Universität Paderborn
Fakultät für Elektrotechnik, Informatik und Mathematik

Deutsches Zentrum für Luft- und Raumfahrt e.V
Institut für Solarforschung

Masterarbeit

Anwendung ableitungsfreier Optimierungsverfahren zur Ermittlung von Verformungsursachen solarthermischer Parabolrinnenspiegel

Helena Klump
6517005

Paderborn, 24. Oktober 2015

Gutachter:
Prof. Dr. Andrea Walther
Prof. Dr. Michael Dellnitz

Erklärung

Hiermit versichere ich, dass ich diese Arbeit selbstständig und ohne unerlaubte fremde Hilfe angefertigt, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe und dass diese Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder inhaltlich den angegebenen Quellen entnommen wurden, sind als solche kenntlich gemacht.

Ort, Datum	Unterschrift

Danksagung

Die vorliegende Masterarbeit entstand am Institut für Solarforschung des Deutschen Zentrums für Luft- und Raumfahrt (DLR) in Köln.

Ich möchte mich an dieser Stelle bei allen bedanken, die zum Gelingen dieser Arbeit beigetragen haben.

Mein Dank gilt Frau Prof. Dr. Andrea Walther, die mir das Gebiet der Optimierung näher gebracht hat und mir darüber hinaus die Möglichkeit gab diese Arbeit am DLR anzufertigen. Ich bedanke mich für die engagierte und fachliche Betreuung.

Besonderer Dank gilt meinem Betreuer beim DLR, M.Sc. Simon Schneider, für die Bereitstellung der Aufgabenstellung, zahlreiche Hinweise, Ratschläge, technische Hilfestellungen, anregenden Diskussionen und eine sehr angenehme Arbeitsatmosphäre.

Ebenfalls bedanken möchte ich mich bei Dipl.-Inform. Andreas Reinholz für die vielen regen Diskussionen, Denkanstöße und konstruktiven Vorschlägen zum Thema Optimierung.

Außerdem bedanke ich mich bei Herrn Prof. Dr. Michael Dellnitz für die Bereitschaft die Korrektur zu übernehmen.

Des Weiteren möchte ich mich bei allen Mitarbeitern und Praktikanten der Abteilung Qualifizierung des DLR für die fachlichen Anregungen und das tolle Umfeld bedanken.

Zum Schluss bedanke ich mich bei meiner Familie und meinem Freund, die mich immer unterstützt und an meinen Erfolg geglaubt haben.

Inhaltsverzeichnis

1 Einleitung **1**

2 Solarthermische Grundlagen **4**
2.1 Parabolrinnentechnologie 4
2.1.1 Parabolrinnenkraftwerke 4
2.1.2 Parabolrinnenkollektoren 7
2.2 Qualifizierung der Komponenten 8
2.2.1 Intercept-Faktor 8
2.2.2 Slope Deviation 9

3 Mathematische Grundlagen **12**
3.1 Finite-Elemente Methode 12
3.1.1 Lineare Elastizitätstheorie 13
3.1.2 Finite-Elemente Methode 14
3.2 Optimierung 15
3.2.1 Problemstellung und Lösung eines Optimierungsproblems 16
3.2.2 CMA-ES 17
3.2.2.1 Idee von Evolutionsstrategien 17
3.2.2.2 Konzept 18
3.2.2.3 Stichprobenauswahl und Mittelwertberechnung 19
3.2.2.4 Schrittweitenwahl 19
3.2.2.5 Anpassung der Kovarianzmatrix 20
3.2.2.6 Konvergenz 21
3.2.3 BOBYQA 21
3.2.3.1 Trust-Region Konzept 21
3.2.3.2 Konvergenztheorie 22
3.2.3.3 BOBYQA Konzept 24
3.2.3.4 Polynominterpolation 27
3.2.3.5 Frobeniusnorm-Updating Prozedur 28
3.2.4 Coordinate-Descent Methode 30
3.2.4.1 Konzept des Coordinate-Descent Verfahrens 30
3.2.4.2 Konvergenz 31
3.2.5 Multilevel-Optimierung 31

4 Stand der Technik **33**
4.1 Deflektometrische Messung . . . 33
4.2 FE-Modelle . . . 35
4.2.1 Annahmen . . . 36
4.2.2 Modellbeschreibung . . . 36
4.2.2.1 Spiegelmodelle für die Optimierung . . . 39
4.2.2.2 Modell mit Laborunterstruktur . . . 41

5 Methoden **42**
5.1 Optimierungsprozess . . . 42
5.1.1 Optimierungs- und Referenzmodell . . . 42
5.1.2 Grundprinzip des Optimierungsprozesses . . . 44
5.1.3 Wahl der Optimierungsalgorithmen . . . 45
5.1.4 Zielfunktion . . . 45
5.1.5 Implementierung . . . 46
5.2 Diskretisierungsuntersuchung . . . 49
5.3 Sensitivitätsanalyse . . . 49
5.4 Komma- und Plus Strategie . . . 50
5.5 Hybridalgorithmus . . . 50
5.6 Variation der Zielfunktion . . . 51
5.6.1 Optimierung der z-Deviation . . . 51
5.6.2 Zielfunktion ohne Wichtungsfaktor . . . 52
5.7 Parameterreduktion . . . 52
5.8 Skalierung . . . 52
5.9 Coordinate-Descent . . . 53
5.10 Vorgehensweise bei der Laborunterstruktur . . . 55
5.11 Vorgehensweise bei der realen Spiegelmessung . . . 55
5.11.1 Multilevelansatz . . . 56

6 Testprobleme und Messergebnisse **58**
6.1 Reduzierte Testprobleme mit variabler Parameterzahl . . . 58
6.1.1 Dreidimensionales Testproblem . . . 58
6.1.1.1 Testproblem 1 . . . 58
6.1.1.2 Testproblem 2 . . . 60
6.1.2 Sechsdimensionales Testproblem . . . 60
6.1.3 12-dimensionales Testproblem . . . 61
6.1.4 20-dimensionales Testproblem . . . 62
6.1.5 Laborunterstruktur . . . 62
6.2 Reale Spiegelmessung . . . 63

7 Ergebnisse **65**
7.1 Dreidimensionales Problem . . . 65
7.1.1 Diskretisierungsanalyse und Sensitivitätsanalyse bzgl. der Startschrittweite 65
7.1.1.1 Slope Deviation Plots verschiedener Diskretisierungen . . . 66
7.1.1.2 BOBYQA . . . 66
7.1.1.3 CMA-ES . . . 69
7.1.1.4 Testproblem 1: Vergleich von BOBYQA und CMA-ES . . . 73
7.1.2 Vergleich zwischen Klebstoff- und Padmodell . . . 74
7.1.2.1 Slope Deviation Plots für Klebstoff- und Padmodell . . . 74
7.1.2.2 BOBYQA . . . 75
7.1.2.3 CMA-ES . . . 75
7.1.3 Testproblem 2: Vergleich von BOBYQA und CMA-ES . . . 77
7.2 Sechsdimensionales Problem . . . 78
7.2.1 Test 1 . . . 78
7.2.2 Test 2 . . . 79
7.2.3 Test 3 . . . 79
7.3 12-dimensionales Problem . . . 85
7.3.1 Hybridalgorithmus . . . 85
7.3.2 Identische Zielfunktionswerte . . . 86
7.3.3 Optimierung der z-Deviation . . . 87
7.3.4 Parameterreduktion . . . 89
7.4 20-dimensionales Problem . . . 90
7.4.1 Coordinate-Descent Methode . . . 90
7.4.2 Parameterreduktion . . . 91
7.5 Laborunterstruktur . . . 92
7.5.1 Rotationsuntersuchung . . . 92
7.5.2 Translation und Rotation . . . 94
7.5.2.1 Skalierung der Rotationskomponenten . . . 94
7.5.2.2 Anzahl an Optimierungsparameter . . . 96
7.5.2.3 Sensitivitätsanalyse bezüglich des finalen Trust-Region Radius . 97
7.6 Reale Messung . . . 98
7.6.1 Multilevelvariante . . . 100

8 Zusammenfassung und Ausblick **103**

Literaturverzeichnis **105**

A Anhang **110**
A.1 Basis Trust-Region Algorithmus . . . 110
A.2 CMA-ES Algorithmus . . . 111
A.3 Ein- und Ausgabewerte für die Optimierung . . . 112
A.3.1 CMA-ES . . . 112
A.3.2 BOBYQA . . . 112

A.4 Lösungen des dreidimensionalen Testproblems 113
A.4.1 Testproblem 1 113
A.5 Lösungen des sechsdimensionalen Testproblems 115
A.6 Lösungen des 12-dimensionalen Testproblems 116
A.7 Lösungen des 20-dimensionalen Testproblems 120
A.8 Lösungen der realen Messung 122
A.8.1 Lösungen des Multilevelansatzes 123

Abbildungsverzeichnis

2.1 Prinzip eines Parabolrinnenkraftwerks mit Salzspeicher und getrenntem Kollektor- und Dampfturbinenkreis [BG 10] 5
2.2 EuroTrough-Kollektormodul [Schi 11] 6
2.3 Funktionsprinzip der Parabolrinne 6
2.4 Aufbau eines Parabolrinnenspiegels [Pit 12] 8
2.5 Optische Fehler und optische Verlustmechanismen bei Parabolrinnenkollektoren [Kis03] 9
2.6 Berechnung der Slope Deviation in x-Richtung durch Projektion der Normalenvektoren in die xz-Ebene 10
2.7 Strahlenverlauf für ideale und reale Oberflächennormalenvektoren 10

3.1 Grundprinzip des CMA-ES Algorithmus 18

4.1 Grundprinzip der Deflektometrie zur Bestimmung von Spiegelformabweichungen [Lüp 12] 34
4.2 Ergebnisplots der lokalen Slope Deviation-Werte in x-Richtung (links) und in y-Richtung (rechts) einer Deflektometriemessung eines RP3-Innenspiegels mit SDx =3.5639 mrad 34
4.3 RP3-Innenspiegel mit vier Aufhängepunkten aus verschiedenen Perspektiven . . 37
4.4 Koordinatensystem eines Parabolrinnenkollektormoduls [MLSP 14] 38
4.5 Innenspiegel mit externen Verschiebungen, d.h. Translationen und Rotationen . . 39
4.6 Rückseite des FE-Modells des RP3 Innenspiegels mit Klebstoffschicht, sowie externen Verschiebungen 40
4.7 Rückseite des FE-Modells des RP3 Innenspiegels mit Pads, sowie externen Verschiebungen 40
4.8 FE-Modell des RP3 Innenspiegels fixiert an die Laborunterkonstruktur 41

