu zwingen, eine einheitliche Architektur zu bestimmen. Es wird daher auch ein leicht von Abb. 4.8 modifizierter Auswahlprozess neuer Stützstellen verwendet. Das Ziel der Koordination ist es nicht, geeignete Stützstellen zur Antwortflächenverbesserung zu bestimmen, wofür die Subraumoptimierungen zuständig sind. Um mit der S-Metrik vergleichbare Pareto-Fronten zu erhalten, wird für die koordinierte Architekturoptimierung der in Abb. 4.14 dargestellte Prozess verwendet.

Wichtig ist, dass, wie zuvor, zuerst möglichst weit von bisherigen Stützstellen entfernte Entwürfe gewählt, aber lediglich identische verworfen werden. Die charakteristische Distanz entspricht folglich $\hat{d} = 0$. Weiterhin wird die Pareto-optimale Entwürfe enthaltende Menge $\mathcal{N}^{(s)*}$ nicht um die Menge Metrik-dominierter Entwürfe $\mathcal{D}^{(s)}$ erweitert. Auf diese Weise werden nur Pareto-optimale Entwürfe gewählt, damit nach Beendigung des Algorithmus eine vergleichbare Bewertung der Fronten durchgeführt werden kann. Aufgrund der abstandsabhängigen Wahl neuer Stützstellen wird dennoch ein Beitrag zur gezielten Verbesserung der Antwortflächen geleistet. Die restlichen Prozessschritte in Abb. 4.12 gleichen denen der ARSM-Strategie.

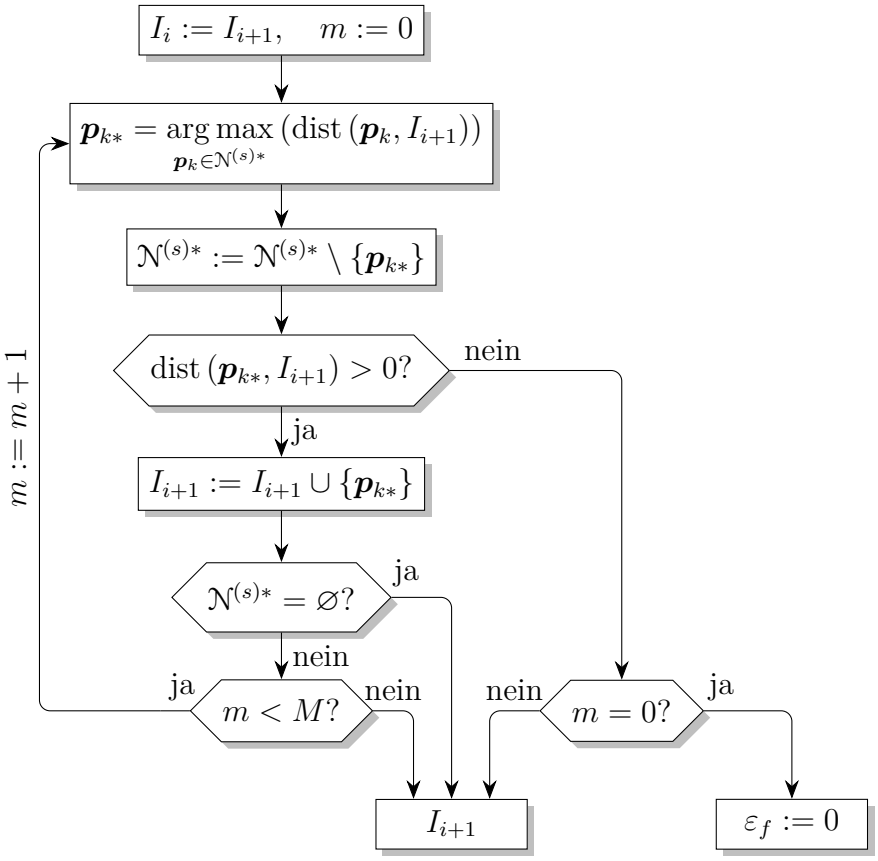

Abbildung 4.14: Bestimmung neuer Stützstellen des Koordinationsschrittes

Die jeweiligen Optimierungsalgorithmen werden mit den gleichen Einstellungen wie bei der ARSM-Strategie in Abschnitt 4.3.1 ausgeführt. In den zusätzlichen Subraumoptimierungen werden maximal 5 Iterationen durchgeführt. Wenn nach 3 Iterationen keine weitere Verbesserung der Fehlertoleranz erreichbar ist, werden die Subraumoptimierungen als konvergiert angenommen. Die Fehlertoleranz wird mit $\varepsilon_{tol}^{SSO} = 0.1$ doppelt so groß wie im äußeren Prozess gewählt. In jeder Iteration werden maximal $M = 5$ neue Stützstellen bestimmt und direkt berechnet.

Der Prozess endet nach insgesamt 1806 direkten Funktionsauswertungen, wovon bei 80 Entwürfen die gleiche Architektur gefunden wird. Diese Entwürfe repräsentieren die in den Ergebnissen Abb. 4.15 dargestellten Pareto-Fronten. Auch diese Strategie findet die gleiche Architektur wie die direkte und die ARSM-Strategie. Die Pareto-Fronten sind im Vergleich zu den mit der ARSM-Strategie gefundenen 120 Entwürfen allerdings etwas dünner besetzt. Aufgrund

der geringeren Besetzungsdichte verliert das Architekturkriterium $f_{\mathbb{S}}^{(A)}$ der CSSO-Strategie im Vergleich zur direkten Lösung mit $-0{,}0082\,\%$ etwas mehr an Qualität als die ARSM-Strategie. Wie bereits erwähnt, liegt der Fokus aber auf dem Finden einer optimalen Architektur, was trotz deutlich gesteigerter Effizienz im Vergleich zur direkten Lösung gelungen ist.

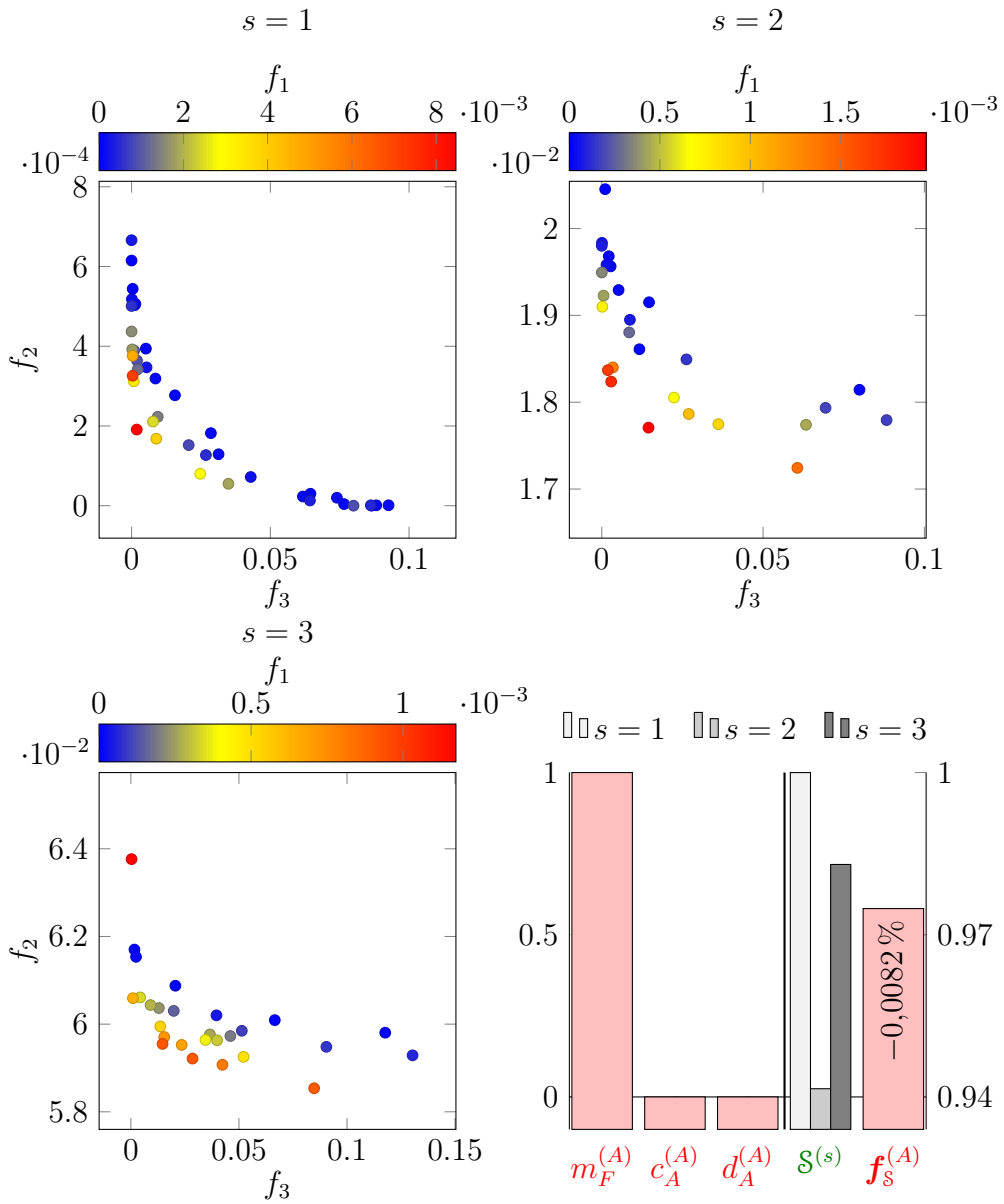

Abbildung 4.15: Optimale Ergebnisse der CSSO-Strategie

4.3.3 Zusammenfassung und Diskussion der Ergebnisse

In diesem Abschnitt wurden verschiedene, auf dem Konzept der kaskadierten Optimierung basierende Strategien zur Lösung von mehrkriteriellen Architekturoptimierungsproblemen unter dem Fokus der Effizienzsteigerung analysiert. Zunächst ist festzuhalten, dass alle untersuchten Strategien geringfügig schlechter abschneiden als eine direkt durchgeführte mehrkriterielle kaskadierte Architekturoptimierung. Die wichtigsten Maße zur Beurteilung der Optimierungsstrategien, wie Anzahl der Funktionsauswertungen, erreichte Leistungsfähigkeit der Architektur und Verlust gegenüber der direkten Lösung, sind in Tabelle 4.3 dargestellt.

Strategie	# direkte Funktionsauswertungen	endgültiger $f_S^{(A)}$ Wert	Verlust zur kaskadierten Opt.
kaskadiert	$5{,}340 \cdot 10^6$	0,974 92	–
ARSM	1656	0,974 90	−0,0019 %
CSSO	1806	0,974 84	−0,0082 %

Tabelle 4.3: Vergleich der Ergebnisse der mehrkriteriellen Architekturoptimierungen

Beide Strategien, die auf reinen adaptiven Antwortflächen und die auf Subraumoptimierungen basierenden, sind verglichen mit der direkten Strategie deutlich effizienter und erzielen annähernd so gute Lösungen. In Zahlen gesprochen benötigen die ARSM und die CSSO-Strategie etwa um den Faktor 3000 weniger direkte Funktionsauswertungen als die direkte Optimierung, während der prozentuale Verlust des auf den S-Metriken basierenden Architekturkriteriums $f_S^{(A)}$ vernachlässigbar klein ist.

Die ARSM-Strategie erweist sich bei dem untersuchten Testproblem als etwas effizienter und genauer als die CSSO-Strategie. Sie benötigt 150 weniger direkte Funktionsauswertungen und erzielt mit 40 zusätzlichen Pareto-optimalen Entwürfen eine dichtere Besetzung der Pareto-Fronten. Der Vorteil der Stützstellengenerierung entlang der globalen Pareto-Front der Subraumoptimierung

zeigt sich in dieser Untersuchung nicht. Werden die Anzahl der benötigten direkten Funktionsauswertungen und das erreichte Optimum in Relation zu denen der direkten Strategie gesehen, sind die Unterschiede zwischen ARSM- und CSSO-Strategie verschwindend gering, so dass alleine aus diesen Größen heraus keine Empfehlung ausgesprochen werden kann.

Generell bietet die CSSO-Strategie mehr Flexibilität und kann beispielsweise durch das Überspringen der Subraumoptimierungen in eine ARSM-Strategie umgewandelt werden. Prinzipiell sollte die CSSO-Strategie aufgrund der zusätzlichen Subraumoptimierungen weniger anfällig gegenüber lokaler Architekturminima sein, da die Antwortflächen entlang der globalen Pareto-Front der Architektur verbessert werden, während die ARSM-Strategie nur bei wiederholter Besetzung derselben als optimal vermuteten Front neue und in anderen Regionen liegende Stützstellen generiert. Es ist deshalb zu empfehlen, wenn Vorwissen über das zu optimierende Problem vorhanden ist und es keine lokalen Minima gibt, die ARSM-Strategie zu verwenden. Handelt es sich um ein neues, unbekanntes Problem oder ist das Auftreten von lokalen Minima wahrscheinlich, so sollte die etwas aufwändigere CSSO-Strategie gewählt werden. Aus diesen Gründen wird die CSSO-Strategie mit adaptiven Antwortflächen für die nachfolgenden Analysen ausgewählt.

Kapitel 5

Robuste Architekturoptimierung unter Berücksichtigung mehrerer Derivate

Ein großer Nachteil der deterministischen Optimierung ist die unbekannte Robustheit des optimierten Systems gegenüber Unsicherheiten. Aus diesem Grund wurde das Konzept der Robust Design Optimization (RDO) als Kombination aus Robust Design Analysis (RDA) und deterministischer Optimierung entwickelt. Wie in Abschnitt 3.3 erwähnt, gibt es verschiedene Interpretationsansätze, mit denen eine RDO auf die Architekturoptimierung adaptiert werden kann. Im Folgenden wird der Ansatz verfolgt, dass eine optimale Architektur für mehrere Derivate mit jeweils unterschiedlichen Ausstattungsvarianten gefunden werden soll. Generell kann die untersuchte Strategie aber auch für die anderen Ansätze der robusten Architekturoptimierung eingesetzt werden.

Zunächst werden verschiedene grundlegende Strategien für die robuste Optimierung und der Einsatz von Antwortflächen diskutiert. Die in Abschnitt 4.3.1 vorgestellte CSSO-Strategie wird dann erweitert, um eine robuste Architekturoptimierung des um unsichere Parameter erweiterten Architekturproblems durchzuführen und mit einer direkt optimierten Lösung zu vergleichen.

5.1 Strategien für die robuste Optimierung

Das Ziel einer RDO ist die Optimierung eines gegebenen Gütekriteriums, während gewisse Randbedingungen mit spezifischen Sicherheitsfaktoren eingehalten oder die durch Unsicherheiten entstehende Varianz des Gütekriteriums minimiert werden. Die robuste Optimierung kann in die stochastische Entwurfsauslegung und Fuzzy Logic unterteilt werden, wobei sich die vorliegende Arbeit auf die stochastische Entwurfsauslegung bezieht. In diesem Bereich gibt es auf Varianz- oder Zuverlässigkeits-basierte Prozesse [14].

Die Wahl einer dieser Techniken hängt von dem spezifischen Einsatzgebiet und dem zu erreichenden Ziel ab. Zuverlässigkeitsanalysen sind zu empfehlen, wenn eine vordefinierte und sehr niedrige Ausfallwahrscheinlichkeit eines Systems nachgewiesen werden soll. Varianzbasierte Analysen benötigen eine akzeptable Anzahl von Funktionsauswertungen und bieten einen Nachweis für Sicherheitsgrenzen bis zu Standardabweichungen von 3σ [53]. Die nachfolgenden Analysen beziehen sich auf varianzbasierte Methoden, weshalb sich die anschließende Benutzung des Begriffes bzw. der Abkürzung RDO auf dieses Themenfeld bezieht.

In der Literatur werden zwei grundlegende varianzbasierte RDO Prozeduren beschrieben: Der integrierte Ansatz, der auch als gekoppelte Prozedur bezeichnet wird, und der iterative Ansatz [53, 70]. Diese beiden Strategien werden nun vorgestellt und eine Auswahl für die weiteren Analysen getroffen.

Iterative RDO führen zunächst eine deterministische Optimierung durch, ohne Robustheitskriterien zu berücksichtigen. Die Robustheit des Systems wird hier erst nach der Optimierung bewertet, wobei üblicherweise Sicherheitsfaktoren eingeführt werden, die es einzuhalten gilt. Wenn ein deterministisches Optimum eine der vordefinierten robustheitsabhängigen Randbedingungen verletzt, werden die Sicherheitsfaktoren aktualisiert, um ein zulässiges Optimum in der nächsten Iteration zu erreichen. Dieses Vorgehen wird wiederholt, bis eine maximale Anzahl von Iterationen oder ein Konvergenzkriterium erreicht ist. Das Flussdiagram dieser Prozedur ist in Abb. 5.1a dargestellt. Der Vorteil

des iterativen Prozesses ist, dass die Robustheitsanalyse nur einmal in jeder Iteration des äußeren Kreislaufes, nämlich für jedes deterministische Optimum, durchgeführt wird. Ein in Bezug auf minimale Streuungen von Systemantworten optimierter robuster Entwurf kann mit dieser Methode allerdings nicht erreicht werden, da die Robustheitskriterien nicht direkt in der Optimierung berücksichtigt werden.

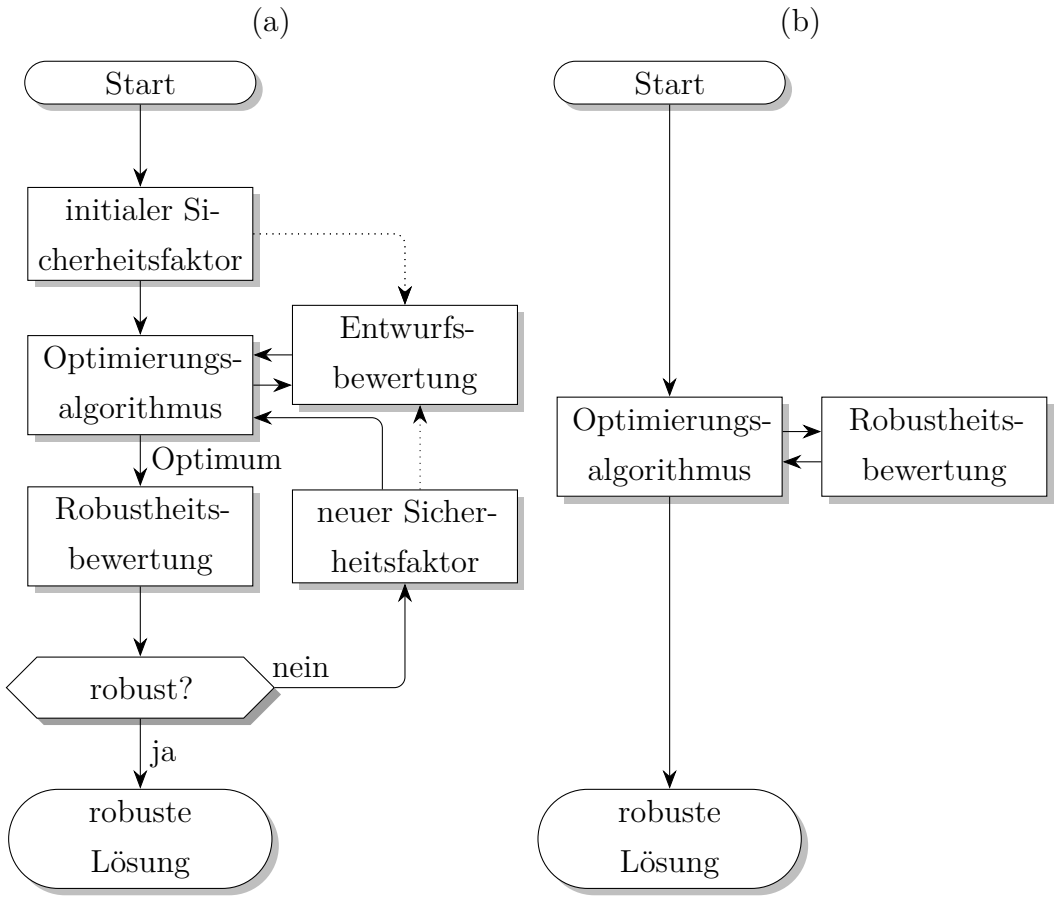

Abbildung 5.1: Ablauf der (a) iterativen und (b) integrierten RDO-Strategie

Wenn Robustheitskriterien direkt in der Optimierung verwendet werden, wird von einer integrierten RDO gesprochen. Hierbei wird für jeden vom Optimierer generierten Entwurf eine Robustheitsanalyse durchgeführt. Die integrierte RDO Prozedur ist in Abb. 5.1b dargestellt. Bei diesem Ansatz können robustheitsbasierte Randbedingungen berücksichtigt und zeitgleich Varianzen und Erwartungswerte der Systemantworten optimiert werden. Neben der größeren Flexibilität bei der Gestaltung der Optimierungsaufgabe benötigt diese

Strategie jedoch deutlich mehr Rechenzeit wegen eines höheren Bedarfs an zeitintensiven direkten Funktionsauswertungen.

In den folgenden Analysen wird Robustheit in Bezug auf die Minimierung von Varianzen und Erwartungswerten benötigt, ohne konkrete Sicherheitsfaktoren einhalten zu müssen. Es wird deshalb die integrierte Strategie gewählt, auch wenn mehr Aufwand in die Reduzierung der Anzahl der direkten Funktionsauswertungen investiert werden muss.

5.2 Formulierung eines robusten Optimierungsproblems

In diesem Abschnitt wird zunächst ein robustes Optimierungsproblem ohne die Berücksichtigung von Architekturen eingeführt. Der Fokus liegt auf der Optimierung der Robustheit und der damit verbundenen Besonderheiten. Hierfür wird eine konkrete Testfunktion mit einer Strategie für effiziente robuste Optimierung optimiert.

Das Ziel der robusten Optimierung ist die Minimierung von Erwartungswert und Standardabweichung einer Gütefunktion $f(\boldsymbol{p};\boldsymbol{r})$ mit einer gegebenen Menge von h Parametern p_i, die in einem Entwurfsvektor $\boldsymbol{p}$ mit den Grenzen $\boldsymbol{p}_u$ und $\boldsymbol{p}_o$ zusammengefasst sind. Zusätzlich gibt es stochastische Variablen $\boldsymbol{r}$, die durch beliebige Verteilungsfunktionen beschrieben sein können, beispielsweise durch eine Normalverteilung $\boldsymbol{r} \sim N(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$ mit Erwartungswert $\boldsymbol{\mu}_r$ und Kovarianzmatrix $\boldsymbol{\Sigma}_r$. Die Formulierung des RDO-Problems lautet dann

$$\min_{\boldsymbol{p}\in P} f(\boldsymbol{p};\boldsymbol{r}) \text{ mit } P = \left\{\boldsymbol{p} \in \mathbb{R}^h \middle| \boldsymbol{p}_u \leq \boldsymbol{p} \leq \boldsymbol{p}_o\right\}. \tag{5.1}$$

Wie bereits erwähnt, wird für die folgenden Untersuchungen der integrierte RDO-Ansatz verwendet, wobei die stochastische Variable $\boldsymbol{r}$ durch das simultane Suchen nach einem optimalen Erwartungswert $\mu_f(\boldsymbol{p}) = E[f(\boldsymbol{p};\boldsymbol{r})]$ und minimaler Streuung $\sigma_f(\boldsymbol{p}) = \sqrt{E\left[\left(f(\boldsymbol{p};\boldsymbol{r}) - \mu_f\right)^2\right]}$ eliminiert wird. Hieraus resultiert das bikriterielle robuste Optimierungsproblem

$$\min_{\boldsymbol{p}\in P} \boldsymbol{f}_{RDO}(\boldsymbol{p}) \text{ mit } P = \left\{\boldsymbol{p} \in \mathbb{R}^h \middle| \boldsymbol{p}_u \leq \boldsymbol{p} \leq \boldsymbol{p}_o\right\} \tag{5.2}$$

mit

$$\boldsymbol{f}_{RDO}\left(\boldsymbol{p}\right) := \begin{bmatrix} \mu_f\left(\boldsymbol{p}\right) \\ \sigma_f\left(\boldsymbol{p}\right) \end{bmatrix}. \tag{5.3}$$

Für die effiziente Lösung dieser Art von Problemen gibt es verschiedene Ansätze in der Literatur. Die meisten RDO-Prozeduren setzen Antwortflächen ein, um den Berechnungsaufwand zu reduzieren. In Fällen, in denen nur die Entwurfsvariablen streuen, bietet es sich an, eine sich über alle Variablen erstreckende globale Antwortfläche zu verwenden. Die Gütekriterien können dann auf dieser Antwortfläche berechnet und die zeitaufwändige Bestimmung der Robustheitskriterien deutlich beschleunigt werden. Gleichwohl ist, wie von Most und Will [53] diskutiert, eine hohe Approximationsqualität notwendig, um vertrauenswürdige Ergebnisse zu erzielen.

Cheng und Lin [19] verwenden eine solche nicht adaptive globale Antwortfläche für die mehrkriterielle robuste Auslegung von Pkw-Fahrwerken. RSM wird hierbei auf Basis einer initialen Stützstellenmenge durchgeführt, die aus einem DoE gewonnen wird. Kang et al. [42] und Park et al. [58] nutzen eine adaptive Strategie. Die Erwartungswerte und Standardabweichungen der Gütekriterien werden gewichtet und aufsummiert, um eine einkriterielle RDO zu erhalten. Die Optimierung wird dann auf den Antwortflächen durchgeführt und der beste gefundene Entwurf direkt berechnet und zur Menge der Stützstellen hinzugefügt, um anschließend eine neue Antwortfläche aufzubauen. Yang et al. [95] benutzen einen ähnlichen Ansatz, wobei eine sogenannte Dual-RSM Methode, die Erwartungswert und Varianz getrennt voneinander approximiert, eingesetzt wird. Eine weitere Methode, um im Rahmen der robusten Optimierung von Antwortflächen zu profitieren, ist die Verwendung lokaler Approximationen, wie sie von Busch und Bestle [15] vorgestellt wird. Um die Bewertungsqualität der Robustheitskriterien zu verbessern, wird für jeden deterministischen Entwurf eine Stichprobe der unsicheren Parameter generiert, berechnet und eine lokale Antwortfläche aufgebaut. Die Berechnung der Kriterien (5.3) wird dann mit einer großen Anzahl von Stichproben auf diesem Modell durchgeführt. Die Anzahl der direkten Funktionsauswertungen wird so reduziert, während die Schätzgenauigkeit von Erwartungswert und Varianz steigt.

Im Rahmen dieser Arbeit wird die bereits bewährte adaptive Strategie in Kombination mit separaten Antwortflächen für Erwartungswert und Standardabweichung verwendet. Im nächsten Abschnitt wird die Idee der robusten Optimierung in Kombination mit adaptiven Antwortflächen anhand einer anschaulichen Testfunktion aufgezeigt.

5.2.1 Robuste Optimierung einer einfachen Testfunktion

Für die robuste Optimierung wird eine einfache Testfunktion [90] verwendet, in der deterministische und robuste Optima klar identifizierbar sind und die sehr schnell berechnet werden kann. Gelöst wird das Optimierungsproblem (5.1), wobei die Gütefunktion

$$f(\boldsymbol{p};\boldsymbol{r}) = \sum_{i=1}^{2} f_{simple}(p_i + r_i) \tag{5.4}$$

wie folgt berechnet wird:

$$f_{simple}(x) = \begin{cases} \cos\frac{x}{2} + 1 & \text{für } -2\pi \leq x < 2\pi, \\ 1.1\cos(x+\pi) + 1.1 & \text{für } 2\pi \leq x < 4\pi, \\ 0 & \text{sonst.} \end{cases} \tag{5.5}$$

Die stochastischen Variablen r_i sind unkorreliert und normalverteilt mit dem Erwartungswert $\mu_{r,i} = 0$ und der Varianz $\sigma_{r,i}^2 = 0.164$. Der mögliche Wertebereich von r_i wird auf $\pm 2\sigma_{r,i}$ beschränkt. Die Problemformulierung (5.2), (5.3) wird dann mithilfe einer adaptiven Strategie gelöst, wobei der Erwartungswert nicht minimiert, sondern maximiert wird:

$$\begin{bmatrix} \max\limits_{\boldsymbol{p}\in P} \mu_f(\boldsymbol{p}) \\ \min\limits_{\boldsymbol{p}\in P} \sigma_f(\boldsymbol{p}) \end{bmatrix} \text{ mit } P = \left\{ \boldsymbol{p} \in \mathbb{R}^2 | -2\pi \leq p_i \leq 4\pi \right\}. \tag{5.6}$$

Die Testfunktion (5.4) ist in Abb. 5.2 für $\boldsymbol{r} \equiv \boldsymbol{0}$ dargestellt.

Das robuste Optimierungsproblem wird mit einer angepassten Variante der in Abb. 4.10 dargestellten ARSM-Strategie aus gelöst. Zunächst wird eine initiale

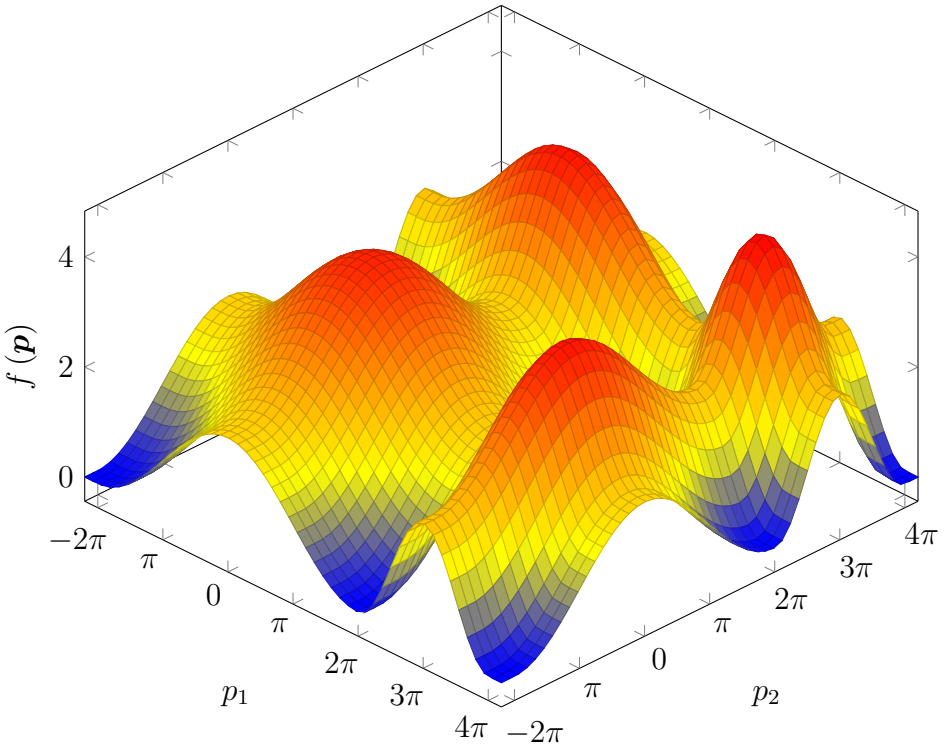

Abbildung 5.2: Darstellung der Testfunktion (5.4) für $\boldsymbol{r} \equiv \boldsymbol{0}$

Stützstellenmenge (4.23) mit dem ALHS für $i = 0$ generiert. Wichtig ist hierbei, dass es keine Architektur oder Segmente, sondern nur eine Stützstellenmenge I_i gibt. Auf Basis der Stützstellen $\boldsymbol{p}_j \in I_0$ und der zugehörigen Funktionswerte $\boldsymbol{f}_{RDO}(\boldsymbol{p}_j)$ werden dann Antwortflächen $\hat{\boldsymbol{f}}_{RDO}(\boldsymbol{p}) = \boldsymbol{f}_{RDO}(\boldsymbol{p}) + \boldsymbol{\varepsilon}$ gebildet. Die Schätzung von $\boldsymbol{f}_{RDO}(\boldsymbol{p}_j)$ nach (5.3) benötigt zusätzlich eine Stichprobe der stochastischen Variablen $\boldsymbol{r}$, die nach dem gleichen Verfahren wie die initialen Entwürfe $\boldsymbol{p}_j$ mit dem ALHS generiert wird, wobei aber die zugrundeliegenden Verteilungsfunktionen berücksichtigt werden.

Im nächsten Schritt wird das Optimierungsproblem (5.2) auf den Antwortflächen gelöst:

$$\min_{\boldsymbol{p} \in P} \hat{\boldsymbol{f}}_{RDO}(\boldsymbol{p}) \text{ mit } P = \left\{ \boldsymbol{p} \in \mathbb{R}^h \middle| \boldsymbol{p}_u \leq \boldsymbol{p} \leq \boldsymbol{p}_o \right\}. \tag{5.7}$$

Das Ergebnis der Optimierung ist eine Pareto-Front mit optimalen Kompromissen zwischen maximalem Erwartungswert μ_f und geringster Streuung σ_f. Es ergeben sich die Mengen (4.13) nicht dominierter $\mathcal{N}$ und dominierter Entwürfe $\mathcal{D}$, die dem Prozess in Abb. 4.8 zur Auswahl neuer Stützstellen zugeführt

werden. Die neu ausgewählten Stützstellen werden dann direkt nachgerechnet, wobei wieder für jede Stützstelle eine Robustheitsanalyse durchgeführt wird, um Erwartungswert und Standardabweichung abzuschätzen. Analog zur ARSM-Strategie wird die Approximationsqualität geprüft und der Prozess wiederholt, bis eine maximale Anzahl von Iterationen erreicht wird oder die Approximationsqualität stagniert.

5.2.2 Darstellung der Ergebnisse

Zunächst wird das Problem (5.2) mit der in Abb. 5.1b dargestellten integrierten RDO Prozedur direkt, also ohne effizienzsteigernde Maßnahmen, gelöst. Für jeden Entwurfspunkt $\boldsymbol{p}_j$ werden jeweils 20 Stichproben der stochastischen Variablen $\boldsymbol{r}$ berechnet, um die Robustheitsanalyse durchzuführen. Der SPEA2 konvergiert nach 2010 Entwurfsauswertungen $\boldsymbol{p}_j$ mit insgesamt $2010 \times 20 = 40200$ Funktionsauswertungen. Das Ergebnis ist in Abb. 5.3a dargestellt, wobei die durch weiße Kreise hervorgehobenen Punkte die gefundenen Pareto-optimalen Entwürfe sind.

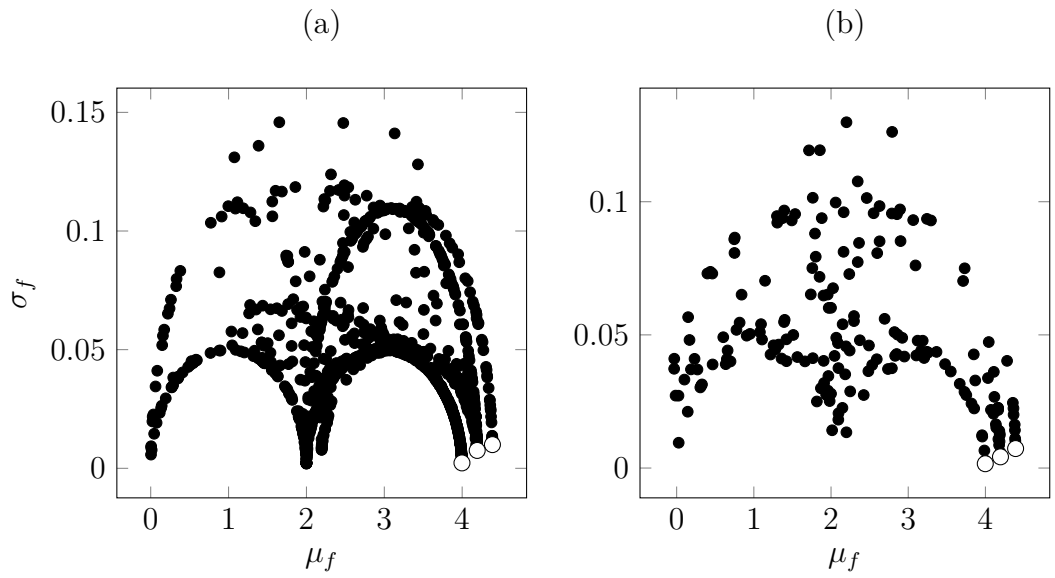

Abbildung 5.3: Berechnete Entwürfe im Kriterienraum der direkten Lösung (a) und der ARSM-Strategie (b)

Es ist deutlich zu sehen, dass es drei nicht dominierte Pareto-optimale Entwürfe gibt. Der Entwurf mit dem besten Erwartungswert (oberer rechter weißer Kreis) hat allerdings auch die größte Standardabweichung und kann in Abb. 5.2 als globaler Maximierer $\boldsymbol{p} = \begin{bmatrix} 3\pi & 3\pi \end{bmatrix}^T$ mit der größten Steigungsänderung unter allen lokalen Maxima identifiziert werden. Im Gegenzug kann der Entwurf mit der besten Standardabweichung (unterer linker weißer Kreis) dem lokalen Maximierer $\boldsymbol{p} = \mathbf{0}$ mit der geringsten Steigungsänderung in der unmittelbaren Umgebung zugewiesen werden. Die mittlere Lösung entspricht einem der beiden aus Sicht der robusten Optimierung gleichwertigen lokalen Maximierer $\boldsymbol{p} = \begin{bmatrix} 3\pi & 0 \end{bmatrix}^T$ oder $\boldsymbol{p} = \begin{bmatrix} 0 & 3\pi \end{bmatrix}^T$ in Abb. 5.2.