5.1 Grundsätzliche Methodik der Optimierung: Anlegen und variieren von externen Verschiebungen an das Optimierungsmodell (links), um die Referenz (rechts) zu erhalten 43
5.2 Grundprinzip des Optimierungsprozesses 44

6.1 Slope Deviation Plot (links) und z-Deviation Plot (rechts) des Padmodells von Testproblem 1 59
6.2 Slope Deviation Plot (links) und z-Deviation Plot (rechts) des Klebstoffmodells von Testproblem 1 59

6.3 Slope Deviation Plot (links) und z-Deviation Plot (rechts) von Testproblem 2 . . 60
6.4 Slope Deviation Plot in x (links) und z-Deviation Plot (rechts) des sechsdimensionalen Testproblems 60
6.5 Slope Deviation Plot (links) und z-Deviation Plot (rechts) des 12-dimensionalen Testproblems 61
6.6 Slope Deviation Plot (links) und z-Deviation Plot (rechts) des 20-dimensionalen Testproblems 62
6.7 Slope Deviation Plot in x (links) und z-Deviation Plot (rechts) der Spiegelfacette mit Laborunterstruktur 63
6.8 Slope Deviation Plot in x (links) und z-Deviation Plot (rechts) des RP3 Innenspiegels 63

7.1 Konvergenz von BOBYQA für verschiedene Diskretisierungen mit $\rho_{beg} = 0.01$ . . 68
7.2 Konvergenz von BOBYQA für verschiedene Diskretisierungen mit $\rho_{beg} = 1.0$ . . . 68
7.3 Konvergenz von BOBYQA für verschiedene Diskretisierungen mit $\rho_{beg} = 2.0$ . . . 69
7.4 Konvergenz von CMA-ES für verschiedene Diskretisierungen mit $\sigma = 1.0$ 70
7.5 Konvergenz von CMA-ES mit $\sigma = 0.01$ 71
7.6 Konvergenz von CMA-ES mit $\sigma = 1.0$ 72
7.7 Konvergenz von CMA-ES mit $\sigma = 2.5$ 72
7.8 Konvergenz von BOBYQA und CMA-ES 73
7.9 Slope Deviation Plots des Klebstoffmodells (links) und des Padmodells (rechts) . 74
7.10 Konvergenz von BOBYQA für das Klebstoff- und Padmodell 76
7.11 Konvergenz von CMA-ES für das Klebstoff- und Padmodell 76
7.12 Konvergenz von BOBYQA und CMA-ES 77
7.13 Konvergenz von BOBYQA und CMA-ES 80
7.14 Konvergenz von BOBYQA und CMA-ES 80
7.15 Konvergenz von BOBYQA für verschiedene Startschrittweiten 81
7.16 Konvergenz von CMA-ES für verschiedene Populationsgrößen 82
7.17 Konvergenz von CMA-ES für verschiedene Startschrittweiten σ 82
7.18 Konvergenz von CMA-ES für die +-Strategie 83
7.19 Konvergenz von BOBYQA und CMA-ES 83
7.20 Konvergenz der Zwischenergebnisse des Hybridalgorithmus 85
7.21 Plots der Slope Deviation in x (links) und y (rechts) 86
7.22 Plots der z-Deviation in $\hat{x}$ (links) und $\tilde{x}$ (rechts) 87
7.23 Plots der z-Deviation von Optimierungsmodell (links) und Referenz (rechts) . . . 88
7.24 Konvergenz für die z-Deviation (links) und zugehöriger Slope Deviation Plot in x (rechts) Optimierung des 12-dimensionalen Testproblem 88
7.25 Konvergenz für das 12-dimensionale Testproblem mit unterschiedlicher Parameteranzahl 89
7.26 Konvergenz für das Coordinate-Descent Verfahren und Optimierungslauf mit 20 Parametern 91
7.27 Konvergenz für das 20-dimensionale Testproblem mit unterschiedlicher Parameteranzahl 92
7.28 Konvergenz mit und ohne Wichtung der Zielfunktion 93

7.29 Plot der lokalen Differenzen der Slope Deviation in x mit gewichteter (links) und ungewichteter (rechts) Zielfunktion (in mrad) 94
7.30 Konvergenz für Rotationsparameter rad und mrad 95
7.31 Konvergenz für verschiedene Wahl von Optimierungsparameter 95
7.32 Plot der lokalen Differenzen der Slope Deviation in x von der Optimierung in rad (links oben), Optimierung mit 21 Parametern (rechts oben) und mit 24 Parametern (unten) 96
7.33 Konvergenz für unterschiedliche finale Trust-Region-Radien 97
7.34 Slope Deviation Plots des Optimierungsmodells (links) und herstellungsbedingte Formabweichungen (rechts) 98
7.35 Plot der Slope Deviation Differenzen der Lösung des Optimierungsmodells in x und y (in mrad) 100
7.36 Plot der Slope Deviation Differenzen der Lösung des Optimierungsmodells mit Multilevelansatz in x und y (in mrad) 101
7.37 Konvergenz von BOBYQA für die reale Messung und der Multilevelvariante . . . 102

A.1 Konvergenz für das Coordinate-Descent Verfahren in rad 122
A.2 Prozentualer Fehler unterschiedlicher Netzwqualitäten [Kle 11] 122
A.3 Slope Deviation Plots des Optimierungsmodells (links) und herstellungsbedingte Formabweichungen (rechts) der Multileveloptimierung 123

Liste der Algorithmen

3.1 BOBYQA Algorithmus 26

3.2 Coordinate-Descent Methode [Tse 01] 30

5.1 Funktionsauswertung.m 48

5.2 CoordinateDescent.m 54

5.3 Funktionsauswertung.m 54

A.1 Basis Trust-Region Algorithmus [Alt 11] 110

A.2 CMA-ES Algorithmus 111

Tabellenverzeichnis

4.1 Positionen der Padkoordinatensysteme des RP3 Innenspiegels [Mei 13] 36
4.2 Gravitationskomponenten für verschiedene Spiegelwinkel des Innenspiegel rechts vom Scheitelpunkt [Mei 13] 38

6.1 Translationen der Pads für die Laborunterstruktur 62

7.1 Slope Deviation Plots von verschiedene Diskretisierungen des dreidimensionalen Testproblems 67
7.2 Resultate von Testproblem 1 für BOBYQA und CMA-ES 74
7.3 Resultate von Testproblem 2 für BOBYQA und CMA-ES 77
7.4 Translationen und Rotationen der Pads 98
7.5 Slope Deviation Plots des Optimierungsmodells (links) und der Messung (rechts) 99
7.6 Slope Deviation Plots des Optimierungsmodells (links) und der Messung (rechts) der Multilevelvariante 101

A.1 Resultate für BOBYQA mit $\rho_{beg} = 0.01$ 113
A.2 Resultate für BOBYQA mit $\rho_{beg} = 1.0$ 113
A.3 Resultate für BOBYQA mit $\rho_{beg} = 2.0$ 113
A.4 Resultate des CMA-ES Algorithmus für verschiedene Diskretisierungen 114
A.5 Resultate für CMA-ES mit $\sigma = 0.01$ 114
A.6 Resultate für CMA-ES mit $\sigma = 1.0$ 114
A.7 Resultate für CMA-ES mit $\sigma = 2.5$ 114
A.8 Resultate des BOBYQA Algorithmus von Klebstoff- und Padmodell 114
A.9 Resultate des CMA-ES Algorithmus von Klebstoff- und Padmodell 115
A.10 Resultate von Test 1, sechsdimensionales Testproblem für BOBYQA und CMA-ES115
A.11 Resultate von Test 2, sechsdimensionales Testproblem für BOBYQA und CMA-ES115
A.12 Resultate von Test 3, sechsdimensionales Testproblem für BOBYQA und CMA-ES116
A.13 Resultate der z-Deviation Optimierung 116
A.14 Resultate vom Hybridalgorithmus, 12-dimensionale Testproblem 117
A.15 Slope Deviation Plots der Zwischenlösungen des Hybridalgorithmus 118
A.16 Slope Deviation Differenzen Plots der Zwischenlösungen des Hybridalgorithmus . 119
A.17 Resultate der Parameterreduktion, 12-dimensionale Testproblem 120
A.18 Resultate der Parameterreduktion, 20-dimensionale Testproblem 120
A.19 Resultate der Coordinate-Descent Methode, 20-dimensionale Testproblem 121
A.20 Translationen und Rotationen der Pads 123

1 Einleitung

Die steigende Weltbevölkerungszahl, sowie die wachsende Wirtschaftskraft sorgen für einen erhöhten weltweiten Energiebedarf. Des Weiteren neigen sich die fossilen Ressourcen in den nächsten Jahrzehnten langsam dem Ende zu. Treibhausgas- und Schadstoffausstöße gelangen vor allem durch die Verbrennung der fossilen Brennstoffe Kohle, Gas und Öl in die Atmosphäre. Die Emission von Treibhausgasen begünstigt den anthropogenen Treibhauseffekt und führt dadurch zu einer Veränderung des globalen Klimasystems. Ca. 85 % des Weltprimärenergiebedarfs werden von fossilen Energieträgern gedeckt (siehe [Qua 09]). Die Ressourcenknappheit, wie auch die drohenden Ausmaße des Klimawandels, führen auf lange Sicht hin zu einem Umstieg auf alternative CO_2-neutrale Energiequellen. Unter erneuerbarer oder auch regenerativer Energie werden alle Energiearten zusammengefasst, die unter menschlichem Zeitrahmen unerschöpflich sind (vgl. [Qua 09]). Dazu zählen neben Solar- auch Windenergie, Wasserkraft, Geothermie und Biokraftstoffe. Ein weiterer Vorteil regenerativer Energieträger ist die Unabhängigkeit von instabilen Märkten für fossile Brennstoffe. Die EU hat den Weg zu erneuerbaren und sauberen Energiequellen bereits eingeschlagen. In der Richtlinie zur Förderung der Nutzung von Energie aus erneuerbaren Quellen vom April 2009 verpflichtet sich die EU bis zum Jahr 2020 mindestens 20 % des gesamten Energieverbrauches mit erneuerbaren Energiequellen zu decken (vgl. [EU 09]).