Das gleiche Problem wird nun mit dem auf der ARSM-Strategie basierenden RDO Prozess gelöst. Die Optimierung benötigt hier nur 180 Entwürfe $\boldsymbol{p}_j$ als Stützstellen für die Antwortflächen und damit nur $180 \times 20 = 3600$ direkte Funktionsauswertungen, was einer Effizienzsteigerung von 91 % entspricht. Die direkt berechneten Entwürfe sind in Abb. 5.3b dargestellt. Offensichtlich wurden die gleichen wichtigen Regionen der Maxima des Entwurfsraums aufgedeckt, wobei die Pareto-optimalen Entwürfe wieder als weiße Kreise dargestellt sind. Die uninteressanten Gebiete des Entwurfsraums sind im Vergleich zu Abb. 5.3a deutlich dünner besetzt. Dies liegt an der in Abschnitt 4.3 vorgestellten Strategie zur Auswahl neuer Stützstellen, da vornehmlich Entwürfe nahe der vermuteten bzw. echten Pareto-Front berücksichtigt und Entwürfe weniger wichtiger Regionen seltener zur Menge der Stützstellen hinzugefügt werden.

Das Konzept der adaptiven dualen Antwortflächen bietet also die Möglichkeit, den immensen Aufwand einer integrierten RDO zu verringern, so dass auch komplexere MKS-Modelle mit vertretbarem Aufwand robust optimierbar sind, siehe Ubben et al. [87]. Durch den Effizienzgewinn sollten auch robuste Architekturprobleme in Verbindung mit den bereits aufgezeigten Konzepten der Architekturoptimierung lösbar sein. Ob sich diese Annahme bewahrheitet, wird im folgenden Abschnitt untersucht, in dem das bereits bekannte Architekturproblem um unsichere Variablen ergänzt und damit als robustes Architekturproblem betrachtet wird.

5.3 Formulierung eines robusten Architekturproblems

Das in Abschnitt 4.1 vorgestellte Architekturproblem mit drei Segmenten $s = 1, 2, 3$ und Federungssystemen nach Abb. 2.2 wird für die Formulierung des robusten Architekturproblems übernommen. Die Einteilung in Architektur- (4.1) und Segmentvariablen (4.2) sowie die zugehörigen Entwurfsräume (4.4) mit den Werten in Tabelle 4.1 bleiben erhalten. Da nun das in Abb. 3.7b dargestellte Konzept mit robusten Segmentnachoptimierungen betrachtet wird, werden die Derivatvariablen (4.3) als streuende bzw. unsichere Variablen $\Delta \boldsymbol{p}^{(s,0)}$ betrachtet. Von den in Abschnitt 3.3 diskutierten Interpretationen einer robusten Architekturoptimierung wird hier die der noch unbekannten Derivate gewählt. Es sind also noch nicht alle Derivate bzw. deren Eigenschaften bekannt oder es ist unsicher, ob einige dieser überhaupt gebaut werden sollen.

Wenn künftige Verkaufszahlen einzelner Derivate oder deren Eigenschaften abgeschätzt werden können, kann jede Derivatvariable mit einer entsprechenden Verteilungsfunktion versehen werden. Mit der Abschätzung der Eigenschaften ist gemeint, dass beispielsweise drei von vier Derivaten eine ähnliche Aufbaumasse, das vierte aber eine stark davon abweichende aufweist. In diesem Fall könnte eine Verteilungsfunktion gewählt werden, die diese Gegebenheit in Form von Auftrittswahrscheinlichkeiten der unterschiedlichen Massen berücksichtigt. Für die folgende Analyse ist kein entsprechendes Vorwissen vorhanden, weshalb die Derivatvariablen als stetig gleichverteilt $\Delta \boldsymbol{p}^{(s,0)} = \mathcal{U}\left(\boldsymbol{p}_u^{(s,0)}, \boldsymbol{p}_o^{(s,0)}\right)$ angenommen werden. Die unteren und oberen Grenzen der Derivatvariablen $\boldsymbol{p}_u^{(s,0)}$ und $\boldsymbol{p}_o^{(s,0)}$ können, ergänzend zu den bisherigen Parameterwerten und -bereichen in Tabelle 4.1, der Tabelle 5.1 entnommen werden.

Ziel der Optimierung ist es nun, die bereits diskutierten Zielwerte zu erreichen und die daraus abgeleiteten Gütekriterien (4.5) zu minimieren. Im Kontext einer robusten Optimierung sind möglichst geringe Erwartungswerte und Streuungen zu erreichen. Für jedes Segment s und jede Stichprobe werden die Kriterien (4.5) zunächst quadriert und auf Basis einer Stichprobe nach den Erläuterungen in Abschnitt 5.2 jeweils Erwartungswert und Standardabweichung berechnet.

	Parameter	Segmente $s=1$		$s=2$		$s=3$	
$\left[\boldsymbol{p}_u^{(s,0)} \quad \boldsymbol{p}_o^{(s,0)}\right]$	m_A[kg]	[225	275]	[475	525]	[325	375]
	m_R[kg]	[25	35]	[45	55]	[35	45]
	c_R[N mm^{-1}]	[190	210]	[150	170]	[170	190]
$\left[\boldsymbol{p}_u^{(s)} \quad \boldsymbol{p}_o^{(s)}\right]$	c_F[N mm^{-1}]	[14	18]	[30	34]	[21	25]
	d_F[N s mm^{-1}]	[1,2	1,8]	[1,2	1,8]	[1,2	1,8]
	c_{F2}[N mm^{-1}]	-		-		[100	1000]
	d_{F2}[N s mm^{-1}]	-		-		[0,1	3]
$\left[\boldsymbol{p}_u^{(A)} \quad \boldsymbol{p}_o^{(A)}\right]$	m_F[kg]	[20	50]				
	c_A[N mm^{-1}]	[800	2000]				
	d_A[N s mm^{-1}]	[0,1	3]				

Tabelle 5.1: Parameterwerte des robusten Testproblems in Abb. 2.2

Anschließend werden diese kombiniert und auf Werte der Referenzlösung normiert, um das bikriterielle Kriterium (5.3) für jedes Segment s zu erhalten:

$$\mu^{(s)} = \sum_{n=1}^{N} \frac{\mu_n^{(s)}}{\mu_{n,ref}^{(s)}}, \quad \sigma^{(s)} = \sum_{n=1}^{N} \frac{\sigma_n^{(s)}}{\sigma_{n,ref}^{(s)}}. \tag{5.8}$$

Die Referenzgrößen werden hierbei für den jeweiligen Referenzentwurf mit den Variablenwerten

$$\boldsymbol{p}_{ref}^{(A)} = \frac{\boldsymbol{p}_o^{(A)} + \boldsymbol{p}_u^{(A)}}{2}, \quad \boldsymbol{p}_{ref}^{(s)} = \frac{\boldsymbol{p}_o^{(s)} + \boldsymbol{p}_u^{(s)}}{2} \tag{5.9}$$

und einer Stichprobengröße von 3000 bestimmt.

Die nachfolgenden Untersuchungen werden mit dem in Abschnitt 3.3 vorgestellten Konzept der robusten Architekturoptimierung aus Abb. 3.7b durchgeführt. Das robuste Segmentoptimierungsproblem (3.5) wird mit den Kriterien (5.8) entsprechend (5.3) segmentweise in die bikriteriellen Probleme

$$\min_{\boldsymbol{p}^{(s)} \in P^{(s)}} \boldsymbol{f}_{RDO}^{(s,0)} \left(\boldsymbol{p}^{(s)}; \boldsymbol{p}^{(A)}\right) \text{ mit } P^{(s)} = \left\{\boldsymbol{p}^{(s)} \in \mathbb{R}^{h^{(s)}} \middle| \boldsymbol{p}_u^{(s)} \leq \boldsymbol{p}^{(s)} \leq \boldsymbol{p}_o^{(s)}\right\} \tag{5.10}$$

überführt. Die Bewertung der sich ergebenden Pareto-Fronten erfolgt mithilfe der S-Metrik $\mathcal{S}_{RDO}^{(s)}$ analog zu (4.30). Nach (4.21) werden diese zu einem

Architekturkriterium $f_{\mathcal{S},RDO}^{(A)}$ kombiniert, das in einem äußeren Kreislauf als Architekturoptimierungsproblem (3.4) maximiert wird.

5.3.1 Direkte Lösung des robusten Architekturproblems

Wie bereits in Abschnitt 4.2.2 wird zunächst eine direkte Optimierung ohne effizienzverbessernde Maßnahmen durchgeführt. Es wird wieder der SPEA2 zur Lösung des Architekturproblems (3.4), (5.10) eingesetzt, wobei für Architektur- und Segmentoptimierungen maximal 90 Generationen mit jeweils 100 und eine Startpopulation von 1000 Individuen generiert werden. Nach 10 Generationen ohne Verbesserung der Kriterien werden die Algorithmen sowohl im inneren als auch im äußeren Kreislauf als konvergiert angenommen. Die mehrkriteriellen Segmentoptimierungen haben ein größeres Archiv mit einer Kapazität von 20 Individuen, um den Fokus der Suche auf den Bereich der Pareto-optimalen Entwürfe zu lenken. Für jeden direkt berechneten Entwurf werden 50 Stichproben ausgewertet, um Erwartungswert und Standardabweichung zu schätzen.

Die Lösung des robusten Architekturoptimierungsproblems mit der direkten kaskadierten gekoppelten RDO benötigt insgesamt $3{,}31 \cdot 10^9$ Funktionsauswertungen. Die Ergebnisse der einzelnen Segmente s sind in Abb. 5.4 dargestellt. Wie bei den anderen Ergebnisdarstellungen von Architekturoptimierungsproblemen sind in der unteren rechten Ecke die Werte der normierten Architekturvariablen $\boldsymbol{p}^{(A)}$, die Hypervolumina der Pareto-Fronten $\mathcal{S}_{RDO}^{(s)}$ und der Gütefunktionswert $f_{\mathcal{S},RDO}^{(A)}$ des besten robusten Architekturentwurfs aufgeführt. Bei der Betrachtung der einzelnen Fronten fällt auf, dass für Segment $s = 1$ nur vier Pareto-optimale Entwürfe gefunden werden und sich die Kriterienwerte dieser im Vergleich zu den Pareto-optimalen Entwürfen der anderen Segmente nur sehr geringfügig unterscheiden. Dies deutet darauf hin, dass die Kriterien, zumindest für den gefundenen Architekturentwurf, nur geringfügig kompromissbehaftet sind und es vielmehr ein einzelnes Optimum im Gegensatz zu einer ausgeprägten Pareto-Front gibt. Die Pareto-Fronten der anderen Segmente sind dichter besetzt und weisen eine größere Ausbreitung im Kriterienraum auf. Im Vergleich zur deterministischen Lösung in Abb. 4.7 hat sich der Wert der Architekturvariablen $m_F^{(A)}$ geändert. Oftmals ist die deterministische Lösung optimal in Bezug auf die Leistungsfähigkeit des Systems oder der Systeme, weisen aber eine starke

Streuung auf, sobald Systemunsicherheiten auftreten. Die gefundene robuste Lösung stellt einen Entwurf dar, der die besten Kompromisse geringer Streuung und hoher mittlerer Leistungsfähigkeit der Systeme bereitstellt. Typischerweise sind diese beiden Lösungen, wie auch beim vorliegenden Fall, nicht identisch.

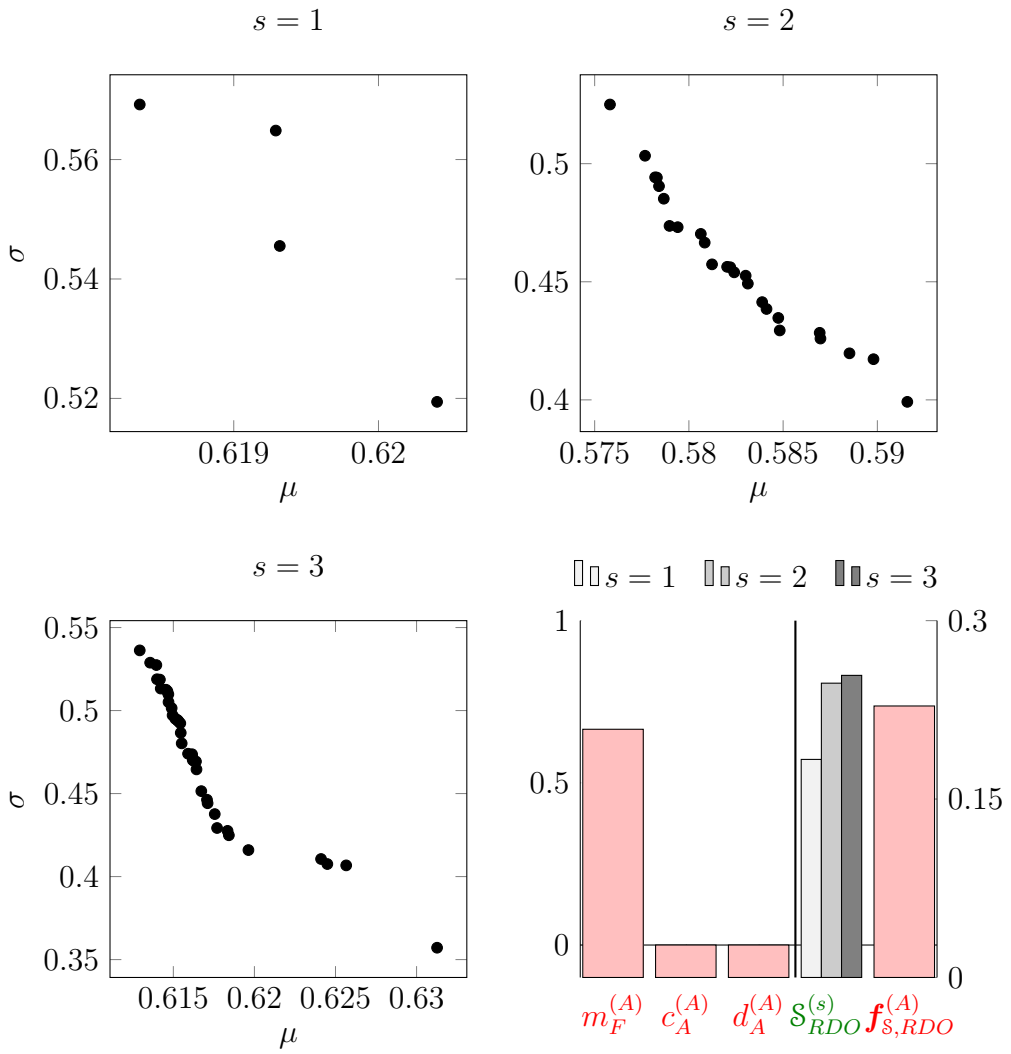

Abbildung 5.4: Optimale Ergebnisse der kaskadierten gekoppelten RDO

5.3.2 Lösung des robusten Architekturproblems mit Subraumoptimierungen

Für die effiziente Lösung des robusten Architekturproblems wird nun die in Abschnitt 4.3.2 vorgestellte Strategie mit Subraumoptimierungen eingesetzt. Wie in Abschnitt 4.3.3 diskutiert, ist die CSSO-Strategie zu empfehlen, wenn

wenig oder kein Vorwissen über das zu optimierende Problem vorhanden ist, wovon bei dieser Analyse ausgegangen wird.

Die robuste CSSO-Strategie läuft nach dem gleichen Prinzip wie die deterministische in Abb. 4.12 und Abb. 4.13 ab. Die Unterschiede liegen darin, dass für jede Stützstelle $\boldsymbol{q}_j^{(s)}$ der nach (4.26) definierten Stützstellenmenge $I_i^{(s)}$ eine Stichprobe für die unsicheren Variablen $\Delta\boldsymbol{p}^{(s,0)}$ durchgeführt wird, um die Kriterien (5.8) abzuschätzen. Die wird auch für alle in jeder Iteration i neu bestimmten Stützstellen durchgeführt. Wie bereits im Rahmen der robusten Optimierung der Testfunktion in Abschnitt 5.2.1 gezeigt, werden die Antwortflächen auf Basis dieser Kriterien aufgebaut:

$$\hat{\boldsymbol{f}}_{RDO}^{(s,0)}\left(\boldsymbol{q}^{(s)}\right) = \boldsymbol{f}_{RDO}^{(s,0)}\left(\boldsymbol{q}^{(s)}\right) + \boldsymbol{\varepsilon}^{(s)}. \tag{5.11}$$

Hierbei entspricht $\boldsymbol{f}_{RDO}^{(s,0)}$ dem Vektorkriterium (5.3) mit Erwartungswert $\mu^{(s)}$ und Standardabweichung $\sigma^{(s)}$ nach (5.8).

Die Optimierungen von Architektur und Segmenten sowie die Subraumoptimierungen laufen analog zur deterministischen CSSO-Strategie auf diesen Antwortflächen ab. Für die robusten Subraumoptimierungen ergeben sich die Probleme

$$\min_{\boldsymbol{q}^{(s)}\in Q^{(s)}} \hat{\boldsymbol{f}}_{\mathrm{S},RDO}^{(A,s)}\left(\boldsymbol{q}^{(s)}\right) \text{ mit } Q^{(s)} = \left\{\boldsymbol{q}^{(s)} \in \mathbb{R}^{h^{(A)}+h^{(s)}} \middle| \boldsymbol{q}_u^{(s)} \leq \boldsymbol{q}^{(s)} \leq \boldsymbol{q}_o^{(s)}\right\} \tag{5.12}$$

mit

$$\hat{\boldsymbol{f}}_{\mathrm{S},RDO}^{(A,s)} = \begin{bmatrix} \hat{\boldsymbol{f}}_{RDO}^{(s,0)} \\ \hat{\boldsymbol{S}}_{RDO}^{(s)} \end{bmatrix}. \tag{5.13}$$

Der Vektor der S-Metriken $\hat{\boldsymbol{S}}_{RDO}^{(s)}$ wird aus den Pareto-Fronten der Optimierungen

$$\min_{\boldsymbol{p}^{(\tau)}\in P^{(\tau)}} \hat{\boldsymbol{f}}_{RDO}^{(\tau,0)}\left(\boldsymbol{p}^{(\tau)};\boldsymbol{p}^{(A,s)}\right) \text{ mit } P^{(\tau)} = \left\{\boldsymbol{p}^{(\tau)} \in \mathbb{R}^{h^{(\tau)}} \middle| \boldsymbol{p}_u^{(\tau)} \leq \boldsymbol{p}^{(\tau)} \leq \boldsymbol{p}_o^{(\tau)}\right\} \tag{5.14}$$

gewonnen. Anschließend werden die aus den Segmentoptimierungen

$$\min_{\boldsymbol{p}^{(s)}\in P^{(s)}} \hat{\boldsymbol{f}}_{RDO}^{(s,0)}\left(\boldsymbol{p}^{(s)};\boldsymbol{p}^{(A)}\right) \text{ mit } P^{(s)} = \left\{\boldsymbol{p}^{(s)} \in \mathbb{R}^{h^{(s)}} \middle| \boldsymbol{p}_u^{(s)} \leq \boldsymbol{p}^{(s)} \leq \boldsymbol{p}_o^{(s)}\right\} \tag{5.15}$$

resultierenden S-Metriken $\hat{\mathbb{S}}_{RDO}^{(s)}$ kombiniert nach (4.21) mit

$$\max_{\boldsymbol{p}^{(A)} \in P^{(A)}} \hat{f}_{\mathbb{S},RDO}^{(A)} \left(\boldsymbol{p}^{(A)} \right) \text{ mit } P^{(A)} = \left\{ \boldsymbol{p}^{(A)} \in \mathbb{R}^{h^{(A)}} \middle| \boldsymbol{p}_u^{(A)} \leq \boldsymbol{p}^{(A)} \leq \boldsymbol{p}_o^{(A)} \right\} \quad (5.16)$$

maximiert.

Die Algorithmen und Fehlertoleranzen werden mit den gleichen Einstellungen wie in Abschnitt 5.3.1 und Abschnitt 4.3.2 konfiguriert. Das in Abb. 5.5 dargestellte Ergebnis wird nach insgesamt 138 350 Funktionsauswertungen erreicht. Die weißen Punkte stellen hierbei tatsächliche Pareto-optimale Entwürfe dar. Die schwarzen Punkte wurden ebenfalls auf Basis der Optimierungen auf den Antwortflächen gewählt, haben sich nach der direkten Berechnung aber als nicht Pareto-optimal herausgestellt. Die Werte der Entwurfsvariablen gleichen sich, nur $m_F^{(A)}$ liegt etwas höher, als es bei der direkten Lösung in Abb. 5.4 der Fall ist. Einer Effizienzsteigerung um einen Faktor von etwa $2{,}4 \cdot 10^4$ steht ein durch kleinere Hypervolumina charakterisierbarer Qualitätsverlust von ca. 14,8 % gegenüber.

5.4 Diskussion der Ergebnisse

In diesem Kapitel wurde die CSSO-Strategie zur Lösung eines robusten Architekturoptimierungsproblems unter dem Fokus der Effizienzsteigerung analysiert. Wie bereits in den vorangegangenen Analysen wurde ein um unsichere Variablen erweitertes Architekturoptimierungsproblem betrachtet, das aus einfachen Viertelfahrzeugmodellen besteht. Zunächst ist festzuhalten, dass die untersuchte Strategie bei deutlich gesteigerter Effizienz etwas schlechter abschneidet als eine direkt durchgeführte robuste Architekturoptimierung.

Zur Analyse beider Ergebnisse bedarf es eines direkten Vergleichs der gefundenen Pareto-Fronten. Diese sind in Abb. 5.6 vergleichend darfestellt. Es ist deutlich zu sehen, dass die Pareto-Fronten der CSSO-Strategie nicht die Qualität der Fronten der direkten Lösung erreichen. Im ersten Segment werden die gefundenen Entwürfe von denen der direkten Strategie dominiert. Das gilt auch für das zweite Segment, wobei die Front der CSSO-Strategie zusätzlich deutlich stärker auf eine Region konzentriert ist. Selbiges gilt für das dritte Segment;

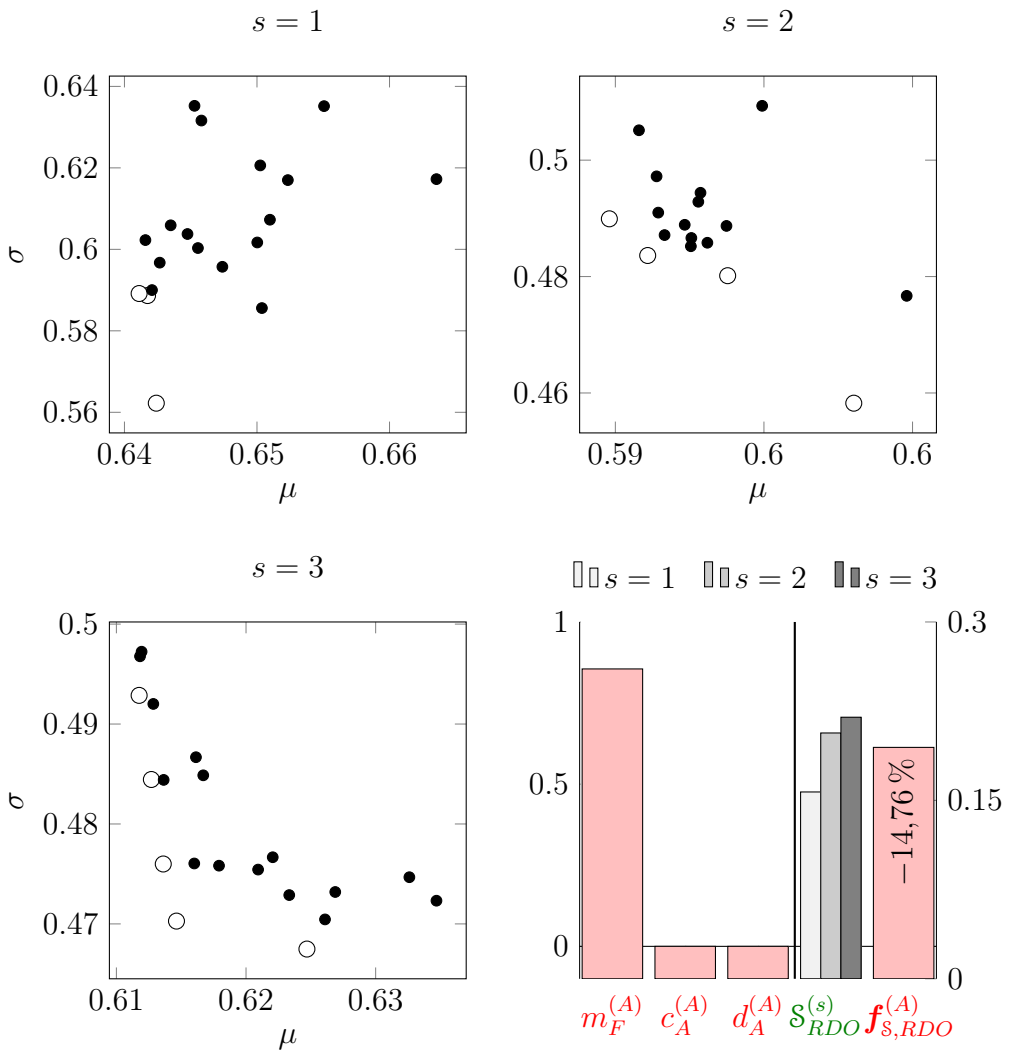

Abbildung 5.5: Optimale Ergebnisse der Subraumoptimierung-basierten RDO

hier liegen die Entwürfe beider Strategien zwar auf dem gleichen Niveau, allerdings deckt die Front der direkten Lösung einen deutlich größeren Bereich ab.

Nun stellt sich die Frage, warum es zu diesem Nachteil der effizienteren CS-SO-Lösung kommt. Ein wichtiger Aspekt ist die Approximationsqualität der Antwortflächen. Da jede Stützstelle $\boldsymbol{q}_j$, auf denen die Antwortflächen basieren, mit einer relativ geringen Stichprobengröße von 50 bewertet wird, kann ein und dieselbe Stützstelle unterschiedliche Erwartungswerte und Standardabweichungen aufweisen. Für eine aussagekräftigere Schätzung ist eine deutlich größere Stichprobenmenge notwendig, die allerdings die Effizienz negativ beeinflusst.

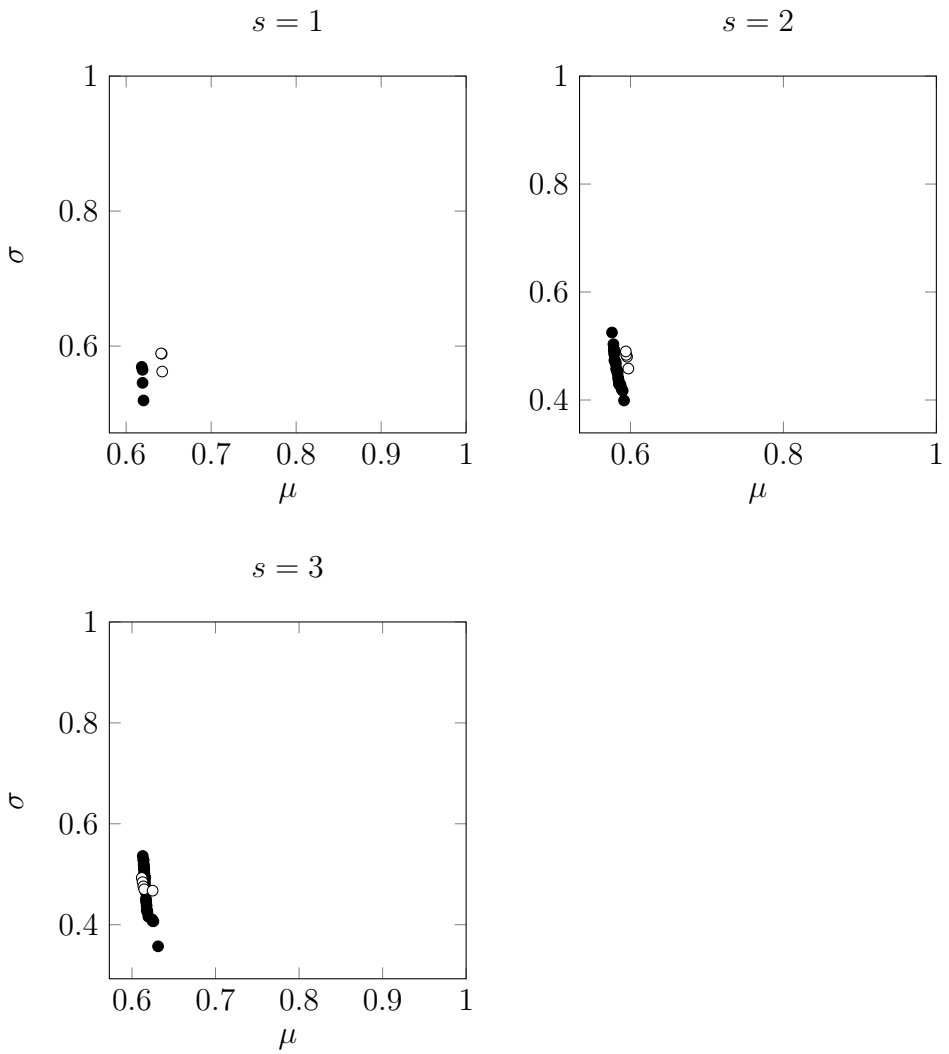

Abbildung 5.6: Pareto-Fronten der direkten (schwarz) und CSSO-Strategie (weiß)

Ein möglicher Lösungsansatz ist die Verwendung von lokalen Antwortflächen zur Abschätzung der Robustheitsmaße, wie sie von Busch und Bestle [15] vorgeschlagen wurde, siehe Abschnitt 5.2.

Ein weiterer Punkt ist die Anzahl der gefundenen Pareto-optimalen Entwürfe. Bei den vorangegangenen deterministischen Optimierungen wurde derselbe Architekturentwurf in mehreren Iterationen als optimale Lösung gefunden und somit eine größere Zahl von Stützstellen für diesen Entwurf berechnet. Bei der robusten Optimierung wurde der gefundene optimale Architekturentwurf nur

in einer Iteration erreicht. Um ein besseres Ergebnis zu erzielen, sollte eine geringere Fehlertoleranz und eine höhere Anzahl zulässiger Iterationen gewählt werden, was wiederum die Effizienz negativ beeinflusst.

Dennoch muss festgehalten werden, dass die CSSO-Strategie Lösungen liefert, die deutlich besser als die Referenz sind. Im Prinzip sind also vorrangig die vorhandenen Ressourcen dafür bestimmend, ob eine weniger genaue aber effizientere Strategie eingesetzt wird. Letztendlich können die gefundenen Ergebnisse auch verwendet und analysiert werden, um anschließend eine weitere, gezieltere Optimierung durchzuführen, um ein noch besseres Ergebnis zu erzielen.

Kapitel 6

Anwendung auf ein Gesamtfahrzeugmodell

Nachdem verschiedene Verfahren und Strategien zur Optimierung und robusten Optimierung von Fahrwerksarchitekturen anhand einfacher Testprobleme diskutiert und deren Funktionalität aufgezeigt wurden, folgt nun die Anwendung einer ausgewählten Strategie auf die Auslegung einer Fahrwerksarchitektur eines Gesamtfahrzeugs. Ob die Anwendbarkeit der Architekturoptimierung mit effizienzsteigernden Maßnahmen bei der zeitintensiven Lösung von MKS-Gesamtfahrzeugmodellen sinnvoll gegeben ist, wird im Folgenden analysiert.

6.1 Formulierung der Optimierungsaufgabe

Die zu optimierende Architektur ist flexibel, wie sie in Abschnitt 1.1 beschrieben und in Abb. 1.3 beispielhaft dargestellt ist. Für das Anwendungsbeispiel wird von einer Produktfamilie von mehreren Fahrzeugmodellen ausgegangen, die auf einer gemeinsamen Architektur basieren. Diese Fahrzeugmodelle können in zwei Segmente unterteilt werden, die unterschiedliche Radstände und Spurweiten aufweisen. Innerhalb dieser Segmente kann es verschiedene Derivate geben. Für das Anwendungsbeispiel wird jeweils ein Auslegungsderivat pro Segment untersucht. Zusammengefasst gibt es eine Architektur A mit zwei Segmenten $S = 2$ und jeweils einem Null-Derivat $d = 0$ pro Segment.

Die Fahrzeuge werden durch das in Abschnitt 2.4 beschriebene MKS-Modell

abgebildet und mit den dort beschriebenen Prinzipien berechnet. Für das Anwendungsbeispiel kommt das in Abb. 3.7a dargestellte Konzept der Architekturoptimierung zur Anwendung. Hierbei gilt es, das Architekturoptimierungsproblem nach (3.1) mit

$$\max_{\boldsymbol{p}^{(A)}\in P^{(A)}} f_{\mathcal{S},MKS}^{(A)}\left(\boldsymbol{p}^{(A)}\right) \text{ mit } P^{(A)} = \left\{\boldsymbol{p}^{(A)} \in \mathbb{R}^{h^{(A)}} \middle| \boldsymbol{p}_u^{(A)} \leq \boldsymbol{p}^{(A)} \leq \boldsymbol{p}_o^{(A)}\right\} \quad (6.1)$$

sowie die mehrkriteriellen Segmentoptimierungsprobleme nach (3.2) mit

$$\min_{\boldsymbol{p}^{(s)}\in P^{(s)}} \boldsymbol{f}_{MKS}^{(s,0)}\left(\boldsymbol{p}^{(s)};\boldsymbol{p}^{(A)},\boldsymbol{p}^{(s,0)}\right) \text{ mit } P^{(s)} = \left\{\boldsymbol{p}^{(s)} \in \mathbb{R}^{h^{(s)}} \middle| \boldsymbol{p}_u^{(s)} \leq \boldsymbol{p}^{(s)} \leq \boldsymbol{p}_o^{(s)}\right\} \quad (6.2)$$

zu lösen. Das Architekturkriterium $f_{\mathcal{S},MKS}^{(A)}$ wird nach (4.21) aus den S-Metriken $\mathcal{S}_{MKS}^{(s)}$ der Mengen Pareto-optimaler Entwürfe (4.19) der Segmente gebildet. Es folgt eine genauere Erläuterung des Kriterienvektors $\boldsymbol{f}_{MKS}^{(s,0)}$ sowie der verschiedenen Typen von Entwurfsvariablen $\boldsymbol{p}^{(A)}$ und $\boldsymbol{p}^{(s)}$.

6.1.1 Auswahl von Kriterien

Die Ziele der folgenden Optimierung sind die Verbesserung der Bremsstabilität und Agilität sowie die Reduzierung der Zahnstangenkräfte bei möglichst gleichbleibendem Spurdifferenzwinkel und maximalem Lenkwinkel. Die verschiedenen Größen können durch geeignete Lastfälle in der Simulation bestimmt und Kriterien für die Optimierung abgeleitet werden. Die verschiedenen Lastfälle werden im Folgenden erläutert und sind schematisch in Abb. 6.1 dargestellt. Hierbei gibt es ein inertiales Koordinatensystem $\{O_I; x_I, y_I, z_I\}$ und im Schwerpunkt S_F des Fahrzeugs liegt ein körperfestes Koordinatensystem $\{C_1; x_1, y_1, z_1\}$. Die x_1-Achse verläuft entgegen der Fahrtrichtung, die y_1-Achse steht parallel zur Fahrbahn und senkrecht zu x_1 und die z_1-Achse verläuft entsprechend eines rechtshändigen Systems nach oben. Die Radlenk- bzw. Spurwinkel werden durch δ_l und δ_r beschrieben, wobei für die folgenden Erläuterungen abkürzend ein mittlerer Spurwinkel $\delta = \frac{\delta_r+\delta_l}{2}$ verwendet wird. Der Gierwinkel Ψ beschreibt die Drehung des Fahrzeugs um die z_I-Achse und ist eine wichtige Größe zur Bewertung der Bremsstabilität, die nun genauer erläutert wird.