Großes Potenzial bietet hier die Nutzung der Sonnenenergie. Die Erde erreicht jährlich eine Energiemenge von $3.9 \cdot 10^{24}$ J an Sonnenenergie. Damit liegt das Strahlungsangebot der Sonne bei ungefähr dem 10 000-fachen des weltweiten Primärenergieverbrauchs und bietet folglich ein großes Potential für die Energieversorgung der Zukunft ([Qua 09]). Eine Nutzungsmöglichkeit der Sonnenenergie stellen solarthermische Kraftwerke dar. Hier wird die Solarstrahlung mit Hilfe von Spiegeln gebündelt und dadurch ein Wärmemedium erhitzt. Dieses treibt eine konventionelle Turbine an, welche Strom erzeugt. Zu den großen Vorteilen dieser Technologie zählt die Möglichkeit der Speicherung der Wärmeenergie. So ist es auch bei reduzierter Sonneneinstrahlung oder nach Sonnenuntergang möglich das Kraftwerk weiter zu betreiben und damit eine konstante Stromversorgung zu ermöglichen. In Gegenden mit hoher Direktsolareinstrahlung wie im Süden Spaniens und in Nordafrika, sowie im Nahen Osten, können solarthermische Kraftwerke einen großen Teil zur Energieerzeugung beisteuern. Solarthermische Kraftwerke müssen weiter optimiert werden, um diese wettbewerbsfähig gegenüber anderen Technologien zu machen. Der technisch ausgereifteste und kommerziell betriebene Kraftwerkstyp ist das Parabolrinnenkraftwerk. Hier bestehen die Spiegelkomponenten aus parabolisch geformten Spiegelfacetten. Die Spiegel allein sind für 15 % der Investitionskosten des Solarfeldes verantwortlich (siehe [Nav 09]). Die Formgenauigkeit der Spiegel hat einen wesentlichen Einfluss auf den Ertrag eines ganzen Kraftwerkes. Weicht die

Spiegelgeometrie von der idealen Parabelform ab, trifft die Solarstrahlung das Absorberrohr, in dem sich das Wärmemedium befindet, gegebenenfalls nicht mehr. Dies hat einen Verlust an nutzbarer Leistung zur Folge. Eine um 1 % geringere optische Qualität der Spiegel führt zu jährlichen Verlusten von ca. 0.5 Mio. US$ für den Kraftwerksbetreiber ([Hof 14]). Aufgrund dessen werden hohe Qualitätsansprüche bei der Herstellung der Spiegel und deren Montage im Feld gestellt. Zu Abweichungen von der idealen Parabelform führt auch die Eigenlast der Spiegel und auftretende Reaktionskräfte an den Spiegelaufhängungen, welche Spiegel und Kollektorunterstruktur verbinden. Folglich spielt ein umfassendes, detailliertes Wissen über die Verformungseigenschaften der Spiegel eine wichtige Rolle. Am Institut für Solarforschung des Deutschen Zentrums für Luft- und Raumfahrt (DLR) in der Abteilung Qualifizierung werden Prüfmethoden zur Beurteilung der Einflüsse und Effekte auf die Leistungsfähigkeit solarthermischer Komponenten entwickelt. Zur Untersuchung der Faktoren, welche die Formgenauigkeit und mögliche Verformungen einer Spiegelfacette beeinflussen, werden am DLR verschiedene Methoden verwendet. Zum einen ist mit Hilfe der Deflektometrie eine hochgenaue Vermessung der Formabweichung eines Spiegels möglich. Dies ist durch die Bestimmung der Steigungsabweichung der Oberflächennormalenvektoren auf der ganzen Spiegelfacette durchführbar. Ferner werden Finite-Elemente Modelle von Parabolrinnenspiegeln erzeugt und strukturmechanische Analysen mit der Simulationssoftware Ansys durchgeführt. Auch hier besteht die Option die Steigungsabweichung der Normalenvektoren mittels Matlab zu bestimmen. Die bereits konstruierten Spiegelmodelle sowie die Berechnungen in Matlab kommen auch in dieser Arbeit zur Anwendung.

Das Ziel dieser Masterarbeit ist Fehlstellungen der Spiegelaufhängungen, welche Reaktionskräfte verursachen, aufzuzeigen, um die Formgenauigkeit der Spiegel im Solarfeld zu optimieren. Aufgrund dessen kommen hier Optimierungsalgorithmen zur Bestimmung von Verformungsursachen der Spiegelfacetten zum Einsatz, welche Rückschlüsse von Spiegelformmessungen auf die ursächlichen Verschiebungen und Verkippungen an den Spiegelaufhängungen ziehen lassen. Es dient ein bereits vorhandenes unverformtes Finite-Elemente Spiegelmodell als Optimierungsmodell, welches so lange verformt wird bis es mit einer gegebenen deformierten Spiegelfacette, der Referenz, übereinstimmt. Den Algorithmen werden dabei die Verschiebungen und Rotationen an den vier Spiegelaufhängepunkten in allen drei Raumdimensionen als Optimierungsparameter übergeben. Somit ergeben sich insgesamt 24 freie Parameter für die Optimierung. Damit können in strukturmechanischen Simulationen die Translationen und Rotationen an den Spiegelaufhängungen der Spiegelmodelle angelegt werden und durch die Finite-Elemente Analyse resultierende Verformungen des Spiegels berechnet werden. Anschließend folgt ein Vergleich mit der Referenz. Dies geschieht durch die Bestimmung des quadratischen Mittels der Differenzen der lokalen Steigungsabweichungen der Oberflächennormalenvektoren von Optimierungs- und Referenzmodell, welche die Zielfunktion für die Optimierung darstellt. Ein Problem bei der Optimierung ist durch den hohen Zeitaufwand für die Berechnung einer Funktionsauswertung gegeben. Des Weiteren ist durch die Auswertung der Spiegelform in Ansys die analytische Form der Zielfunktion unklar und muss daher als Black-Box-Funktion betrachtet werden. Darüber hinaus stehen durch die Verwendung von Ansys keine Ableitungen zur Verfügung. Aufgrund dessen fällt die Wahl in dieser Arbeit auf die Verwendung von ableitungsfreien Optimierungsverfahren. Diese Schwierigkeiten sowie die hohe Problemdimension verlangen nach effizienten und robusten Optimierungsverfahren.

Die Masterarbeit ist folgendermaßen gegliedert: Im zweiten Kapitel werden die solarthermischen Grundlagen, welche zum weiteren Verständnis dieser Arbeit dienen, erläutert. Dazu zählt der allgemeine Aufbau eines Parabolrinnenkraftwerks und die Beschaffenheit der Spiegelkomponenten. Es werden außerdem Gütekriterien für die Parabolrinnenkollektoren, wie der Intercept-Faktor und die Slope Deviation, die auch als Steigungsabweichung der Oberflächennormalenvektoren bekannt ist, aufgezeigt. Im dritten Kapitel geht es um die mathematischen Grundlagen dieser Arbeit. Es wird die lineare Elastizitätstheorie eingeführt und im Zusammenhang mit der Finite-Elemente Methode eine numerische Lösung für gegebene partielle Differenzialgleichung gefunden. Nachfolgend werden fundamentale Grundlagen aus dem Bereich der Optimierung vorgestellt. Die Algorithmen BOBYQA von Powell (siehe [Pow 09]) und CMA-ES von Hansen (vgl. [HO 96]), welche auf die Problemstellung angewendet werden, werden in diesem Kapitel vorgestellt. Das Coordinate Descent Verfahren und der Multilevel-Ansatz werden anschließend eingeführt. Kapitel 4 beschäftigt sich mit den Fundamenten der Deflektometrie, welche verwendet wird, um die Formabweichungen eines Spiegels zu bestimmen. Darüber hinaus werden die bereits vorhandenen Finite-Elemente Spiegelmodelle, welche in dieser Arbeit verwendet werden, mit den gemachten Annahmen und Vereinfachungen vorgestellt. In Kapitel 5 wird die Vorgehensweise des Optimierungsprozesses präzisiert. Des Weiteren wird auf die Wahl der hier verwendeten Optimierungsalgorithmen eingegangen. Außerdem wird eine Diskretisierungsanalyse, eine Sensitivitätsanalyse bezüglich der Startschrittweite sowie des Startwertes durchgeführt. Zusätzlich kommt eine Variante des Coordinate Descent Verfahrens und des Multilevel-Ansatzes zum Einsatz, welche in diesem Abschnitt in Bezug auf die Problemstellung erläutert werden und ein Verbesserung der Optimierung zum Ziel haben. In Kapitel 6 werden Testprobleme konstruiert, sodass es möglich ist die Algorithmen zu untersuchen und Aussagen über ihre Exaktheit und Laufzeit zu machen. Die verschiedenen Versionen der verwendeten Algorithmen werden auf die konstruierten Testprobleme, welche verformte Spiegelmodelle darstellen, angewendet und die Ergebnisse in Kapitel 7 vorgestellt. Des Weiteren findet eine Anwendung des Algorithmus BOBYQA auf einen real vermessenen Spiegel statt. Das Resultat wird ebenfalls in diesem Kapitel dargestellt. Zuletzt folgt in Kapitel 8 eine Zusammenfassung der Hauptresultate dieser Arbeit und ein Ausblick bezüglich weiterer Untersuchungen und Entwicklungsmöglichkeiten.