Als Bremsstabilität wird die Spurtreue eines Fahrzeugs unter Verzögerung bezeichnet. Eine weitere gängige Bezeichnung für dieses Bewertungskriterium

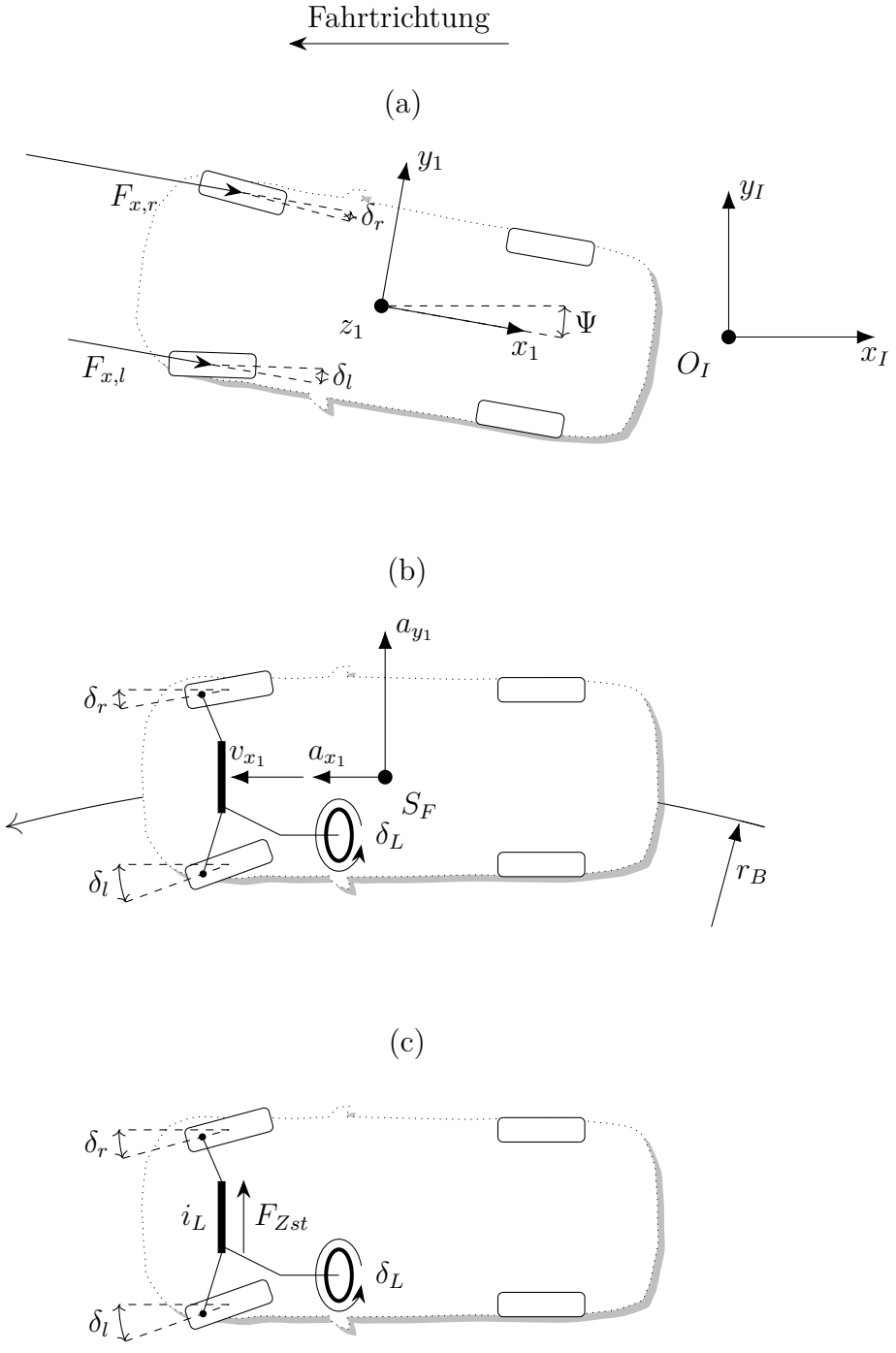

Abbildung 6.1: Schematische Darstellung der Lastfälle (a) Bremsverhalten, (b) stationäre Kreisfahrt und (c) Parkieren

ist das Bremsabziehen. Prinzipiell führen Asymmetrien im Fahrzeug bei starker Verzögerung zu einer Gierbewegung, die möglichst gering ausfallen sollte. Typischerweise wird das Verhalten bei einer Bremsung analysiert, bei der es noch zu keinem ABS-Eingriff kommt, um das Grundfahrwerk und deren Neigung zum Bremsabziehen bewerten zu können. Die Asymmetrien können hierbei aus der Gewichtsverteilung und aus Reibwert- oder Bremsmomentunterschieden entstehen. Für die Bremsstabilität ist es vorteilhaft, wenn das aufgrund der Gierbewegung stärker belastete kurvenäußere Rad der Vorderachse in Nachspur geht, um eine der vorherrschenden Gierbewegung entgegengesetzte Reaktion herbeizuführen [38]. Für die Bewertung der Bremsstabilität in der Simulation werden dem Fahrzeug am linken und rechten Vorderreifen unterschiedlich starke Bremskräfte $F_{x,r} > F_{x,l}$ aufgeprägt, wie in Abb. 6.1a dargestellt. Aufgrund der unterschiedlichen Bremskräfte resultiert ein Moment um die negative z_1-Achse, wodurch das Fahrzeug die oben beschriebene Gierbewegung ausführt. Ab dem Zeitpunkt der Bremskrafteinleitung wird nach Verstreichen einer vordefinierten Zeit $t_{\dot{\Psi}}$ die dann resultierende Giergeschwindigkeit $\dot{\Psi}\left(t_{\dot{\Psi}}\right)$ als Kriterium zur Bewertung der Bremsstabilität betrachtet. Je niedriger der Betrag der Giergeschwindigkeit $|\dot{\Psi}\left(t_{\dot{\Psi}}\right)|$ ausfällt, umso stabiler verhält sich das Fahrzeug, wobei ein Wert von $\dot{\Psi}\left(t_{\dot{\Psi}}\right) = 0\,^\circ\,\mathrm{s}^{-1}$ das Optimum darstellt. Verläufe der Giergeschwindigkeit $\dot{\Psi}$ über die Zeit t des beschriebenen Manövers sind in Abb. 6.4 abgebildet.

Ein typisches Maß zur Bewertung der Agilität eines Fahrzeugs ist der Eigenlenkgradient EG. Er beschreibt die Lenkarbeit, die notwendig ist, um ein mit einer konstanten Geschwindigkeit v_{x_1} fahrendes Fahrzeug auf einer Bahn mit gleichbleibendem Bahnradius r_B zu halten, Abb. 6.1b. Um das Eigenlenkverhalten über einen größeren Querbeschleunigungsverlauf bewerten zu können, wird das Fahrzeug typischerweise langsam mit a_{x_1} beschleunigt. Hierbei wird dann von einer quasi stationären Kreisfahrt gesprochen. Der EG berechnet sich dann aus

$$EG = \frac{1}{i_L}\frac{\mathrm{d}\delta_L}{\mathrm{d}a_{y_1}} - \frac{\mathrm{d}\delta_A}{\mathrm{d}a_{y_1}}, \tag{6.3}$$

wobei i_L die Gesamtlenkübersetzung, δ_L der Lenkradwinkel, δ_A der Ackermannwinkel und a_{y_1} die Fahrzeugquerbeschleunigung sind. Der Ackermannwinkel δ_A beschreibt hierbei den Spurwinkel des Rads, der nötig ist, um die Kreisbahn bei $v_{x_1} < 1\,\mathrm{m\,s}^{-1}$ bzw. $a_{y_1} \approx 0\,\mathrm{m\,s}^{-2}$ zu durchfahren. Näherungsweise kann der

Ackermannwinkel $\delta_A \approx \frac{l_R}{r_B}$ als Quotient aus Radstand und Kreisbahn beschrieben werden. Wie oben beschrieben, wird eine konstante Kreisbahn durchfahren, die Änderung des Ackermannwinkels ist auf Basis der Näherung somit $\mathrm{d}\delta_A = 0$, wodurch sich (6.3) zu

$$EG = \frac{1}{i_L} \frac{\mathrm{d}\delta_L}{\mathrm{d}a_{y_1}} \tag{6.4}$$

vereinfacht. Für die hier durchgeführte Analyse wird der EG für den Querbeschleunigungsbereich $a_{y_1,1} < a_{y_1} < a_{y_1,4}$ mit $a_{y_1,1} = 1\,\mathrm{m\,s^{-2}}$ und $a_{y_1,4} = 4\,\mathrm{m\,s^{-2}}$ betrachtet. In diesem Bereich verhält sich der EG annähernd linear. Nach (6.4) wird der Eigenlenkgradient dann mit

$$EG_{1-4} = \frac{\delta_{EG}\left(a_{y_1,4}\right) - \delta_{EG}\left(a_{y_1,1}\right)}{a_{y_1,4} - a_{y_1,1}} \tag{6.5}$$

berechnet, wobei $\delta_{EG}\left(a_{y_1}\right) = \frac{\delta_L\left(a_{y_1}\right)}{i_L} - \delta_A$ der Eigenlenkbedarf ist. Dessen charakteristischer Verlauf ist in Abb. 6.4 bezogen auf $a_{y_1,1}$ als $\bar{\delta}_{EG}\left(a_{y_1}\right) = \delta_{EG}\left(a_{y_1}\right) - \delta_{EG}\left(a_{y_1,1}\right)$ dargestellt.

Der EG gibt nun Aufschluss über die Über- bzw. Untersteuertendenz eines Fahrzeugs. Ist $EG > 0$, verhält sich das Fahrzeug untersteuernd, der Fahrer muss also den Lenkwinkel vergrößern, um einer Kurvenbahn bei steigender Geschwindigkeit und damit steigender Querbeschleunigung a_{y_1} folgen zu können, wie es auch in Abb. 6.4 der Fall ist. Ist $EG < 0$, so verhält es sich genau umgekehrt und das Fahrzeug übersteuert. Aus Gründen der Fahrsicherheit haben Serienfahrzeuge zumeist einen positiven EG, da ein untersteuerndes Fahrzeug stets stabil ist. Generell steht die Höhe des EG dann für die Agilität eines Pkw: je höher er ist, umso weniger agil ist das Fahrzeug. Das Ziel für die zu optimierende Architektur ist eine möglichst hohe Agilität, also ein möglichst kleiner EG.

Das dritte Kriterium bezieht sich auf die maximal wirkende Kraft auf die Zahnstange F_{Zst} einer elektrisch unterstützten Lenkung. Die größten Kräfte treten beim Lastfall Parkieren auf, Abb. 6.1c. Hierbei steht das Fahrzeug und die Räder sind festgebremst. In diesem Zustand wird in beide Richtungen bis zu einem elektronisch begrenzten Lenkradwinkel $\delta_{L,lim}$ eingelenkt. Eine Pkw-Lenkung weist einen mechanischen Anschlag auf, um zu große Spurwinkel

zu verhindern, allerdings nimmt die Elektronik die Unterstützung vor dem Erreichen des Anschlags zum Schutz der Komponenten zurück. Für den Fahrer wird es ohne elektrische Unterstützung somit nahezu unmöglich, insbesondere im Stand, einen noch größeren Lenkwinkel zu erreichen. Die maximal wirkende Kraft F_{Zst} wird also nicht durch den mechanischen Anschlag, sondern durch die Reibeigenschaften der montierten Reifen und den beim zugehörigen Lenkradwinkel $\delta_{L,lim}$ erreichten mittleren Spurwinkel $\delta_{lim} = \delta_{L,lim}/i_L$ bestimmt. Neben dem Reifentyp spielt die Bewegung des Reifens beim Lenken eine Rolle. Je nach Auslegung der Kinematik wird er nicht nur um seine Hochachse gedreht, sondern führt eine Kombination aus Drehungen und Verschiebungen in mehrere Raumrichtungen durch. Zusätzlich gibt es verschiedene Hebelverhältnisse, die das Kraftniveau direkt beeinflussen.

In der für die Optimierung durchgeführten Simulation werden weder der Elektromotor noch ein mechanischer Anschlag modelliert. Der Elektromotor wird nicht benötigt, da in der Simulation beliebige Kräfte und Momente bereitgestellt werden können. Im Rahmen der Optimierung werden Kinematikpunkte des Fahrwerks verschoben, wodurch die Gesamtlenkübersetzung i_L geändert wird. Um bei allen sich aus der Kombinatorik der Entwurfsvariablen ergebenden Entwürfe den oben beschriebenen, durch die Unterstützung des Elektromotors begrenzten Spurwinkel δ_{lim} zu erreichen, ist es notwendig, über den mechanischen Anschlag der Lenkung hinaus zu lenken. Wäre dies nicht der Fall, so könnte es passieren, dass bei gleichbleibendem Lenkradwinkel $\delta_{L,lim}$ der Entwurf $\boldsymbol{p}'$ das Optimum darstellt, bei dem der resultierende mittlere Spurwinkel $\delta'_{lim} = \delta_{L,lim}/i'_L < \delta_{lim}$ am kleinsten ist. Um den Einfluss des Spurwinkels zu reduzieren, wird die wirkende Kraft F_{Zst} stets bei dem gleichen Spurwinkel δ_1 erfasst. Hierfür ist es dann ggf. notwendig, einen höheren Lenkradwinkel $\delta_{L,max} > \delta_{L,lim}$ anzusteuern, als ursprünglich für das Fahrzeug vorgesehen, damit jeder Entwurf den zur Auswertung gewählten Spurwinkel δ_1 erreicht. Verläufe der Zahnstangenkraft F_{Zst} über dem mittleren Radwinkel δ sind ebenfalls in Abb. 6.4 abgebildet.

Damit die Lenkcharakteristik des Ausgangsentwurfs der Fahrzeuge, nachfolgend als Referenz bezeichnet, erhalten bleibt, werden zusätzlich der Lenkfehler ε_δ bei δ_1 und der bei einem Lenkradwinkel $\delta_{L,max}$ auftretende maximale mittlere

Spurwinkel δ_{max} betrachtet. Der Lenkfehler ε_δ stellt hierbei die Differenz der Winkel vom linken zum rechten Rad $\varepsilon_\delta = \delta_l - \delta_r$ dar. Beide Größen sollen bei denen vom Optimierer vorgeschlagenen Entwürfen möglichst wenig vom ursprünglichen Referenzwert abweichen.

Zusammenfassend ergeben sich folgende, zu minimierende Kriterien $f_{1,MKS}$ bis $f_{4,MKS}$:

$$\begin{aligned} f_{1,MKS} &= \frac{EG_{1-4}}{EG_{1-4,ref}}, \quad f_{2,MKS} = \frac{\dot{\Psi}\left(t_{\dot{\Psi}}\right)}{\dot{\Psi}_{ref}\left(t_{\dot{\Psi}}\right)}, \quad f_{3,MKS} = \frac{F_{Zst}\left(\delta_1\right)}{F_{Zst,ref}\left(\delta_1\right)}, \\ f_{4,MKS} &= \sqrt{\frac{1}{2}\left(\left(\frac{\varepsilon_\delta - \varepsilon_{\delta,ref}}{\varepsilon_{\delta,ref}}\right)^2 + \left(\frac{\delta_{max} - \delta_{max,ref}}{\delta_{max,ref}}\right)^2\right)}. \end{aligned} \tag{6.6}$$

Alle Kriterien werden auf den zugehörigen Referenzwert normiert, wobei für $f_{4,MKS}$ die Bewertung der Abweichungen von den Referenzwerten wichtig ist und deshalb ein Root-Mean-Square Ansatz für Lenkfehler ε_δ und maximalem Lenkwinkel δ_{max} zum Einsatz kommt.

6.1.2 Definition von Entwurfsvariablen

Die hier durchgeführte Analyse beschränkt sich auf die Vorderachse der betrachteten Segmente und deren Derivate. Als Architektur wird hierbei die Lenkung gewählt. Genauer gesagt geht es um eine einheitliche Lenkungslage, also die Positionierung der Lenkung in x_1- und z_1-Richtung relativ zur Mitte der Vorderachse. Die Entwurfsvariablen der Architektur x_{Zst} und z_{Zst}, werden zu einem Entwurfsvektor zusammengefasst:

$$\boldsymbol{p}^{(A)} = \begin{bmatrix} x_{Zst} & z_{Zst} \end{bmatrix} \tag{6.7}$$

Für die Architektur gelten untere und obere Grenzen $\boldsymbol{p}_u^{(A)}$ und $\boldsymbol{p}_o^{(A)}$, womit der Entwurfsraum $P^{(A)}$ der Architektur nach (4.4) aufgespannt wird. Die Architekturvariablen sind schematisch in Abb. 6.2 dargestellt. Aus kinematischer Sicht ist der äußere Anlenkpunkt zwischen Lenkung und Spurstange wichtig, der in Abb. 6.2 als architekturrelevantes Element rot markiert ist. Die Lenkung stellt die Verbindung dieser Anlenkpunkte zwischen der rechten und linken Fahrzeugseite dar.

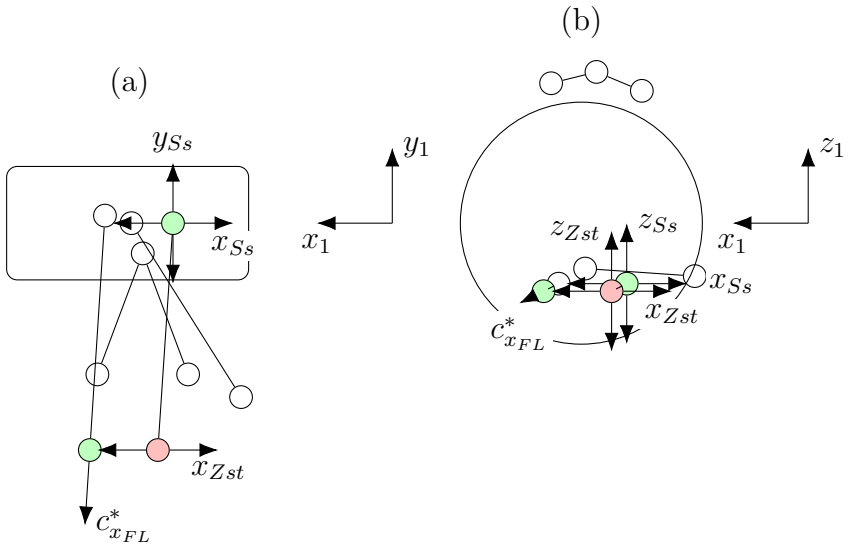

Abbildung 6.2: Ober- (a) und Seitenansicht (b) der untersuchten Vorderachse

Segmentspezifisch sind die äußeren Anlenkpunkte zwischen Spurstange und Radträger, die in allen Raumrichtungen relativ zur Mitte der Vorderachse durch die Variablen x_{Ss}, y_{Ss} und z_{Ss} beschrieben sind. Zusätzlich wird entsprechend der Erläuterungen aus Abschnitt 2.4.2 die Steifigkeit des Federlenkerlagers in Strebenrichtung $c^*_{x_{FL}}$ variiert. Es ergibt sich dann der Entwurfsvektor für die Segmentvariablen

$$\boldsymbol{p}^{(s)} = \begin{bmatrix} x_{Ss} & y_{Ss} & z_{Ss} & c^*_{x_{FL}} \end{bmatrix}. \tag{6.8}$$

Auch hier gelten segmentspezifische Grenzen $\boldsymbol{p}_u^{(s)}$ und $\boldsymbol{p}_o^{(s)}$ und nach (4.4) entsprechende Entwurfsräume $P^{(s)}$. In der Optimierung werden sämtliche Entwurfsvariablen jeweils normiert betrachtet.

6.2 Diskussion der Optimierungsergebnisse

Mit den oben beschriebenen Gütekriterien und Entwurfsvariablen werden nun die verschiedenen Optimierungsprobleme für Architektur (6.1) und Segmente (6.2) der kaskadierten Optimierung mithilfe der CSSO-Strategie gelöst. Der Kriterienvektor für die Segmente $\boldsymbol{f}_{MKS}^{(s,0)}$ lautet mit den Kriterien (6.6)

$$\boldsymbol{f}_{MKS}^{(s,0)} = \begin{bmatrix} f_{1,MKS} & f_{2,MKS} & f_{3,MKS} & f_{4,MKS} \end{bmatrix}. \tag{6.9}$$

Die Optimierung läuft dann nach dem Prozess in Abb. 4.12 ab. Zu Beginn werden für jedes Segment $J_0^{(s)} = 30$ initiale Stützstellen berechnet, mit denen

die im weiteren Verlauf der Optimierung verwendeten Antwortflächen trainiert werden. In den Subraumoptimierungen, siehe Abb. 4.13, werden maximal 5 Iterationen durchgeführt, wobei jeweils 5 neue Stützstellen generiert werden. Die Fehlertoleranz, ab der die Antwortflächenqualität als ausreichend genug angenommen wird, beträgt $\varepsilon_{tol}^{SSO} = 1\,\%$. Wird nach 3 Iterationen keine weitere Verbesserung der Antwortflächenqualität erreicht oder die Fehlertoleranz wiederholt unterschritten, enden die Subraumoptimierungen.

Der äußere Kreislauf mit der kaskadierten Architekturoptimierung wird nach 10 Iterationen ohne Verbesserung der Antwortflächen oder wiederholtem Erreichen der Fehlertoleranz als konvergiert angenommen. Die Fehlertoleranz wird mit $\varepsilon_{tol} = 0{,}5\,\%$ strenger gewählt, da die hier gefundenen Entwürfe der zu suchenden Architektur entsprechen und somit eine höhere Antwortflächenqualität anzustreben ist. Maximal werden 50 äußere Iterationen berechnet. Die Optimierungen auf den Antwortflächen von Architektur, Segmenten und Subräumen nutzen eine Startpopulation von 1000 Individuen und berechnen maximal 90 Generationen mit jeweils 100 Individuen. Die Archive der mehrkriteriellen Segmentoptimierungen und Subraumoptimierungen behalten die 20 besten Individuen, die bis zum aktuellen Zeitpunkt der Optimierung gefunden wurden. Voruntersuchungen haben gezeigt, dass die Referenzlösung, die nach (6.6) und (6.9) auf $\boldsymbol{f} = \begin{bmatrix}1 & 1 & 1 & 0\end{bmatrix}^T$ führt, bereits sehr gut ist, weshalb der Referenzpunkt zur Berechnung der Hypervolumina auf $\boldsymbol{f}_{ref} = \begin{bmatrix}2 & 2 & 2 & 2\end{bmatrix}^T$ gesetzt wird. Dies ist notwendig, da sonst ein Hypervolumen von $\mathcal{S}_{MKS}^{(s)} > 0$ nur selten gefunden wird und der Optimierer sich mit der gezielten Verbesserung des Volumens schwer tut.

Die Strategie konvergiert nach 17 äußeren Iterationen und benötigt insgesamt, inklusive der SSO Iterationen, 56 Iterationen. Dabei werden für das erste Segment 635 und für das zweite 614 Funktionsauswertungen benötigt. Die Pareto-Fronten der Architektur mit dem höchsten mittleren dominierten Hypervolumen $f_{\mathcal{S},MKS}^{(A)}$ sind in Abb. 6.3 dargestellt. Für den zugehörigen Architekturentwurf ergeben sich die Architekturvariablenwerte $x_{Zst} = 1$ und $z_{Zst} = 0{,}457$. Die Lenkung sollte also innerhalb der Grenzen weitestgehend im Fahrzeug nach hinten verschoben werden, wohingegen die Höhe der Lenkung

nahezu unverändert verbleiben kann.

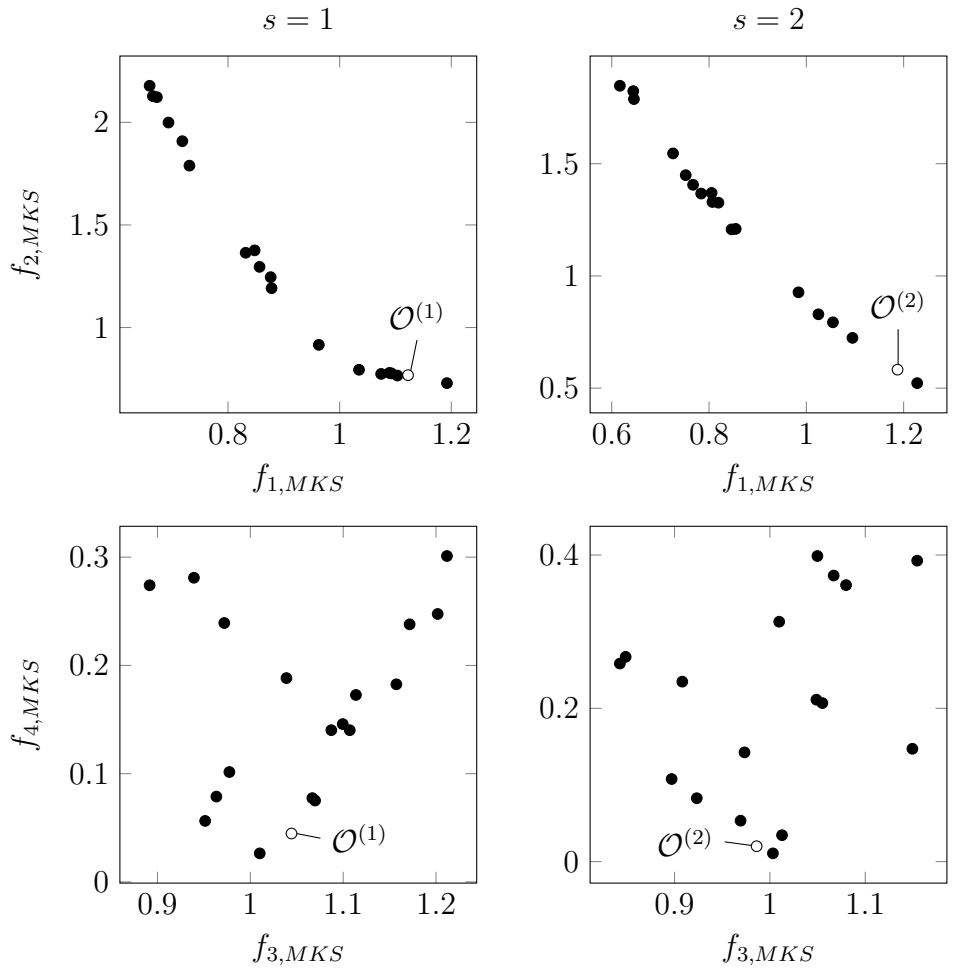

Abbildung 6.3: Optimale Ergebnisse des Anwendungsbeispiels mit ausgewählten Kompromisslösungen $\mathcal{O}^{(s)}$

Deutlich zu sehen ist der Kompromiss zwischen Agilität, repräsentiert durch $f_{1,MKS}$, und Bremsstabilität, repräsentiert durch $f_{2,MKS}$. Die Verbesserung der Agilität führt unweigerlich zu einer Verschlechterung der Spurtreue beim Bremsen. Widersprüchlichkeit zeigt sich auch bei den Kriterien $f_{3,MKS}$ und $f_{4,MKS}$, wobei jedoch keine direkte Korrelation zu erkennen ist. Dementsprechend wurden nur wenige Entwürfe gefunden, bei denen die Kriterien $f_{3,MKS}$ kleiner 1 und $f_{4,MKS}$ nahe 0 liegen.

Für beide Segmente wird jeweils eine Kompromisslösung $\mathcal{O}^{(s)}$ für eine genauere

Analyse ausgewählt. Bei den gewählten Lösungen wird der Fokus auf die Fahrsicherheit gelegt und sie einer besseren Agilität vorgezogen. Für die optimale Auslegung einer einheitlichen Lenkung sind die Kompromisslösungen $\mathcal{O}^{(s)}$ so gewählt, dass die Zahnstangenkräfte F_{zst} beider Segmente sich angleichen, worauf später genauer eingegangen wird. Zusätzlich liegen die Kriterienwerte von $f_{4,MKS}$ nahe 0, damit die gewünschte Lenkcharakteristik erhalten bleibt. Ein Vergleich ausgewählter Größen von Referenz- und Kompromisslösungen sind in Abb. 6.4 dargestellt.

Es ist zu sehen, was bereits oben erwähnt wurde, nämlich dass die Agilität beider Segmente, repräsentiert durch den Eigenlenkbedarf $\bar{\delta}_{EG}$, geringer als bei den jeweiligen Referenzvarianten ausfällt. Der Eigenlenkbedarf der Kompromisslösungen steigt stärker an, ein Fahrer muss also mehr Lenkarbeit verrichten, um das Fahrzeug auf einer Soll-Trajektorie zu halten. Stark verbessert hat sich das Bremsverhalten, zu sehen an den Verläufen der Giergeschwindigkeit $\dot{\Psi}$. Insbesondere bei Segment $s = 2$ konnte die Spurstabilität beim Bremsen massiv verbessert werden. Bei der Zahnstangenkraft F_{zst} wird eine Erhöhung für Segment $s = 1$ bewusst in Kauf genommen, während die Kraftmaxima von Segment $s = 2$ geringer als bei der Referenz ausfallen. Insgesamt liegen die Kräfte beider Segmente nun auf einem ähnlichen Niveau, während die Ausgangsvarianten deutlich größere Unterschiede aufweisen. Das Ziel einer einheitlichen Architektur ist die Verwendung von Gleichteilen, es würde also wenig Sinn ergeben, ein Segment mit einer geringeren Zahnstangenkraft anzustreben, da in diesem Fall das andere Segment mit der höheren Zahnstangenkraft die Auslegung der Lenkung bestimmen würde.

Zusammenfassend lässt sich festhalten, dass sich mithilfe der CSSO-Strategie auch komplexe und durch zeitaufwändige Mehrkörpersimulationen zu berechnende Fahrzeugmodelle mit noch akzeptablem Aufwand auslegen lassen. Die gesamte Optimierung hat etwa $3\frac{1}{2}$ Tage benötigt, wobei bis zu 10 MKS-Simulationen parallel durchgeführt wurden. Es hat sich gezeigt, dass trotz vereinheitlichter Elemente ausreichend Flexibilität erhalten bleibt, so dass aus einer Architektur Segmente mit unterschiedlichen Eigenschaften abgeleitet werden können. Die Anzahl der direkt berechneten Pareto-optimalen Entwürfe ist allerdings noch zu

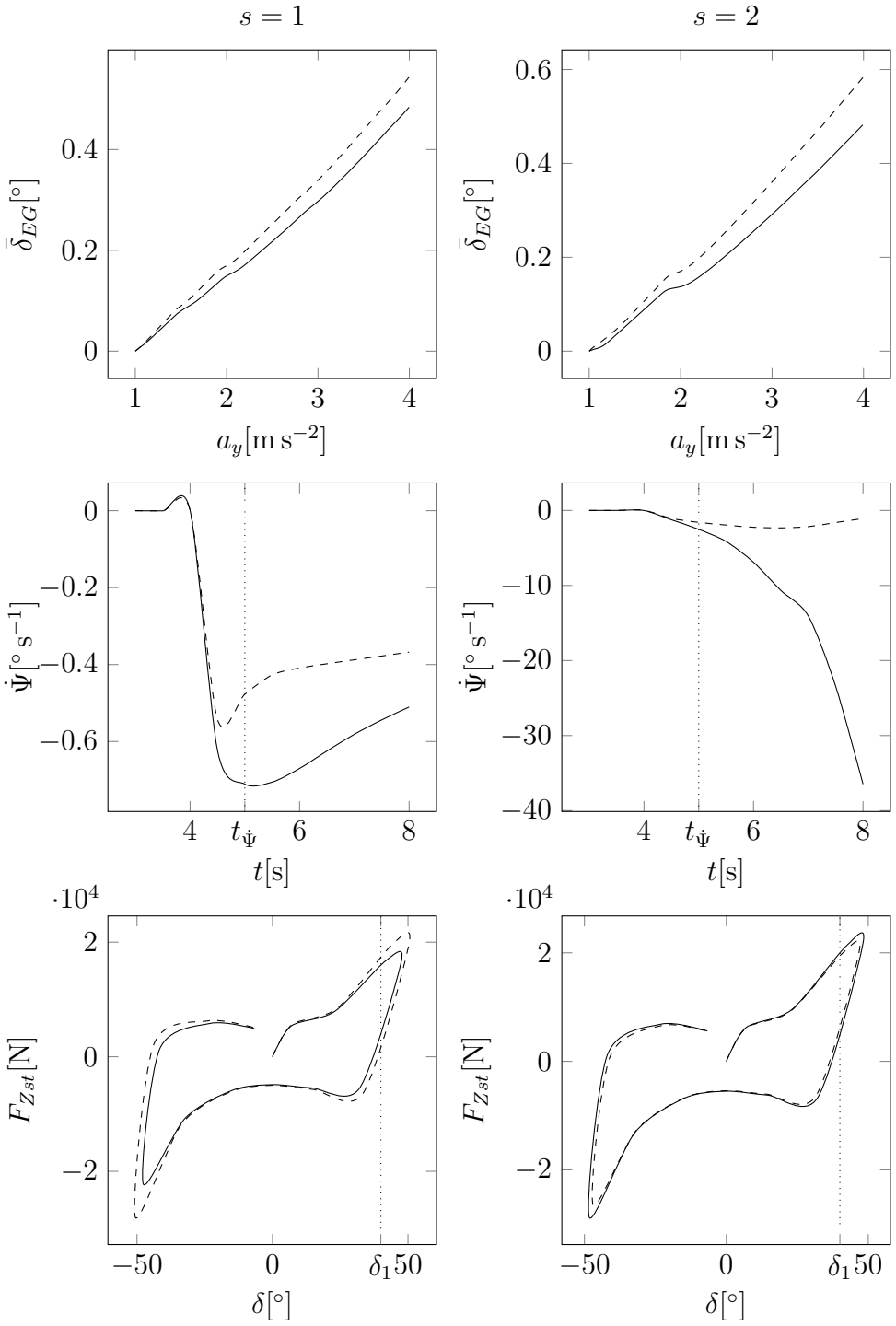

Abbildung 6.4: Darstellung der Kurvenverläufe ausgewählter Größen der Referenz- (durchgezogen) und der Kompromisslösungen $\mathcal{O}^{(s)}$ (gestrichelt)

gering, so dass die Wahl von Kompromisslösungen eingeschränkt ist. Auf Basis der gefundenen Architektur bietet es sich daher an, weitere Pareto-optimale Entwürfe der Segmente zu berechnen, um eine noch größere Wahlmenge potentieller Kompromisslösungen bereit zu halten.

Kapitel 7

Zusammenfassung und Ausblick

Gemeinsame Plattformen und Architekturen gewinnen in der Produktentwicklung und insbesondere im Automobilbereich zunehmend an Bedeutung. Geringeren Kosten und kürzeren Entwicklungszeiten steht eine deutlich gesteigerte Systemkomplexität gegenüber, die es von den Entwicklungsingenieuren zu beherrschen gilt. Um die Ingenieure bei der effizienten Auslegung von Architekturen zu unterstützen, bedarf es der Bereitstellung verschiedener Werkzeuge. Computerbasierte Methoden haben sich bereits in der Vergangenheit als hilfreich erwiesen und haben sich im Entwicklungsprozess etabliert. Für die Auslegung von Architekturen und im Besonderen Fahrwerksarchitekturen, die im Interesse dieser Arbeit stehen, scheint die gezielte Kombinationen dieser Methoden das Mittel der Wahl zu sein, was in dieser Arbeit untersucht wird.

Eines dieser Werkzeuge ist die bereits etablierte Mehrkörpersimulation, mit deren Hilfe digitale Prototypen simuliert und bewertet werden. Die Analyse vieler verschiedener Varianten wird möglich, da keine zeitaufwändigen realen Prototypen hergestellt werden müssen. Die MKS ist deshalb ein wichtiger Baustein für eine erfolgreiche Architekturentwicklung. Ein weiterer Baustein ist die numerische Optimierung. Mit ihr können automatisiert neue Fahrzeugvarianten generiert und in Kopplung mit der MKS simuliert und bewertet sowie gezielt nach einer besten Lösung gesucht werden. Für die Architekturoptimierung ist eine Kopplung dieser Verfahren alleine aber nicht ausreichend. In der vorliegenden Arbeit werden daher verschiedene Konzepte und Strategien entwickelt und untersucht, die Erweiterungen der klassischen Optimierung von einzelnen

Systemen darstellen.

Ein wichtiger Schritt ist die Einführung einer geeigneten Parametrik, um die Besonderheiten von Architekturen und den zugehörigen Parametern in der automatisierten Optimierung berücksichtigen zu können. Auf Basis dieser Parametrik werden anschließend verschiedene Konzepte der Architekturoptimierung vorgestellt und anhand eines einfachen Testproblems analysiert und verglichen. Zusätzlich steht die Mehrzieloptimierung im Fokus. Es wurden zwar bereits mehrkriterielle Architekturoptimierungen in der Literatur behandelt, aber mit dem Ziel, den Kompromiss zwischen Vereinheitlichung und Leistungsfähigkeit einer Architektur aufzudecken. Im Rahmen dieser Arbeit wird ein Verfahren vorgestellt, das eine Architektur bei a priori definierter Vereinheitlichung findet, die die besten Kompromisslösungen der einzelnen Fahrzeuge der Familie in Bezug auf gewählte Kriterien aufdeckt. Die gefundene Architektur bietet dem Hersteller dann die Möglichkeit, kurzfristig neue Fahrzeugsegmente zu bedienen oder auf Marktsituationen zu reagieren, indem die Fahrzeugfamilie, abhängig von der ursprünglichen Planung, auf größte Diversität oder Homogenität umgestellt wird. Als erfolgversprechend hat sich hierbei eine kaskadierte Optimierungsstrategie herausgestellt, die allerdings den massiven Einsatz von zeitaufwändigen Funktionsauswertungen voraussetzt.

Ein Lösungsansatz zur Steigerung der Effizienz der kaskadierten Strategie ist der Einsatz von adaptiven Antwortflächen. Zwei auf einem Prozess zur gezielten Bestimmung neuer Stützstellen basierende Strategien werden untersucht, die deutlich effizienter und fast ebenso genau wie eine direkt durchgeführte Optimierung sind. Damit ist der praxisnahe Einsatz dieser Verfahren und die zugehörige Verwendung zeitintensiver komplexer Gesamtfahrzeugsimulationen möglich und wird am Anwendungsbeispiel veranschaulicht. Hier wird deutlich, dass sich das über Jahrzehnte angeeignete Know-How der Entwickler in bereits nahezu optimal ausgelegten Fahrzeugen widerspiegelt. Dennoch kann die vorgestellte Methode dabei unterstützen, eine geeignete Architektur und hinreichende Kompromisslösungen bereitzustellen.

Die Erweiterung der Strategien zu einer Mehrzieloptimierung ermöglicht dann

die Anwendung einer robusten Architekturoptimierung. Aus verschiedenen vorgestellten Gründen treten unsichere Parameter auf, die zu streuenden Gütekriterien führen. Die Anfälligkeit bzw. Robustheit gegenüber diesen Unsicherheiten kann über den Erwartungswert und die Standardabweichung erfasst werden. Der Kompromiss zwischen höchster mittlerer Leistungsfähigkeit und geringster Streuung der Kriterien wird dann mithilfe der robusten Optimierung aufgedeckt. Auch hier erweist sich die gewählte Strategie als sehr effizient, das erzielte Ergebnis reicht aber nicht an die Qualität einer direkt ausgeführten robusten Architekturoptimierung heran.