2 Solarthermische Grundlagen

Der Begriff der Solarthermie beinhaltet alle Bereiche der thermischen Nutzung von solarer Strahlung. Die Solarthermie zeichnet sich dadurch aus, dass thermische Energie durch die Fokussierung der Solarstrahlung zur Gewinnung von elektrischem Strom genutzt wird. Die Anwendungsgebiete der solarthermischen Stromerzeugung sind die solare Trinkwasseraufbereitung, Prozesswärmebereitstellung und solarthermische Kraftwerke (vergleiche [WSLF 13]). Der Vorteil von solarthermischen Kraftwerken gegenüber Windkraftanlagen oder Fotovoltaikanlagen ist, dass die Wärme kostengünstig und umweltfreundlich gespeichert werden kann, sodass auch nach Sonnenuntergang oder bei schlechten Wetterverhältnissen eine konstantere Stromversorgung erreicht werden kann (siehe [Scho 13]). Bei solarthermischen Kraftwerken, auch CSP-Kraftwerke (CSP = Concentrated Solar Power) genannt, wird zwischen punkt- und linienfokussierenden Systemen unterschieden. Zu den punktfokussierenden Systemen zählt beispielsweise das Turmkraftwerk. Hier wird die Solarstrahlung auf einen Punkt am Turm, an dem sich der Receiver befindet, durch Spiegel fokussiert. Im Gegensatz dazu gehören Fresnel- und Parabolrinnenkraftwerke zu den linienfokussierenden Systemen. Ein Fresnelkollektor wird aus planaren Spiegeln zusammengesetzt, wobei Parabolrinnenkollektoren hingegen die Form eines halben parabolischen Zylinders haben. In beiden Fällen werden die Sonnenstrahlen auf ein Absorberrohr gebündelt. Eine Übersicht bietet [Qua 09]. Von den kommerziell betriebenen solarthermischen Kraftwerksanlagen sind aktuell mehr als 95 % Parabolrinnenkraftwerke (vgl. [PHHB 13]).

2.1 Parabolrinnentechnologie

Zunächst wird auf den allgemeinen Aufbau und die Funktionsweise von Parabolrinnenkraftwerken eingegangen. Anschließend werden die Solarkollektoren, die Hauptkomponenten solarthermischer Anlagen, welche für die Umwandlung von elektromagnetischer Strahlenenergie in thermische Energie verantwortlich sind, näher erläutert.

2.1.1 Parabolrinnenkraftwerke

Die Parabolrinnentechnologie wurde bereits im Jahr 1907 patentiert. In den folgenden Jahren wurden Demonstrationskraftwerke in den USA sowie Ägypten erbaut. Allerdings wurde der Ansatz der Stromerzeugung aufgrund von technischen Schwierigkeiten sowie Materialproblemen

und durch das Finden neuer Ölquellen nicht weiter verfolgt. Die Ölkrise der 70 er-Jahre führte jedoch zu einem Umdenken, sodass nach alternativen Energieerzeugern gesucht wurde und diese auch subventioniert wurden. So entstand im Jahr 1984 das erste kommerzielle solarthermische Parabolrinnenkraftwerk in der kalifornischen Mojave-Wüste. Es wurde auch das Versuchskraftwerk Plataforma Solar de Almería in Spanien errichtet, welches heute Europas größtes Testzentrum für Solartechnik ist. Jedoch führten fallende Energiepreise Mitte der 1980 er-Jahre dazu, dass staatliche Förderprogramme gekürzt bzw. komplett gestrichen wurden. Infolge dessen wurde diese Technologie zunächst nicht weiter verfolgt. Angeregt durch den Gedanken des Klimawandels begünstigten steuerliche Vergütungen beispielsweise in Spanien, Deutschland und den USA Anfang des neuen Jahrtausends den Ausbau von erneuerbarer und damit CO_2-freier Energie. [Gey et al. 02] [Qua 09] [Pit 12]

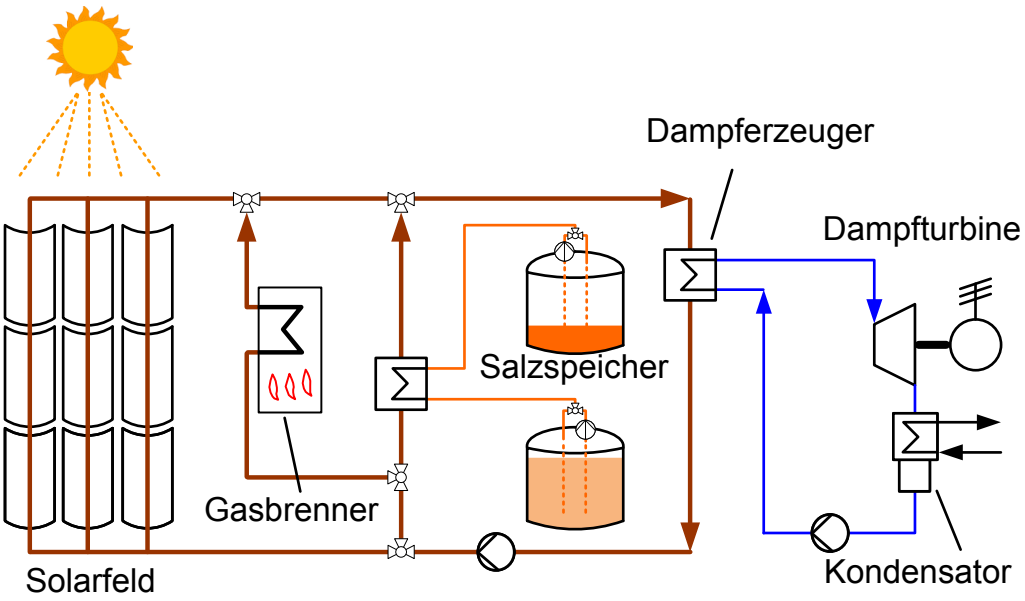


Abbildung 2.1: Prinzip eines Parabolrinnenkraftwerks mit Salzspeicher und getrenntem Kollektor- und Dampfturbinenkreis [BG 10]

Das Parabolrinnenkraftwerk besteht aus einem Kollektorfeld, auch Solarfeld genannt, dem Kollektor- und dem Dampfturbinenkreislauf mit einem Dampferzeuger und dem Kraftwerksblock. Das Kollektorfeld besteht aus in Reihe geschalteten Sonnenkollektormodulen. Manche Anlagen besitzen zusätzlich Wärmespeichersysteme. Als Wärmespeicher kann beispielsweise Salz dienen. Das Schaltbild eines solchen Parabolrinnenkraftwerks mit Speicher ist in Abbildung 2.1 dargestellt. Im Folgenden werden beispielhaft einige Daten zum 50 MW_{el} Andasol Kraftwerk im Süden Spaniens aufgezeigt. Es besitzt eine Grundfläche von ungefähr 2 km^2. Das Kollektorfeld besteht aus 300 Reihen parallel angeordneter Kollektoren und weist eine Spiegelfläche von ca. 510 000 m^2 auf (vgl. [Ra et al. 04]). Dazu sind 7 488 Konzentratormodule, auf denen 210 000 Spiegelfacetten an 840 000 Spiegelmontagepunkte befestigt sind, nötig (vgl. [Lü et al. 07b]).

Ein Kollektormodul besteht aus einem Receiver, in dem sich das Wärmeträgerfluid befindet und den Reflektorspiegeln, die dafür sorgen, dass die Sonnenstrahlen den Receiver treffen und so das Wärmeträgerfluid erhitzen. Des Weiteren gehören zu den Komponenten des Moduls ein

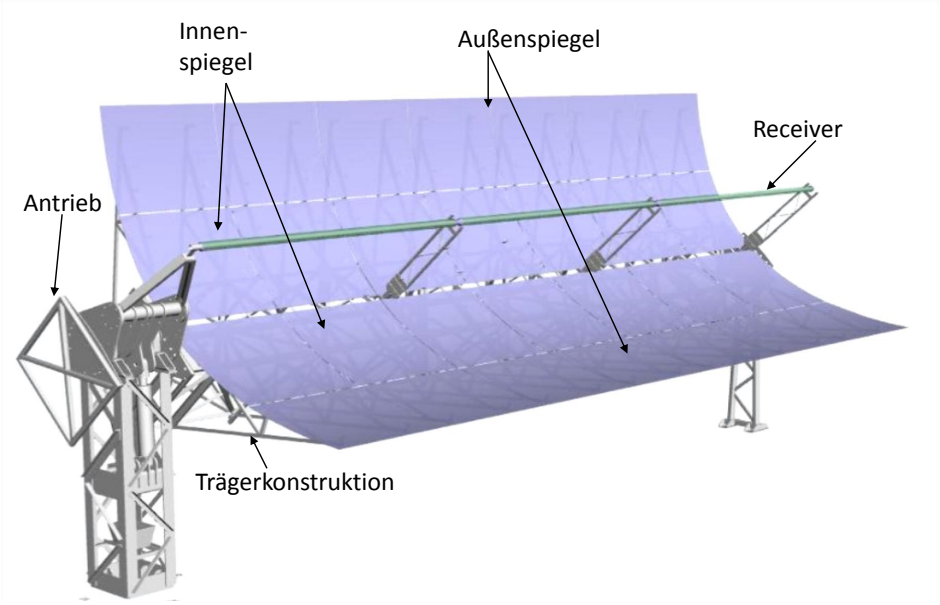


Abbildung 2.2: EuroTrough-Kollektormodul [Schi 11]

Antrieb, welcher dafür sorgt, dass das Modul der Sonne nachgeführt wird und die Kollektorunterkonstruktion, auf welcher die Spiegel montiert sind. Die heutigen Kollektormodule, wie beispielsweise der EuroTrough, welcher in Abbildung 2.2 dargestellt ist, besitzen vier gekrümmte Glasspiegel, welche zusammen die Parabel formen. Es wird zwischen den beiden Außen- und den zwei Innenspiegeln der Kollektoren unterschieden. Auf der Stahlunterkonstruktion eines Kollektormoduls sind 28 Spiegelfacetten befestigt. [RR 13]

Die Parabel bietet die ideale Form für die Spiegel, da die Strahlung, welche parallel zur optischen Achse auf die parabolisch geformten Spiegel einfällt, im Brennpunkt gebündelt wird, wie Abbildung 2.3 verdeutlicht. Die Sonnenstrahlen werden linienförmig von den Parabolspiegeln auf einen Receiver in Form eines Absorberrohres konzentriert ([Gün 15]). Es ist zu beachten, dass Parabolrinnenkraftwerke nur direkte Solarstrahlung nutzen, da nicht-gerichtete Strahlen nicht fokussiert werden können.

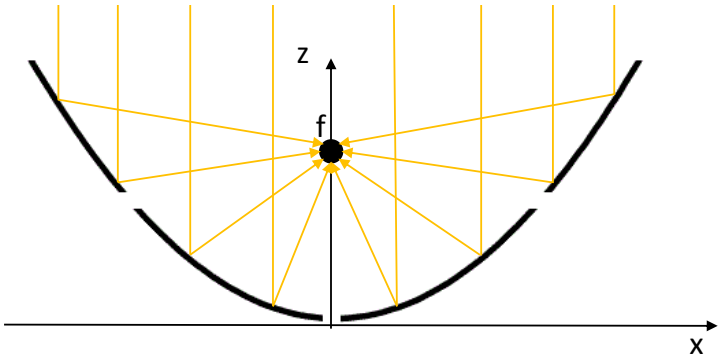


Abbildung 2.3: Funktionsprinzip der Parabolrinne

Das Absorberrohr besteht aus einem Stahlrohr, durch das ein Wärmeträgermedium, meist ein Thermoöl, fließt. Das Stahlrohr wird durch ein vakuumisoliertes Glashüllrohr umschlossen, welches vor konvektiven Wärmeverlusten schützt. Es wird durch die Strahlungseinwirkung auf bis zu 400 °C aufgeheizt. Diese Temperatur wird unter anderem durch das Thermoöl festgelegt, da sich dieses bei höheren Temperaturen zersetzt. Die durch die Einstrahlung aufgenommene thermische Energie wird anschließend durch einen Wärmeübertrager an den Dampfkreislauf übertragen. Der so erzeugte Wasserdampf treibt nun eine Dampfturbine und einen Stromgenerator an. Durch den Einbau eines Wärmespeichers ist es möglich die thermische Energie beispielsweise zur Erhitzung von flüssigem Salz zu verwenden, welches fähig ist die zugeführte Wärme über mehrere Wochen hinweg zu speichern. Bei Bedarf wird die Wärme aus dem Salz wieder zur Stromerzeugung genutzt. [WSLF 13] [Qua 09]

Die Spiegelrinnenkollektoren werden einachsig der Sonne nachgeführt. So lässt sich die direkte Solarstrahlung um das 80-fache konzentrieren. Eine zweiachsige Nachführung wird in den meisten Fällen aus Kostengründen abgelehnt. Jedoch ergeben sich durch die einachsige Sonnennachführung Kosinus-Verluste. Das heißt Sonnenstrahlen fallen vermehrt nicht im richtigen Winkel ein, sodass es zu Konzentrationsverlusten kommt. [Wat 09]

2.1.2 Parabolrinnenkollektoren

Die Module des EuroTrough, einem weit verbreiteten Kollektortypen, bestehen aus vier einzelnen gekrümmten Glasspiegeln, welche zusammen die Parabelform erzeugen. Es wird ein Koordinatensystem durch die Spiegel gelegt, dessen Ursprung im Scheitelpunkt der Parabel liegt, wie bereits in Abbildung 2.3 dargestellt. Per Definition zeigt die z-Achse vom Scheitelpunkt der Parabel zum Fokalpunkt, in dem die Solarstrahlung fokussiert wird. Die y-Achse verläuft entlang der Symmetrieachse der Parabel und die x-Achse bezieht sich auf die gekrümmte Spiegelseite. Die Parabelgleichung, wobei f den Abstand vom Ursprung zum Fokalpunkt beschreiben, lautet:

$$z = \frac{x^2}{4f}. \tag{2.1}$$

Beim EuroTrough-Kollektor beträgt die Fokallänge f =1710 mm (vgl. [Pit 12]). Der Innenspiegel der Firma Flabeg mit der Designbezeichnung RP3 (reflector panel 3) misst in x-Richtung 1600 mm und in der ungekrümmten y-Richtung 1700 mm. Der Außenspiegel hat in x-Richtung eine Länge von 1500 mm. [Pit 12]

Die parabolisch geformten Spiegel werden aus Flachglasscheiben gefertigt. Auf der Rückseite der Spiegel wird eine reflektierende Silberschicht aufgetragen. Um diese vor Korrosion und Witterungseinflüssen zu schützen wird zusätzlich eine Kupferschicht und drei weitere Lackschichten aufgebracht. Auch das 4 mm dicke Flachglas dient als Wetterschutz für den spiegelnden Silberfilm. Dadurch ist die Reflektivität dieser Spiegel auch nach 20 Betriebsjahren kaum reduziert. Es treten durchschnittlich 0.5 % an Reflektionsverlusten auf. Abbildung 2.4 zeigt den Aufbau eines Parabolrinnenspiegels mit den verschiedenen Schichten. Hier ist das Keramikpolster zu erkennen.

Auf jede Spiegelrückseite werden vier Keramikmontierungspolster, auch Pads genannt, hochgenau an den Spiegel mittels Klebstoff angebracht und durch Schrauben an die Trägerkonstruktion montiert. Die Pads dienen als Bindeglied zwischen dem Spiegel und der Unterkonstruktion. [Pit 12] [FG 09]

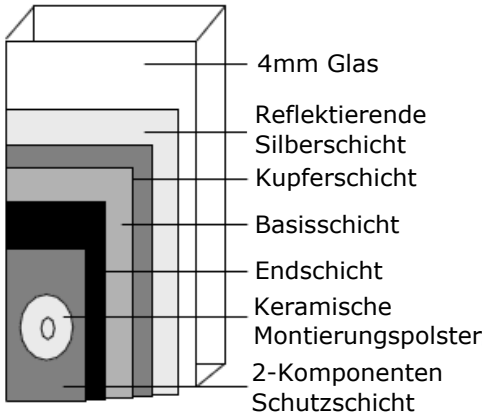


Abbildung 2.4: Aufbau eines Parabolrinnenspiegels [Pit 12]

2.2 Qualifizierung der Komponenten

Die Qualität der Komponenten in solarthermischen Parabolrinnenkraftwerken spielt eine große Rolle, um hohe Wirkungsgrade zu erreichen. Abweichungen von der idealen Spiegelform können einen großen Einfluss auf die Effizienz dieses Kraftwerkstyps haben. Auf Grund von Spiegelformabweichungen treffen die Sonnenstrahlen das Absorberrohr gegebenenfalls nicht mehr. Dadurch kann nutzbare Leistung verloren gehen, sodass der Jahresertrag eines Kraftwerkes verringert wird. Beispielsweise stellt ein 100 MW Parabolrinnenkraftwerk mit einer jährlichen Betriebszeit von 40 % pro Jahr 100 MW·3504 h = 350 400 MWh an Strom bereit. Nur 1 % Verlust beim Reflektionsgrad der Spiegel bedeutet Einbußen von 3 504 MWh im Jahr. Bei einer Einspeisevergütung von 0.14 US$/kWh folgen jährliche Verluste von 490 560 US$ (siehe [Hof 14]). Damit basiert die Leistungsfähigkeit des Kollektorfelds auf der Qualität der Kollektoren.

Für die Beschreibung der optischen Performance eines Parabolrinnenkollektors dient der Intercept-Faktor und als Qualitätskriterium für die Spiegelform wird die ortsaufgelöste Steigungsabweichung der Oberflächennormalenvektoren definiert, welche im Folgenden erörtert werden.

2.2.1 Intercept-Faktor

Die Leistungsfähigkeit eines Kollektors lässt sich mittels des optischen Wirkungsgrades wie folgt beschreiben:

$$\eta_{\text{opt}} = \rho_{\text{refl}} \cdot \tau_{\text{rec}} \cdot \alpha_{\text{rec}} \cdot \gamma, \tag{2.2}$$

wobei ρ_{refl} die Reflektivität des Spiegels, τ_{rec} die Transmissivität des Hüllrohrs, α_{rec} den Absorbtionsgrad des Absorberrohrs und γ den Intercept-Faktor beschreibt. Der Intercept-Faktor ist als Quotient aus dem Anteil eingestrahlter Leistung, welche das Absorberrohr trifft und der gesamten Strahlung, welche auf die Aperturebene einfällt, definiert. Er beinhaltet alle Verluste aufgrund von Spiegelfehlern sowie Fertigungstoleranzen, der Fehlstellung des Absorberrohrs und Tracking-Ungenauigkeiten. Ferner gehören die Einbußen durch die nicht ideale Form der Sonne zum Intercept-Faktor. In Abbildung 2.5 werden diese Faktoren veranschaulicht. [Pit 12]

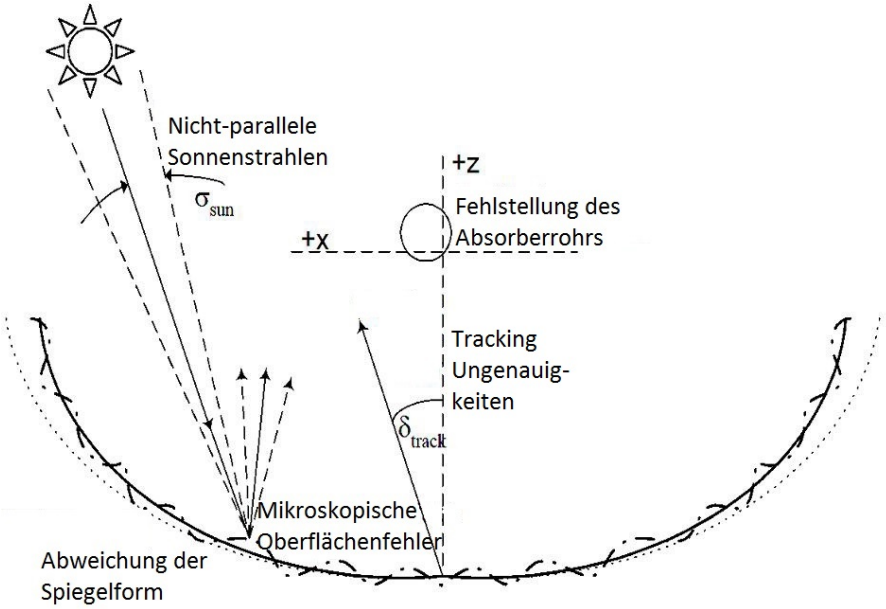


Abbildung 2.5: Optische Fehler und optische Verlustmechanismen bei Parabolrinnenkollektoren [Kis03]