Abschließend lässt sich festhalten, dass das Anwendungsbeispiel und die gezeigte robuste Architekturoptimierung noch weiteres Verbesserungspotential offenbaren. Bei beiden Optimierungen wird die als optimal vermutete Architektur in nur einer Iteration gefunden, wodurch die Anzahl Pareto-optimaler Entwürfe gering ausfällt. Neben der einfachen Verschärfung von Konvergenzkriterien oder der Erhöhung der maximal zulässigen Anzahl von Iterationen könnte ein zusätzliches, auf das wiederholte Finden desselben oder sehr ähnlicher Architekturentwürfe achtendes Kriterium eingeführt werden. Somit könnte das Bestimmen ausreichend vieler Pareto-optimaler Entwürfe sichergestellt werden, ohne dass die Anzahl neuer Stützstellen bzw. direkte Funktionsauswertungen pro Iteration steigt, die auch in uninteressanten Regionen des Entwurfsraums liegen können. Für die robuste Architekturoptimierung bieten sich die in Abschnitt 5.4 erwähnten lokalen Antwortflächen zur Approximation von Erwartungswert und Standardabweichung an, um trotz weniger direkter Funktionsauswertungen die Schätzgenauigkeit und somit das Rauschen der Kriterien zu reduzieren. Zu guter Letzt würde die mehrkriterielle Betrachtung der Architekturoptimierung weiteres Potential bieten. Der Raum der Architekturkriterien würde sich über die Hypervolumina der Segmente aufspannen. Eine Optimierung würde dann die Pareto-Front ergeben, die abhängig von den gewählten Architekturvariablen den Kompromiss zwischen möglicher Leistungsfähigkeit und Diversität aller Segmente aufzeigt.

Anhang A

Berechnung des Hypervolumens

In dieser Arbeit wird ein von Fonseca et al. [30] vorgestellter *Dimension-Sweep* Algorithmus zur Berechnung des von Pareto-optimalen Punkten dominierten Hypervolumens verwendet. Der Algorithmus bietet sich für die Anwendung an, da er als Python-Bibliothek verfügbar ist und somit einfach in eigene Python-Skripte integriert werden kann. Die Funktionsweise des Algorithmus wird hier beispielhaft für $h = 3$ Dimensionen erläutert. Die Funktionsweise für $h > 3$ Dimensionen kann in der angegebenen Quelle nachgeschlagen werden.

Gegeben ist eine Menge

$$P = \{\boldsymbol{p}_k, k = 1 \dots K\} \subset \mathbb{R}^3 \tag{A.1}$$

von (Pareto-optimalen) Punkten

$$\boldsymbol{p}_k = \begin{bmatrix} p_{k,x} & p_{k,y} & p_{k,z} \end{bmatrix}^T, \tag{A.2}$$

deren dominiertes Hypervolumen in Bezug auf einen Referenzpunkt $\boldsymbol{p}^{ref}$ berechnet werden soll. *Dimension-Sweep* bedeutet nun, dass der Algorithmus sich entlang einer Dimension des Raumes hangelt. Das Problem wird dann mit $h_{sub} = h - 1$ in h_{sub}-dimensionale Teilprobleme unterteilt, die nacheinander gelöst werden. Dafür werden die Punkte $\boldsymbol{p}_k$ in aufsteigender Reihenfolge ihrer $p_{k,z}$-Werte in eine Liste L einsortiert. Zusätzlich gibt es einen Binärbaum T. Darin gespeicherte Punkte entsprechen nicht-dominierten Projektionen in der xy-Ebene, dargestellt durch die gestrichelte Linie in der beispielhaften Darstellung in Abb. A.1a. Wichtig ist, dass jede nicht-dominierte Menge von Punkten

in zwei Dimensionen streng entlang beider Dimensionen, in einer aufsteigend und in der anderen absteigend, sortiert werden kann. Innerhalb von T wird mit $z^+(\boldsymbol{p})$ und $z^-(\boldsymbol{p})$ der Punkt direkt über bzw. unter $\boldsymbol{p}$ entlang der z-Achse beschrieben. Dies gilt analog für y mit $y^+(\boldsymbol{p})$ und $y^-(\boldsymbol{p})$.

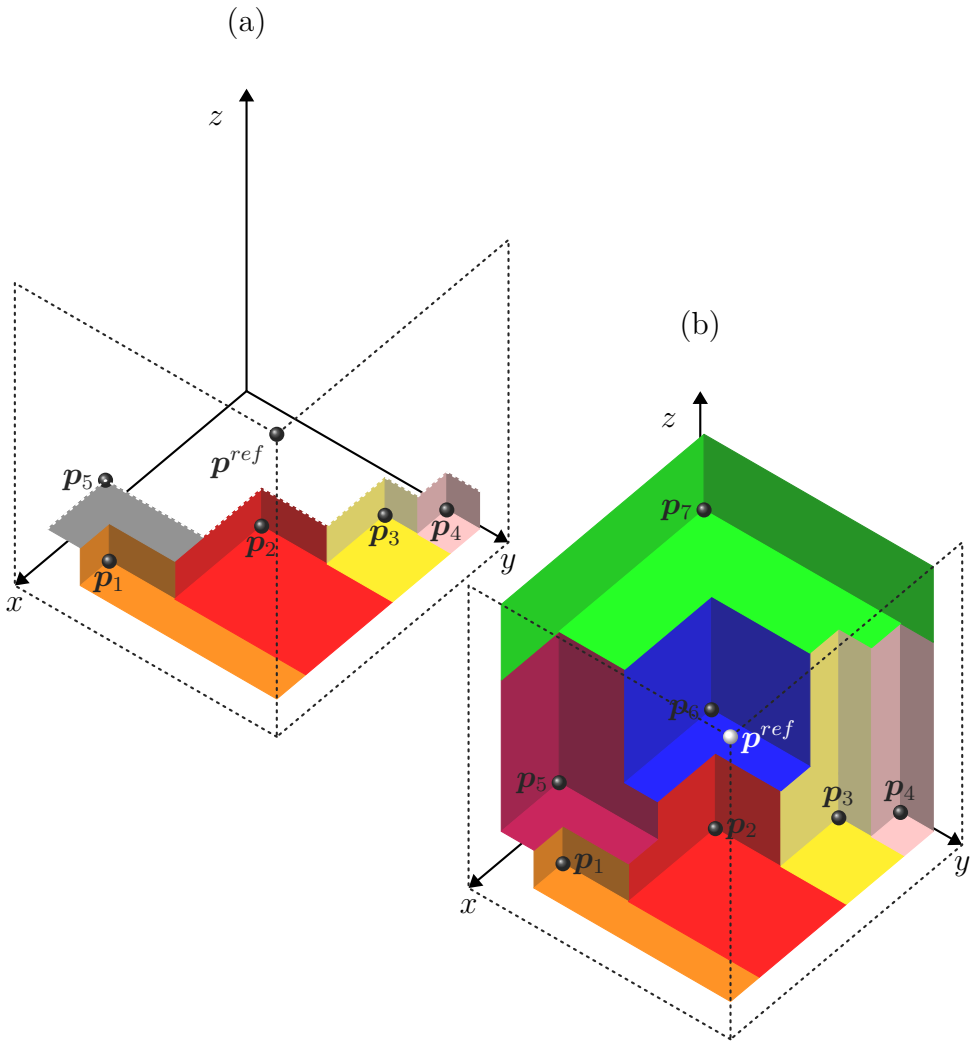

Abbildung A.1: Abschnittweise Berechnung des Hypervolumens entlang der z-Achse (a) und vollständig berechnetes Hypervolumen (b)

Im ersten Schritt wird der erste Punkt aus L entnommen und zum Binärbaum T, der den initialen Wächterknoten $\boldsymbol{p}^{ref}$ enthält, hinzugefügt. Der Wächterknoten hat die Eigenschaft, dass er nicht aus T entfernt werden kann. Im Beispiel Abb. A.1 gibt es mehrere Punkte die den gleichen niedrigsten z-Wert aufweisen. In diesem Fall werden Punkte in aufsteigender y-Richtung sortiert.

Im Beispiel ist das zunächst $\boldsymbol{p}_1$. Das Hypervolumen wird mit $\mathcal{S} := 0$ initialisiert und die Fläche der Projektion auf $A := \left(p_x^{ref} - p_{1,x}\right)\left(p_y^{ref} - p_{1,y}\right)$ gesetzt. Für jeden weiteren Punkt $\boldsymbol{p}_i \in L$ wird dann entsprechend folgender Erläuterungen vorgegangen.

Zunächst wird der aktuelle Punkt $\boldsymbol{p}_i$, im Beispiel $\boldsymbol{p}_2$, aus L entfernt. Jeder Punkt, dessen Koordinaten in mindestens einer Richtung größer oder gleich der des Referenzpunktes $\boldsymbol{p}^{ref}$ ist, also $p_{i,x} \geq p_x^{ref}$, $p_{i,y} \geq p_y^{ref}$ oder $p_{i,z} \geq p_z^{ref}$ gilt, wird verworfen und mit dem nächsten fortgefahren. Für einen gültigen Punkt $\boldsymbol{p}_i$ wird das Volumen auf Basis des vorherigen Punktes in z-Richtung aus T mit

$$\mathcal{S} := \mathcal{S} + A\left(p_{i,z} - z^{-}\left(\boldsymbol{p}_i\right)_z\right) \tag{A.3}$$

aktualisiert. Gibt es zwei Punkte mit dem selben z-Wert, ergibt sich keine Volumenänderung. Anschließend wird ein Punkt $\boldsymbol{q} = y^{-}\left(\boldsymbol{p}_i\right)$ in T gesucht, der den nächst kleineren y-Wert aufweist. Wird kein regulärer Punkt $\boldsymbol{q}$ in T gefunden, wird der Wächterknoten $\boldsymbol{q} = \boldsymbol{p}^{ref}$ ausgewählt. Mit $q_x \leq p_{i,x}$ wird dann geprüft, ob dieser $\boldsymbol{p}_i$ dominiert. Wenn ja, wird $\boldsymbol{p}_i$ verworfen und mit dem nächsten Punkt $\boldsymbol{p}_i$ mit $i := i + 1$ aus L fortgefahren. Im Beispiel entspricht $\boldsymbol{q} = \boldsymbol{p}_1 = y^{-}\left(\boldsymbol{p}_2\right)$ und $\boldsymbol{p}_2$ wird nicht dominiert.

Wenn dieser $\boldsymbol{p}_i$ nicht dominiert, wird die Existenz eines Punktes $\boldsymbol{t} = y^{+}\left(\boldsymbol{p}_i\right)$ in T geprüft, der den nächst höheren y-Wert aufweist. Existiert dieser und gilt $p_{i,x} \leq t_x$, so wird $\boldsymbol{t}$ aus T entfernt und die dominierte Fläche mit

$$A := A - \left(q_x - t_x\right)\left(y^{+}\left(\boldsymbol{t}\right)_y - t_y\right) \tag{A.4}$$

aktualisiert. Dies wird so lange wiederholt, bis ein Punkt $\boldsymbol{t}$ in T nicht von $\boldsymbol{p}_i$ dominiert wird oder bis kein regulärer Punkt mehr gefunden werden kann. In diesem Fall wird für die Flächenberechnung der Wächterknoten $\boldsymbol{t} = \boldsymbol{p}^{ref}$ ausgewählt. Schlussendlich wird $\boldsymbol{p}_i$ zu T hinzugefügt, die Fläche mit

$$A := A + \left(q_x - p_{i,x}\right)\left(t_y - p_{i,y}\right) \tag{A.5}$$

aktualisiert und mit dem nächsten Punkt aus L fortgefahren. Sollte L keine weiteren Punkte enthalten, ist die Berechnung des Volumens beendet.

Im Beispiel wird mit $\boldsymbol{p}_4$ der letzte reguläre Punkt mit gleichem z-Wert gefunden und die resultierende dominierte Fläche berechnet. Der Punkt mit dem nächst höheren z-Wert ist $\boldsymbol{p}_5$, so dass das Volumen (A.3) berechnet wird und der Algorithmus fortfährt, bis das gesamte Volumen in Abb. A.1b berechnet wurde.

Literaturverzeichnis

[1] Albers, I.: *Auslegungs- und Optimierungswerkzeuge für die effiziente Fahrwerkentwicklung.* Aachen: Forschungsges. Kraftfahrwesen, 2009.

[2] Allemang, R. J.: The Modal Assurance Criterion – Twenty Years of Use and Abuse. *Sound and Vibration* 37 (8), 2003, 14–23.

[3] Anderson, E.; Bai, Z.; Bischof, C.; Blackford, S.; Demmel, J.; Dongarra, J.; Du Croz, J.; Greenbaum, A.; Hammarling, S.; McKenney, A.; Sorensen, D.: *LAPACK Users' Guide.* Philadelphia: Society for Industrial and Applied Mathematics, 1999.

[4] Baldwin, C. Y.; Woodard, C. J.: *The Architecture of Platforms: A Unified View.* Boston: Harvard Business School, 2008. (Working Paper)

[5] Bestle, D.: *Analyse und Optimierung von Mehrkörpersystemen: Grundlagen und rechnergestützte Methoden.* Berlin: Springer, 1994.

[6] Bestle, D.: *Optimierung dynamischer Systeme: Vorlesungsskript SS 2013.* Cottbus: Brandenburgische Technische Universität, 2013.

[7] Bestle, D.: *Systematische Entwurfsstrategien für automatisierte Entwurfsprozesse: Effizientsteigerung und Robustheit durch Mehrkriterienoptimierung.* Berlin: Haus der Technik, Außeninstitut der RWTH Aachen, 2013. (Seminarunterlage)

[8] Beyer, H.-G.; Sendhoff, B.: Robust Optimization – A Comprehensive Survey. *Computer Methods in Applied Mechanics and Engineering* 196 (33-34), 2007, 3190–3218 oder DOI 10.1016/j.cma.2007.03.003.

[9] Blum, S.; Will, J.: *Combining Robustness Evaluation with Currend Automotive MDO Application.* Weimar: Dynardo GmbH,

2006. http://www.dynardo.de/fileadmin/Material_Dynardo/bibliothek/WOST_3.0/WOST_3_RobustnessMDO_En.pdf, Zugriffsdatum 01/2015.

[10] Box, G. E. P.; Wilson, K. B.: On the Experimental Attainment of Optimal Conditions. *Journal of the Royal Statistical Society* 13 (1), 1951, 1–45.

[11] Braess, H.-H.; Seiffert, U.: *Vieweg Handbuch Kraftfahrzeugtechnik*. Wiesbaden: Vieweg+Teubner Verlag, 2012.

[12] BROYDEN, C. G.: The Convergence of a Class of Double-rank Minimization Algorithms. *Journal of the Institute of Mathematics and its Applications* 6 (1), 1970, 76–90 oder DOI 10.1093/imamat/6.1.76.

[13] BROYDEN, C. G.: The Convergence of a Class of Double-rank Minimization Algorithms. *Journal of the Institute of Mathematics and its Applications* 6 (3), 1970, 222–231 oder DOI 10.1093/imamat/6.3.222.

[14] Bucher, C.: *Using Response Surface Methodology for Robust Design Optimization: Introduction and Overview*. Weimar: Dynardo GmbH, 2007. http://www.dynardo.de/fileadmin/Material_Dynardo/WOST/Paper/wost4.0/Presentation_Bucher.pdf, Zugriffsdatum 04/2015.

[15] Busch, J.; Bestle, D.: Optimisation of Lateral Car Dynamics Taking Into Account Parameter Uncertainties. *Vehicle System Dynamics* 52 (2), 2014, 166–185 oder DOI 10.1080/00423114.2013.868006.

[16] Cavazzuti, M.: *Optimization Methods: From Theory to Design – Scientific and Technological Aspects in Mechanics*. Heidelberg: Springer, 2013.

[17] Chen, A. L. ; Martinez, D. H.: A Heuristic Method Based on Genetic Algorithm for the Baseline-Product Design. *Expert Systems with Applications* 39 (5), 2012, 5829–5837 oder DOI 10.1016/j.eswa.2011.11.084.

[18] Chen, C.–B.; Wang, L.–Y.: A Modified Genetic Algorithm for Product Family Optimization with Platform Specified by Information Theoretical Approach. *J. Shanghai Jiaotong Univ. (Sci.)* 13 (3), 2008, 304–311 oder DOI 10.1007/s12204–008–0304–4.

[19] Cheng, X.; Lin, Y.: Multiobjective Robust Design of the Double Wishbone Suspension System Based on Particle Swarm Optimization. *The Scientific World Journal* 2014, 2014, 1–7 oder DOI 10.1155/2014/354857.

[20] Coello Coello, C. A.; Lamont, G. B.; Van Veldhuisen, D. A.: *Evolutionary Algorithms for Solving Multi-Objective Problems*. New York: Springer, 2007.

[21] Cornet, A.: *Plattformkonzepte in der Automobilentwicklung*. Wiesbaden: Dt. Univ.-Verl., 2002.

[22] de Weck, O. L.; Suh, E. S.; Chang, D.: *Product Family Strategy and Platform Design Optimization*. Cambridge: Massachusetts Institute of Technology, 2004. (Working Paper)

[23] DYNARDO GmbH: *optiSLang - Software für Sensitivitätsanalyse, Mehrzieloptimierung, multidisziplinäre Optimierung, Robustheitsbewertung, Zuverlässigkeitsanalyse und Robust Design Optimierung*. `http://www.dynardo.de/software/optislang.html`, Zugriffsdatum 04/2015.

[24] DYNARDO GmbH: *Recent Developments in the Metamodel of Optimal Prognosis*. Weimar: Dynardo GmbH, 2014. `http://www.dynardo.de/fileadmin/Material_Dynardo/bibliothek/WOST11/11_WOST2014_Session3_Most.pdf`, Zugriffsdatum 01/2015.

[25] Emmerich, M.; Beume, N.; Naujoks, B.: An EMO Algorithm Using the Hypervolume Measure as Selection Criterion. In: *Proceedings of Evolutionary Multi-Criterion Optimization: Third International Conference*. Berlin: Springer, 2005, 62–76 oder DOI 10.1007/978-3-540-31880-4_5.

[26] Fellini, R.; Kokkolaras, M.; Papalambros, P. Y.: Commonality Decisions in Product Family Design. In: *Product Platform and Product Family Design*. New York: Springer, 2006, 157–185 oder DOI 10.1007/0-387-29197-0_9.

[27] Fellini, R.; Kokkolaras, M.; Papalambros, P.; Perez-Duarte, A.: Platform Selection Under Performance Bounds in Optimal Design of

Product Families. *Journal of Mechanical Design* 127 (4), 2005, 524–535 oder DOI 10.1115/1.1899176.

[28] Fisher, R. A.: *The Design of Experiments*. Edinburgh: Oliver & Boyd, 1935.

[29] Fletcher, R.: A New Approach to Variable Metric Algorithms. *The Computer Journal* 13 (3), 1970, 317–322 oder DOI 10.1093/comjnl/13.3.317.