2.2.2 Slope Deviation

Bei Spiegelformabweichungen von der idealen Parabelform spielen herstellungsbedingte Ungenauigkeiten, die Eigenlast und auftretende Reaktionskräfte an den Spiegelaufhängungen eine Rolle. Als Qualitätskriterium wird die ortsaufgelöste Steigungsabweichung der Oberflächennormalenvektoren, die so genannte Slope Deviation, von der jeweiligen idealen Richtung ermittelt. Wird der reale und der ideale Oberflächennormalenvektor n_{real} und n_{ideal} in die xz-Ebene abgebildet, dann kann die Steigungsabweichung in x-Richtung bestimmt werden. Dies wird in Abbildung 2.6 veranschaulicht. Dementsprechend wird die lokale Slope Deviation wie folgt definiert:

$$sdx = \beta - \alpha = \arctan\left(\frac{n_{real_x}}{n_{real_z}}\right) - \arctan\left(\frac{n_{ideal_x}}{n_{ideal_z}}\right). \tag{2.3}$$

Die Abweichung in y-Richtung, welche entlang der longitudinalen Spiegelrichtung verläuft, wird hier nicht weiter betrachtet, da durch Winkelabweichungen in y-Richtung trotzdem gewährleistet

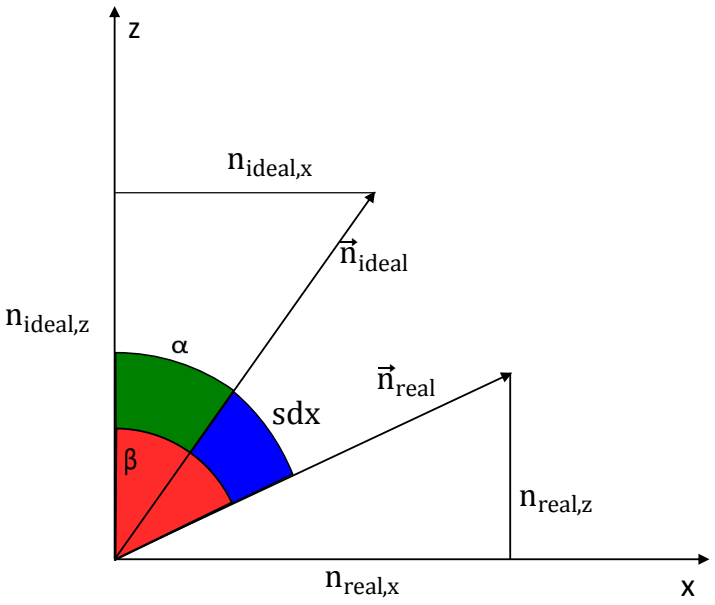


Abbildung 2.6: Berechnung der Slope Deviation in x-Richtung durch Projektion der Normalenvektoren in die xz-Ebene

ist, dass die Sonnenstrahlen das Absorberrohr treffen. Sie wird jedoch analog zur Slope Deviation in x ermittelt. Eine Rotation des realen Normalenvektors nach außen im Vergleich zum idealen Normalenvektor wird mit einer positiven Slope Deviation beschrieben. Eine Rotation nach innen ist negativ definiert. Per Definition zeigen Rotationen nach außen zu den Rändern der Parabolrinne, Rotationen nach innen zum Zentrum der Parabel. Dies wird in Abbildung 2.7 veranschaulicht. Die gelben Pfeile stellen die Solarstrahlung, sowie die Reflektion dieser an der Spiegelfacette dar. Die durchgezogene gelbe Linie nach der Reflexion zeigt den Strahlengang für einen idealen Spiegel, während die gestrichelte gelbe Linie die Solarstrahlung für einen verformten Spiegel illustriert. Hier wird auch deutlich, dass sich der Winkelfehler durch die

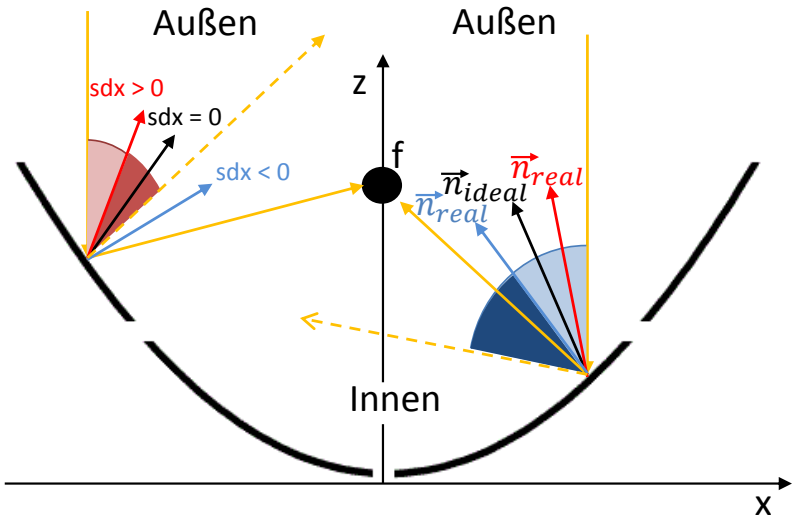


Abbildung 2.7: Strahlenverlauf für ideale und reale Oberflächennormalenvektoren

Reflexion verdoppelt. Damit können minimale Spiegelformfehler in der Größenordnung einiger mrad Auswirkung auf die Effizienz eines Kollektors haben.

Der flächengewichtete Mittelwert der lokalen Slope Deviation-Werte, die Slope Deviation SDx, beschreibt prägnant die Formgenauigkeit der ganzen Spiegelfacette. Für die Berechnung der Slope Deviation wird die lokale Slope Deviation an p Punkten der Spiegelfacette ausgewertet und ein gewichteter Mittelwert bestimmt:

$$\text{SDx} := \sqrt{\sum_{k=1}^{p} \left(sdx_k^2 \frac{a_k}{A_{tot}} \right)}, \tag{2.4}$$

wobei sdx_k der lokale Slope Deviation-Wert an der Stelle k darstellt, a_k das in die Aperturebene projizierte Oberflächenelement und A_{tot} die totale Aperturfläche der Facette (vgl. [Wri 15]).

Der Vergleich von zwei Spiegeln mittels Slope Deviation in x erfolgt durch das quadratischen Mittel der Differenzen der lokalen Slope Deviations der beiden Spiegel. Der RMS (Root Mean Square) der Steigungsabweichungsdifferenzen wird folgendermaßen berechnet:

$$\triangle\text{SDx} := \sqrt{\sum_{k=1}^{p} \left((sdx_{2_k} - sdx_{1_k})^2 \frac{a_k}{A_{tot}} \right)}, \tag{2.5}$$

mit den lokalen Steigungsabweichungen der beiden Spiegelfacetten sdx_1 und sdx_2 sowie mit den gleichen Oberflächenelementen a_k für beide Spiegelfacetten (siehe [Wri 15]). Das Kriterium wird in dieser Arbeit als Zielfunktion für die Optimierung gewählt. Damit stellt sdx_{2_k}, $k = 1, ..., p$, die lokalen Slope Deviation-Werte der durch Ansys simulierten Spiegelfacette, welche als Optimierungsmodell dient und den zu optimierenden Spiegel nachahmt, dar. Weiterhin repräsentieren sdx_{1_k}, $k = 1, ..., p$, die lokalen Slope Deviation-Werte der Referenz.

3 Mathematische Grundlagen

In diesem Kapitel werden für diese Arbeit relevante mathematische Themen behandelt. Die physikalischen Eigenschaften des Spiegelmodells liegen der linearen Elastizitätstheorie zugrunde und das Spiegelmodell in dieser Arbeit wird durch Finite-Elemente diskretisiert. Aufgrund dessen wird zunächst auf die lineare Elastizitätstheorie eingegangen und mit Hilfe dieser das lineare Gleichungssystem aufgestellt, welches durch den Einsatz der Finite-Elemente Methode zustande kommt. Anschließend werden Grundlagen der Optimierung behandelt. Der evolutionäre Algorithmus CMA-ES und die ableitungsfreie Trust-Region Methode BOBYQA werden gewählt, welche nachfolgend erläutert werden. Im Anschluss wird auf die Coordinate-Descent Methode und den Multilevelansatz eingegangen, welche für die Verbesserung des Optimierungsvorgangs eingesetzt werden.

3.1 Finite-Elemente Methode

Probleme aus der Mechanik können in Form von partiellen Differentialgleichungen ausgedrückt werden. Diese sind jedoch nicht immer analytisch lösbar. Deshalb wird auf Approximationstechniken wie beispielsweise die Finite-Elemente (FE) Methode zurückgegriffen. Das Ziel ist es, nicht oder nur schwer lösbare Differentialgleichungen so umzuwandeln, dass das Problem in ein lineares bzw. nichtlineares Gleichungssystem transformiert wird. Die Grundidee des Finite-Elemente Ansatzes besteht aus einer Unterteilung des Gebietes, auf der die Differentialgleichung gelöst werden. Die endlich vielen Teilgebiete werden auch Elemente genannt und es werden Basisfunktionen gewählt, welche auf den meisten Elementen gleich Null sind. Anschließend werden die Funktionen in die Differentialgleichungen eingesetzt, sodass ein Gleichungssystem entsteht, welches nun lösbar ist. Mit Hilfe der linearen Elastizitätstheorie lassen sich Spannungs- und Deformationseigenschaften, der hier verwendeten physikalischen Strukturen, beschreiben.

Die folgenden Abschnitte orientieren sich soweit nicht anders vermerkt an [GHW 14] und an [KS 09].

3.1.1 Lineare Elastizitätstheorie

Sei σ_{ij} der ij-te Eintrag des symmetrischen Spannungstensors $\sigma \in \mathbb{R}^{3\times3}$, ϵ_{ij} der ij-te Eintrag des symmetrischen Dehnungstensors $\epsilon \in \mathbb{R}^{3\times3}$ und u_i die i-te Komponente des Verschiebungsvektors $u \in \mathbb{R}^3$ mit $1 \leq i, j \leq 3$. Der gesuchte Verschiebungsvektor, Spannung- und Dehnungstensor werden mittels geltender Grundgleichungen der linearen Elastizitätstheorie beschrieben. Diese Grundgleichungen sind die Gleichgewichtsbedingung, die Verschiebungs-Dehnungsgleichung und die Spannungs-Dehnungsgleichung, welche im Folgenden beschrieben werden. Die Gleichgewichtsbedingungen für den statischen Fall ist gegeben durch:

$$\frac{\partial \sigma_{ij}}{\partial x_j} + f_i = 0, \qquad 1 \leq i, j \leq 3, \tag{3.1}$$

und zeigt das Gleichgewicht der Kräfte an einem Objekt zwischen den Volumenkräften f_i und den zusammengefassten Oberflächenkräften $\frac{\partial \sigma ij}{\partial x_j}$. Die Gleichgewichtsbedingungen gelten für jeden Punkt im Körperinneren V, wobei $V \subset \mathbb{R}^3$ offen. An der Oberfläche eines Körpers A_t ist eine Flächenlast t^*, welche die äußere Belastung beschreibt, gegeben. Der Spannungsvektor t^* ist durch die Komponente des Spannungstensors σ eindeutig bestimmt:

$$\sigma_{ij} n_j = t_i^*, \qquad 1 \leq i, j \leq 3. \tag{3.2}$$

Die zweite Grundgleichung ist die Verschiebungs-Dehnung Gleichung, welche kleine Verzerrungen erlaubt. Damit lassen sich die Einträge des Dehnungstensors ϵ_{ij} wie folgt beschreiben:

$$\epsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \qquad 1 \leq i, j \leq 3. \tag{3.3}$$

Spannung und Dehnung werden in der dritten Gleichung zur linearen Elastizitätstheorie in Relation zueinander gesetzt:

$$\sigma ij = E_{ijkl}\epsilon_{kl}, \qquad 1 \leq i, j, k, l \leq 3, \tag{3.4}$$

für den Elastizitätstensor vierter Stufe E_{ijkl}. Eine einfachere Variante ist für ein homogenes, das heißt in allen Punkten des Körpers herrschen gleiche Materialeigenschaften, und isotropes, also für richtungsunabhängige Materialeigenschaften, Medium gegeben durch:

$$\sigma_{ij} = \lambda \epsilon_{kk}\delta_{ij} + 2\mu\epsilon_{ij}, \qquad 1 \leq i, j, k \leq 3, \tag{3.5}$$

wobei δ_{ij} das Kronecker-Delta darstellt und λ und μ unabhängige elastische Konstanten repräsentieren.

Damit stehen 15 Gleichungen mit 15 Unbekannten zur Verfügung. Die Unbekannten sind durch jeweils sechs Elemente des Dehnungs- und Spannungstensors sowie drei Unbekannte durch den Verschiebungsvektor gegeben. Hinzu kommen Randbedingungen wie beispielsweise (3.2). Aus dem Einsetzen der Gleichungen (3.3) und (3.5) in (3.1) folgt die sogenannte Lamé-Navier-Gleichung,

welche die Verformung elastischer Körper unter Einwirkung äußerer Kräfte charakterisiert:

$$(\lambda + \mu)\frac{\partial^2 u_j}{\partial x_j \partial x_i} + \mu \frac{\partial^2 u_i}{\partial x_j^2} + f_i = 0, \qquad 1 \leq i, j \leq 3. \tag{3.6}$$

3.1.2 Finite-Elemente Methode

Um mit Hilfe der Finiten-Elemente eine numerische Lösung einer Differentialgleichung zu ermitteln, muss diese in der schwachen Variationsformulierung ausgedrückt werden. Es wird nochmal die partielle Differentialgleichung (3.1) und (3.2) für den Spannungstensor σ mit Randbedingungen betrachtet:

$$\frac{\partial \sigma ij}{\partial x_j} + f_i = 0 \qquad \text{in } V \tag{3.7}$$

$$\sigma_{ij} n_j = t_i^* \qquad \text{auf } A_t. \tag{3.8}$$

Durch die Multiplikation eine virtuelle Verschiebung δv_i an (3.7) und Integration über V wird die schwache Form der partiellen Differentialgleichung beschrieben:

$$\int_V \frac{\partial \sigma_{ij}}{\partial x_j} \delta v_i dV + \int_V f_i \delta v_i dV = 0. \tag{3.9}$$

Durch Anwendung des Gaußschen Integralsatzes folgt:

$$\int_V \sigma_{ij} \delta \epsilon_{ij} dV - \int_V f_i \delta v_i dV - \int_{A_t} \sigma_{ij} n_j \delta v_i dA = 0. \tag{3.10}$$

Aufgrund von (3.3) und von (3.4) sowie mit Ausnutzung von Symmetrieeigenschaften gilt für $\delta \epsilon_{ij} = \frac{\partial \delta v_i}{\partial x_j}$ und $\sigma_{ij} = E_{ijkl} \frac{\partial u_i}{\partial x_l}$. Durch das Einsetzen der schwachen Form (3.10) und Nutzung der Randbedingung (3.8) folgt:

$$\int_V E_{ijkl} \frac{\partial u_k}{\partial x_l} \frac{\partial \delta v_i}{\partial x_j} dV - \int_V f_i \delta v_i dV - \int_{A_t} t_i^* \delta v_i dA = 0. \tag{3.11}$$

Nun wird das Gebiet V diskretisiert, indem es in Gitterelemente T unterteilt wird. Diese Teilelemente T sind im $\mathbb{R}^2$ beispielsweise durch Dreiecke und im $\mathbb{R}^3$ durch Tetraeder gegeben. Auf diesen Teilgebieten werden stückweise Polynome definiert. So wird ein endlichdimensionaler Raum $\hat{V}_h$ aufgespannt. $\hat{V}_h$ wird durch Funktionen aufgespannt, deren Einschränkung auf T Splines, also stückweise Polynome, sind. Des Weiteren sei $\{\phi_1, ...\phi_N\}$ eine Basis von $\hat{V}_h$, wobei die Basiselemente einfach zu bestimmen sind und einen kleinen Träger haben. Für die Basisfunktionen gelte: $\phi_k(z_i) = \delta_{ik}$ für alle $i, j = 1, ..., N$ und einen Gitterpunkt $z_i = (x_1, x_2, x_3)$. Auf diese Weise kann eine Funktion $u_h(x) \in \hat{V}_h$ als Linearkombination in dieser Basis ausgedrückt werden:

$$u_h(x) = \sum_{p=1}^{N} u_j \phi_p(x), \quad x \in V. \tag{3.12}$$

Dann kann (3.12) in Gleichung (3.11) eingesetzt werden. Da Gleichung (3.11) für jedes $\delta v_i \in V_h$, gilt diese insbesondere für alle Basisfunktionen ϕ_i. Dies führt zu einem linearen Gleichungssystem:

$$\sum_{p=1}^{N} \underbrace{\int_V E_{ijkl} \frac{\partial \phi_p(x)}{\partial x_l} \frac{\partial \phi_q(x)}{\partial x_j} dV}_{A_{pq}} u_{k_p} = \underbrace{\int_V f_i \phi_q(x) dV + \int_{A_t} t_i^* \phi_q(x) dA}_{b_q}, \tag{3.13}$$

$$\Rightarrow \sum_{p=1}^{N} A_{pq} u_p = b_q, \qquad q = 1, ..., N. \tag{3.14}$$

Es ergibt sich das lineare Gleichungssystem $Au = b$, wobei A die Steifigkeitsmatrix und u die gesuchte Lösung des Systems ist. Zur Lösung des Gleichungssystems bietet sich zum Beispiel das Gaußsche Eliminationsverfahren an.

Das betrachtete Objekt wird durch diskrete Knoten approximiert. Nur an diesen existiert eine Lösung der Differentialgleichung. Folglich ist die Qualität des Approximationsverfahrens stark von der Anzahl der Knoten und damit der Elemente abhängig. Ein feines Gitter sorgt zwar für eine gute Näherung, ist jedoch mit einer langen Rechenzeit verbunden. Es muss ein Mittelweg zwischen der geforderten Genauigkeit der Lösung und der erforderlichen Rechenzeit gefunden werden.

3.2 Optimierung

Es gibt verschiedene Methoden, um eine differenzierbare Funktion, ohne zur Verfügung stehende Ableitungen, zu minimieren. Diese Methoden können in vier verschiedene Klassen unterteilt werden. Zunächst kann die Ableitung approximativ berechnet werden, zum Beispiel durch das Anwenden von Finite-Differenzen (beispielsweise in Gill at al. [GMW 81] und in Dennis und Schnabel [DS 96]) oder durch Automatisches Differenzieren (siehe zum Beispiel Griewank und Walther [GW 08]). Die zweite Klasse von Verfahren zur Minimierung einer Funktion, ohne die Nutzung von Ableitungen, sind direkte Suchmethoden. Die zusätzlichen Informationen werden in diesem Fall durch das Auswerten der Zielfunktion an bestimmten Punkten generiert. Dazu zählt unter anderem das Nelder-Mead-Simplex Verfahren (vergleiche Nelder und Mead [NM 65]). Des Weiteren zählen modellbasierte Verfahren zur dritten Klasse der ableitungsfreien Optimierungsmethoden. Hier wird die Zielfunktion durch eine lokale Modellfunktion approximiert. Das Modell spiegelt das Verhalten der Zielfunktion in einem Bereich wider. Für diese Arbeit spielen die Trust-Region-basierten Verfahren wie etwa von Powell ([Pow 00], [Pow 02], [Pow 04b] und [Pow 09]) und Conn at al. (siehe [CSV 08]) eine wichtige Rolle. Zur vierten Klasse von Lösungsverfahren gehören die populationsbasierten Verfahren, in denen eine Vielzahl von Zielfunktionsauswertungen parallel berechnet werden und mit Hilfe von Prozessen aus der Natur die Lösung über mehrere Generationen erzeugt wird. Hier wird zwischen genetischen Algorithmen, Evolutionsstrategien, genetischer Programmierung und evolutionärer Programmierung unterschieden. In dieser Arbeit liegt der Fokus auf den Evolutionsstrategien (siehe beispielsweise [Rec 73]).