[30] Fonseca, C. M.; Paquete, L.; Lopez-Ibanez, M.: An Improved Dimension-Sweep Algorithm for the Hypervolume Indicator. In: *Proceedings of the 2006 IEEE International Conference on Evolutionary Computation (CEC)*. Piscataway: IEEE Press, 2006.

[31] Friedrich, T. ; Menzel, S.: A Cascaded Evolutionary Multi-objective Optimization for Solving the Unbiased Universal Electric Motor Family Problem. In: *Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC)*. Piscataway: IEEE Press, 2014, 3184-3191 oder DOI 10.1109/CEC.2014.6900605.

[32] FROST & SULLIVAN: *Analysis of Vehicle Platform Strategies of Key Global OEMs: 30 Percent Reduction in Vehicle Platforms by 2020*. `http://www.frost.com/sublib/display-report.do?id=M726-01-00-00-00`, Zugriffsdatum 05/2016.

[33] Goldfarb, D.: A Family of Variable-Metric Methods Derived by Variational Means. *Mathematics of Computation* 24 (109), 1970, 23–26 oder DOI 10.2307/2004873.

[34] Göpfert, J.: *Modulare Produktentwicklung: Zur gemeinsamen Gestaltung von Technik und Organisation*. Wiesbaden: Deutscher Universitätsverlag, 1998.

[35] Griffiths, N.; Hayfield, A.; Couchman, C.; Fulbrook, A.; Fulthorpe, Mark: *Five Critical Challenges Facing the Automotive Industry: A Guide for Strategic Planners*. Englewood: IHS Automotive, 2015. `http://cdn.ihs.com/www/pdf/AUT-TL-WhitePaper-5.pdf`, Zugriffsdatum 12/2016.

[36] Haug, E. J.: *Computer Aided Kinematics and Dynamics of Mechanical Systems*. Boston: Allyn and Bacon, 1989.

[37] Heide, F. G.: Mauerblümchen am deutschen Auto-Markt: CAR-Studie zur Typenvielfalt. Duisburg/Düsseldorf: Handelsblatt, 2011. http://www.handelsblatt.com/auto/nachrichten/car-studie-zur-typenvielfalt-mauerbluemchen-am-deutschen-auto-markt/4555126.html, Zugriffsdatum 05/2016.

[38] Heissing, B.: *Fahrwerkhandbuch: Grundlagen, Fahrdynamik, Komponenten, Systeme, Mechatronik, Perspektiven*. Wiesbaden: Vieweg+Teubner Verlag / Springer Fachmedien, 2011.

[39] Huntington, D. E. ; Lyrintzis, C. S.: Improvements to and Limitations of Latin Hypercube Sampling. *Probabilistic Engineering Mechanics* 13 (4), 1998, 245–253 oder DOI 10.1016/S0266–8920(97)00013–1.

[40] Jiao, J.; Simpson, T. W.; Siddique, Z.: Product Family Design and Platform-based Product Development: A State-of-the-art Review. *Journal of Intelligent Manufacturing* 18 (1), 2007, 5–29 oder DOI 10.1007/s10845–007–0003–2.

[41] Kacher, G.: Der Stern soll weiter leuchten: Zukunft von Mercedes. München: Süddeutsche Zeitung, 2014. http://www.sueddeutsche.de/auto/zukunft-von-mercedes-der-stern-soll-weiter-leuchten-1.2185552, Zugriffsdatum 10/2014.

[42] Kang, D. O.; Heo, S. J.; Kim, M. S.; Choi, W. C.; Kim, I. H.: Robust Design Optimization of Suspension System by Using Target Cascading Method. *International Journal of Automotive Technology* 13 (1), 2011, 109–122 oder DOI 10.1007/s12239–012–0010–y.

[43] Karius, A.: Chefplaner Klein: "Nissans neue Plattform treibt Absatz und Profit": Common Module Architektur (CMF). Landsberg: Automobil-Produktion, 2015. http://www.automobil-produktion.de/hersteller/wirtschaft/nissans-neue-plattform-treibt-absatz-und-profit-118.html, Zugriffsdatum 04/2016.

[44] Khire, R.; Wang, J.; Bailey, T.; Lin, Y.; Simpson, T. W.: Product Family Commonality Selection Using Optimization and Interactive Visualization. In: *Advances in Product Family and Product Platform Design*. New York: Springer, 2014, 449–471 oder DOI 10.1007/978-1-4614-7937-6_18.

[45] Kim, H. M. ; Michelena, N. F. ; Papalambros, P. Y. ; Jiang, T.: Target Cascading in Optimal System Design. *Journal of Mechanical Design* 125 (3), 2003, 474–480 oder DOI 10.1115/1.1582501.

[46] Kokkolaras, M. ; Fellini, R. ; Kim, H. M. ; Michelena, N. F. ; Papalambros, P. Y.: Extension of the Target Cascading Formulation to the Design of Product Families. *Structural and Multidisciplinary Optimization* 24 (4), 2002, 293–301 oder DOI 10.1007/s00158–002–0240–0.

[47] Kraftfahrt-Bundesamt: *Methodische Erläuterungen und Begriffsbestimmungen zu Statistiken über Fahrzeugzulassungen*. Flensburg: Kraftfahrt-Bundesamt, 2016. `http://www.kba.de/DE/Statistik/Fahrzeuge/fz_methodische_erlaueterungen_201601_pdf.pdf?__blob=publicationFile&v=3`, Zugriffsdatum 01/2016.

[48] Kraftfahrt-Bundesamt: *Neuzulassungen von Personenkraftwagen im April 2016 nach Segmenten und Modellreihen*. Flensburg: Kraftfahrt-Bundesamt, 2016. `http://www.kba.de/SharedDocs/Publikationen/DE/Statistik/Fahrzeuge/FZ/2016_monatlich/FZ11/fz11_2016_04_pdf.pdf?__blob=publicationFile&v=2`, Zugriffsdatum 01/2016.

[49] Lehnerd, A. P.: Revitalizing the Manufacture and Design of Mature Global Products. In: *Technology and Global Industry*. Washington, D.C.: National Academies Press, 1987, 49–64.

[50] Meyer, M. H.; Lehnerd, A. P.: *The Power of Product Platforms: Building Value and Cost Leadership*. New York: Free Press, 1997.

[51] Mitschke, M.; Wallentowitz, H.: *Dynamik der Kraftfahrzeuge*. Wiesbaden: Springer Vieweg, 2014.

[52] Moon, S. K. ; McAdams, D.l A.: Universal Product Platform and Family Design for Uncertain Market. In: *Proceedings of ICED'09, the 17th International Conference on Engineering Design.* Glasgow: Design Society, 2009, 59–70.

[53] Most, T.; Will, J.; *Robust Design Optimization in Industrial Virtual Product Development.* Weimar: Dynardo GmbH, 2012. `http://www.dynardo.de/fileadmin/Material_Dynardo/bibliothek/RDO/MOST_2012_REC_RobustDesignOptimizationInIndustrialVirtualProductDevelopment.pdf`, Zugriffsdatum 01/2015.

[54] Müller, P. C.; Schiehlen, W. O.: *Lineare Schwingungen: Theoretische Behandlung von mehrfachen Schwingern.* Wiesbaden: Akad. Verl.-Anst, 1976.

[55] Nelder, J. A. ; Mead, R.: A Simplex Method for Function Minimization. *The Computer Journal* 7 (4), 1965, 308–313.

[56] Papageorgiou, M.; Leibold, M.; Buss, M.: *Optimierung: Statische, dynamische, stochastische Verfahren für die Anwendung.* Berlin: Springer, 2012.

[57] Pareto, V.: Manuale di Economia Politica con una Introduzione alla Scienza Sociale. In: *Piccola biblioteca scientifica.* Milano: Societa Editrice Libraria, 1909.

[58] Park, K.; Heo, S. J.; Kang, D. O.; Jeong, J. I.; Yi, J. H.; Lee, J. H.; Kim, K. W.: Robust Design Optimization of Suspension System Considering Steering Pull Reduction. *International Journal of Automotive Technology* 14 (6), 2013, 927–933 oder DOI 10.1007/s12239-013-0102-3.

[59] Pfeffer, P.; Hofer, K.: Einfaches nichtlineares Modell für Elastomer- und Hydrolager. *ATZ* 104 (5), 2002, 442–446 und 450–451.

[60] Pine, B. J.: *Mass Customization: The New Frontier in Business Competition.* Boston: Harvard Business School Press, 1993.

[61] Pirmoradi, Z.; Wang, G. G. ; Simpson, T. W.: A Review of Recent Literature in Product Family Design and Platform-Based Product Development. In: *Advances in Product Family and Product Platform Design*. New York: Springer, 2014, 1–46 oder DOI 10.1007/978-1-4614-7937-6_1.

[62] PROGRENIUM GmbH & Co. KG: *Das Dilemma mit der Vielfalt*. München: Progrenium GmbH, 2015. http://www.progenium.com/Publikationen/DE/data/upload/publikation/PROGENIUM_Pressemitteilung_Das%20Dilemma%20mit%20der%201424941380.pdf, Zugriffsdatum 02/2015.

[63] Ranft, N.: Der Baukasten für die Zukunft: Mit Audi A3 und Golf starten 2012 die ersten Fahrzeuge auf Basis des MQB. In: *autogramm* (1-2), 2012. http://autogramm.volkswagen.de/01-02_12/standorte/standorte_01.html, Zugriffsdatum 08/2016.

[64] Rill, G.; Schaeffer, T.: *Grundlagen und Methodik der Mehrkörpersimulation*. Wiesbaden: Vieweg+Teubner Verlag / GWV Fachverlage, 2010.

[65] Roos, D.; Most, T.; Unger, J.; Will, J.: *Advanced Surrogate Models within the Robustness Evaluation*. Weimar: Dynardo GmbH, 2007. http://www.dynardo.de/fileadmin/Material_Dynardo/WOST/Paper/wost4.0/Paper_Roos_Most_Unger.pdf, Zugriffsdatum 08/2016.

[66] Rothwell, R.; Gardiner, P.: Robustness and Product Design Families. In: *Design Management*. Cambridge: Blackwell Reference, 1990, 279-292.

[67] Sabbagh, K.: *21st Century Jet: The Making and Marketing of the Boeing 777*. New York: Scribner, 1996.

[68] Schiehlen, W.: *Multibody Systems Handbook*. Berlin: Springer, 1990.

[69] Schittkowski, K.: *Sequential Quadratic Programming Methods with Distributed and Non-Monotone Line Search*. Weimar: Dynardo GmbH, 2004. http://www.dynardo.de/fileadmin/Material_Dynardo/bibliothek/WOST_1.0/WOST_1_SequentialQuadraticProgrammingMethods_En.pdf, Zugriffsdatum 05/2016.

[70] Schneider, D.; Bucher, C.: *Efficient RDO Using Sample Recycling.* Weimar: Dynardo GmbH, 2008. http://www.dynardo.de/fileadmin/Material_Dynardo/bibliothek/WOST_5.0/WOST_5_Paper_Schneider.pdf, Zugriffsdatum 04/2015.

[71] Schuh, G.; Arnoscht, J.; Rudolf, S.; Korthals, K.: Modular Chassis Product Platform Considering Variable Quantities for an Economical Electric Vehicle Production. In: *Future Trends in Production Engineering.* Berlin: Springer, 2013, 73–82 oder DOI 10.1007/978-3-642-24491-9_8.

[72] Sedlaczek, K.; Dronka, S.; Rauh, J.: Advanced Modular Modelling of Rubber Bushings for Vehicle Simulations. *Vehicle System Dynamics* 49 (5), 2010, 741–759 oder DOI 10.1080/00423111003739806.

[73] Sellar, R. S.; Stelmack, M. A.; Batill, S. M.; Renaud, J. E.: Response Surface Approximations for Discipline Coordination in Multidisciplinary Design Optimization. In: *AIAA Structures, Structural Dynamics, and Materials Conference and Exhibit* 37. Reston: AIAA, 1996, 583–593.

[74] Shanno, D. F.: Conditioning of Quasi-Newton Methods for Function Minimization. *Mathematics of Computation* 24 (111), 1970, 647–656 oder DOI 10.1090/S0025–5718–1970–0274029–X.

[75] Siebertz, K.; van Bebber, D.; Hochkirchen, T.: *Statistische Versuchsplanung: Design of Experiments (DoE).* Heidelberg: Springer, 2010.

[76] Siemens AG: *LMS Virtual.Lab: Accurate 3D Performance Simulation.* https://www.plm.automation.siemens.com/de_de/products/lms/virtual-lab/#lightview%26url=/de_de/Images/Siemens-PLM-LMS-Virtual-Lab-br_tcm73-218353.pdf%26title=LMSVirtual.LabBrochure%26description=Accurate3Dperformancesimulation%26docType=pdf, Zugriffsdatum 07/2016.

[77] Simpson, T. W.: Methods for Optimizing Product Platforms and Product Families: Overview and Classification. In: *Product Platform and Product Family Design.* New York: Springer, 2006, 133–156 oder DOI 10.1007/0-387-29197-0_8.

[78] Simpson, T. W.; Chen, W.; Allen, J. K.; Mistree, F.: Use of the Robust Concept Exploration Method to Facilitate the Design of a Family of Products. In: *Simultaneous Engineering: Methodologies and Applications*. Amsterdam: Gordon and Breach Science Publishers, 1999, 247-278.

[79] Simpson, T. W.; Jiao, J.; Siddique, Z.; Hölttä-Otto, K.: *Advances in Product Family and Product Platform Design*. New York: Springer, 2014.

[80] Simpson, T. W.; Siddique, Z.; Jiao, J.: Platform-Based Product Family Development. In: *Product Platform and Product Family Design*. New York: Springer, 2006, 1–15 oder DOI 10.1007/0-387-29197-0_1.

[81] Simpson, T. W.; Siddique, Z.; Jiao, J.: *Product Platform and Product Family Design: Methods and Applications*. New York: Springer, 2006.

[82] Eckardt, S.: *Modularer Querbaukasten: Volkswagen setzt auf flexible Fahrzeugarchitektur*. Haar: WEKA Fachmedien, 2012. `http://www.elektroniknet.de/automotive/sonstiges/artikel/85505/0/`, Zugriffsdatum 04/2016.

[83] Taguchi, G.; Wu, Y.: *Introduction to Off-line Quality Control*. Nagaya: Central Japan Quality Control Association, 1985.

[84] Tarkian, M.; Ölvander, J.; Feng, X.; Pettersson, M.: Product Platform Automation for Optimal Configuration of Industrial Robot Families. In: *Proceedings of the 18th International Conference on Engineering Design (ICED)*. Great Glasgow: Design Society, 2011, 1–10.

[85] The MathWorks, Inc.: *MATLAB Release 2014b*. Natick: The MathWorks, Inc., 2014.

[86] Trzesniowski, M.: *Rennwagentechnik: Grundlagen, Konstruktion, Komponenten, Systeme*. Wiesbaden: Vieweg+Teubner Verlag, 2012.

[87] Ubben, P. T. ; Haug, J. ; Bestle, D.: *Robust Automotive Suspension Design Using Adaptive Response Surface Based Multi-Objective Optimization*. Weimar: Dynardo GmbH, 2014. `http://www.dynardo.de/fileadmin/Material_Dynardo/bi`

bliothek/WOST11/09_WOST2014_Session3_Ubben.pdf, Zugriffsdatum 03/2016.

[88] Ubben, P. T.; Haug, J.; Bestle, D.: Robust Automotive Suspension Design Using Multi-Objective Optimization. *RDO-Journal* (2), 2015, 10–15. https://www.dynardo.de/fileadmin/Material_Dynardo/dokumente/broschuere/RDO_02_2015_Journal_P_Web.pdf, Zugriffsdatum 03/2016.

[89] Ubben, P. T.; Haug, J.; Bestle, D.: A Concept for Optimal Design of Automotive Chassis Architectures. In: *Proceedings of 7th International Munich Chassis Symposium 2016*. Wiesbaden: Springer, 2016, 327–348 oder DOI 10.1007/978-3-658-14219-3_24.

[90] Vis, J. K.: *Particle Swarm Optimizer for Finding Robust Optima*. Leiden: Leiden Institute of Advanced Computer Science, 2009. http://www.liacs.nl/assets/Bachelorscripties/2009-12JonathanVis.pdf, Zugriffsdatum 01/2015.

[91] Wang, W.: Scalable Platform Design Optimization Using Hybrid Coevolutionary Algorithms. In: *Proceedings of 2nd International Conference on Software Engineering and Service Science (ICSESS)*. Piscataway: IEEE Press, 2011, 270–273 oder DOI 10.1109/ICSESS.2011.5982306.

[92] Wedeniwski, S.: *Mobilitätsrevolution in der Automobilindustrie: Letzte Ausfahrt digital!* Berlin: Springer, 2015.

[93] Weise, T.: *Global Optimization Algorithms - Theory and Application*. Kassel: University of Kassel, Distributed Systems Group, 2009. http://www.it-weise.de/projects/book.pdf, Zugriffsdatum 12/2016.

[94] Will, J.; Most, T.: *Metamodell of Optimal Prognosis (MOP) - An Automatic Approach for User-friendly Parameter Optimization*. Weimar: Dynardo GmbH, 2009. http://www.dynardo.de/fileadmin/Material_Dynardo/bibliothek/WOST_6.0/WOST_6_Paper_Will_Most.pdf, Zugriffsdatum 01/2015.

[95] Yang, Q.; Huang, J.; Wang, G.; Karimi, H. R.: An Adaptive Metamodel-Based Optimization Approach for Vehicle Suspension System

Design. *Mathematical Problems in Engineering* 2014, 2014, 1–9 oder DOI 10.1155/2014/965157.

[96] Zapico, M.; Eckert, C.; Jowers, I.; Earl, C.: Towards Product Platform Introduction: Optimising Commonality of Components. In: *Proceedings of the 20th International Conference on Engineering Design (ICED)*. Glasgow: Design Society, 2015, 23–32.

[97] Zeller, P.: *Handbuch Fahrzeugakustik: Grundlagen, Auslegung, Berechnung, Versuch*. Wiesbaden: Vieweg + Teubner, 2012.

[98] Zitzler, E.; Thiele, L.; Laumanns, M.; Fonseca, C. M.; da Fonseca, V. G.: Performance Assessment of Multiobjective Optimizers: an Analysis and Review. *IEEE Transactions on Evolutionary Computation* 7 (2), 2003, 117–132 oder DOI 10.1109/TEVC.2003.810758.

[99] Zitzler, E.; Laumanns, M.; Thiele, L.: *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*. Zürich: Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische Informatik und Kommunikationsnetze (TIK), 2001. `http://e-collection.library.ethz.ch/eserv/eth:24689/eth-24689-01.pdf`.

[100] Zitzler, E.; Thiele, L.: Multiobjective Optimization Using Evolutionary Algorithms – A Comparative Case Study. In: *Parallel Problem Solving from Nature – Proceedings of 5th International Conference Amsterdam*. Berlin: Springer, 1998, 292–301 oder DOI 10.1007/BFb0056872.