Zuerst wird in diesem Kapitel ein Einblick in die endlichdimensionale nichtlineare Optimierung gegeben, indem einige grundlegende Definitionen und Begriffe erläutert werden. Auf die Wahl der beiden Algorithmen wird in Abschnitt 5.1.3 eingegangen. Anschließend wird das Prinzip von evolutionären Algorithmen vorgestellt und im näheren Bezug auf den CMA-ES Algorithmus von Nikolaus Hansen (siehe [HO 96]) eingegangen. Darauf folgend wird die Idee von Trust-Region Verfahren erklärt und auf den ableitungsfreien Fall am Beispiel des BOBYQA Algorithmus von Powell (vgl. [Pow 09]) erweitert.

3.2.1 Problemstellung und Lösung eines Optimierungsproblems

Der folgende Abschnitt orientiert sich an [Alt 11] und [GT 93]. Ziel der Optimierung ist es ein Minimum oder Maximum einer Funktion zu finden. Das heißt es wird für eine zu minimierende Funktion, die *Zielfunktion*, $f : \mathbb{R}^n \to \mathbb{R}$ ein $x \in M$ gesucht, sodass

$$f(x) \leq f(y) \qquad \text{für alle } y \in M, \tag{3.15}$$

wobei M die *zulässige Menge* ist.
Optimierungsprobleme können ganz allgemein in zwei Klassen unterteilt werden. Zum einen existiert die Klasse der unbeschränkten Optimierungsprobleme, welche in der Standardform mit:

$$\min_{x \in \mathbb{R}^n} f(x), \tag{3.16}$$

beschrieben wird. Hier gilt für die zulässige Menge $M = \mathbb{R}^n$ oder $M \subseteq \mathbb{R}^n$, M offen. Andererseits können beschränkten Optimierungsprobleme wie folgt dargestellt werden:

$$\min_{x \in \mathbb{R}^n} f(x), \qquad \text{so dass} \tag{3.17}$$

$$\begin{aligned} g_i(x) &\leq 0 \quad \text{für } i = 1, ..., p, \\ h_j(x) &= 0 \quad \text{für } j = 1, ..., m, \end{aligned} \tag{3.18}$$

wobei $g_i, h_j : \mathbb{R}^n \to \mathbb{R}$ allgemein als Nebenbedingungen bezeichnet werden. Die Gleichung- und Ungleichungsbedingungen (3.18) beschreiben die Beschränkungen des Optimierungsproblems, sodass die zulässige Menge durch $M = \{x \in \mathbb{R}^n \mid g_i(x) \leq 0 \text{ für } i = 1, ..., p,\ h_j(x) = 0 \text{ für } j = 1, ..., m\}$ definiert ist. Ein Punkt $x \in \mathbb{R}^n$ heißt *zulässig*, wenn er alle Nebenbedingungen erfüllt.
Mit Hilfe der folgenden Gleichung ist es möglich jedes Maximierungsproblem in ein Minimierungsproblem umzuwandeln:

$$\min_{x \in M} f(x) = -\max_{x \in M} -f(x). \tag{3.19}$$

Für $x \in \mathbb{R}^n$ und $r > 0$ wird mit

$$\mathcal{B}_r(x) = \{y \in \mathbb{R}^n | \|y - x\| < r\}, \tag{3.20}$$

die offenen Kugel mit Radius r um den Punkt x bezeichnet. Ein Punkt $\hat{x} \in M$ heißt *lokales Minimum* von f, falls es ein $r > 0$ mit

$$f(\hat{x}) \leq f(y) \qquad \text{für alle } y \in M \cap \mathcal{B}_r(\hat{x}), \tag{3.21}$$

gilt. Des Weiteren heißt $\hat{x} \in \mathbb{R}^n$ *globales Minimum* von f, falls

$$f(\hat{x}) \leq f(y) \qquad \text{für alle } y \in M, \tag{3.22}$$

gilt. Funktionen wie $f(x) = cos(x)$ verdeutlichen, dass es in der nichtlinearen Optimierung viele lokale sowie globale Optima geben kann.

3.2.2 CMA-ES

Der CMA-ES (Covariance Matrix Adaption - Evolutionary Strategy) Algorithmus wurde 1996 von Nikolaus Hansen und Andreas Ostermeier in [HO 96] vorgestellt. Er gehört zu der Klasse der Evolutionsstrategien und basiert auf einer adaptiven Approximation der Hessematrix. Zuächst wird kurz die grundlegende Idee von Evolutionsstrategien vorgestellt und im Anschluss näher auf den CMA-ES Algorithmus eingegangen. Die nachfolgenden Unterkapitel orientieren sich vor allem an Veröffentlichungen von Hansen selbst (siehe [HK 04], [Han 06], [BF 07] und [Han 11]).

3.2.2.1 Idee von Evolutionsstrategien

Evolutionsstrategien sind durch den Prozess der natürlichen Evolution inspiriert. Es werden die Vorgänge der Selektion, Rekombination und Mutation simuliert, um eine Lösung des Problems (3.16) bzw. (3.17) und 3.18) zu finden. Diese Art von Algorithmen arbeitet auf stochastischer Basis und findet besonders bei komplexen oder unstetigen Zielfunktionen Anwendung. Der Vorteil solcher Verfahren ist, dass sie auf eine große Anzahl von Problemklassen anwendbar sind. Jedoch sind Algorithmen aus dem Bereich oft langsam, da sie nur mit der Auswertung der Zielfunktion arbeiten und große Datenmengen erzeugen müssen, um den Suchraum stochastisch abzutasten. Es gibt eine Vielzahl evolutionärer Algorithmen, die aber alle die gleiche Grundstruktur aufweisen. Zunächst wird eine Anfangspopulation aus mehreren zufällig gewählten Punkten, den sogenannten Individuen, welche zusammen eine Population bilden, erzeugt. Im nächsten Schritt wird die Population so modifiziert, dass Individuen mit einer größeren Fitness begünstigt werden. Das heißt Individuen, die für die Problemstellung die besten Lösungen liefern, werden präferiert. Es werden die fittesten Individuen als Eltern der nächsten Generation bevorzugt bzw. gehören selbst zur nächsten Generation. Dies entspricht dem Prozess der natürlichen Selektion, welcher als Wichtungsfaktor für die Individuen betrachtet werden kann. Mittels Rekombination werden die Elternlösungen durch Kombination zu einem neuen Individuum der nächsten Generation. Mit Hilfe der Mutation wird die genetische Vielfalt vergrößert, indem Individuen zufällig verändert werden.

3.2.2.2 Konzept

Der wesentliche Gedanke des Algorithmus ist die Datengewinnung mittels gewichteter empirischer Schätzung einer Gaußverteilung. Auf den Startvektor $x_0 \in \mathbb{R}^n$ wird eine n-dimensionale Normalverteilung $\mathcal{N} \sim (m_0, C_0)$ angewendet, um die Startpopulation zu erzeugen, wobei der Mittelwert $m_0 = x_0$ ist und die Kovarianzmatrix C_0 auf die Einheitsmatrix $I^{n \times n}$ gesetzt wird. So entsteht eine Population der Größe λ mit den Individuen $x_{k,t}$ für $k = 1, ..., \lambda$ der Generation bzw. Iteration t. Anschließend werden die Individuen auf ihre Fitness hin bewertet und nach dieser sortiert. Da es hier gilt eine Minimierungsaufgabe zu lösen, werden Individuen $x_{k,t}$ mit kleinem Zielfunktionswerte bevorzugt und damit mehr gewichtet. Aus den μ ($\leq \lambda$) besten Individuen wird ein neuer gewichteter empirischer Mittelwert m_t erzeugt und die neue Kovarianzmatrix C_t aus den Evolutionspfaden p_σ und p_c und dem Skalierungskoeffizienten σ_t erstellt. Die nächste Generation wird durch die Anwendung der neuen Kovarianzmatrix $\mathcal{N} \sim (m_k, C_k)$ erzeugt. Durch Rotation und Skalierung bestimmt die Kovarianzmatrix die Form der Verteilung der Punkte im Raum. Das

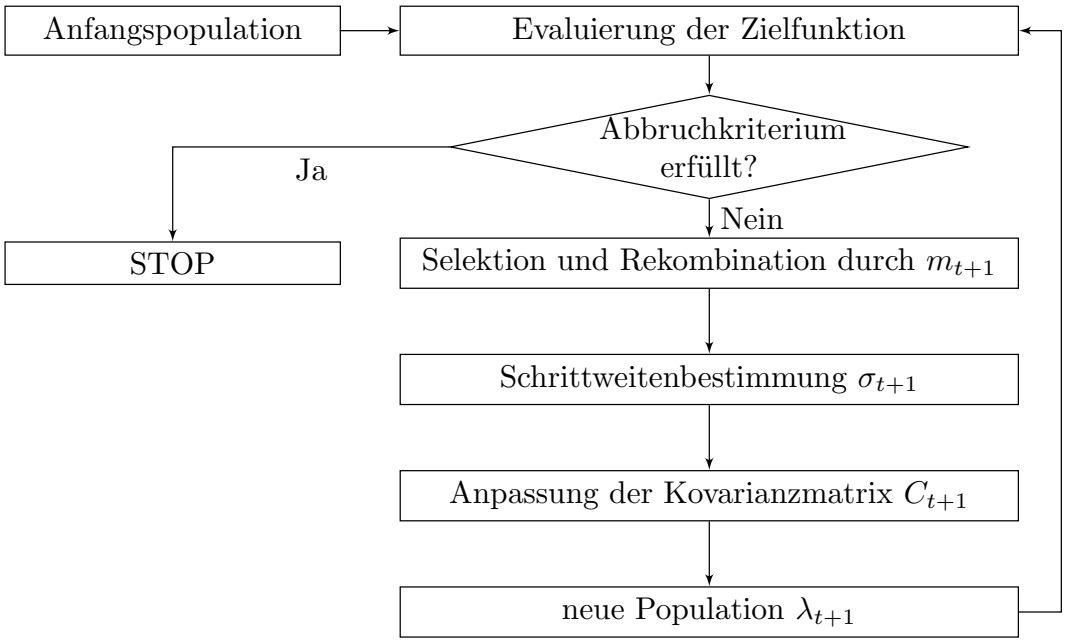


Abbildung 3.1: Grundprinzip des CMA-ES Algorithmus

Verfahren bricht ab, falls die Bedingung der maximalen Anzahl an Funktionsauswertungen oder Iterationen erreicht ist oder die Kondition der Kovarianzmatrix einen bestimmten Wert erreicht. Abbildung 3.1 zeigt die einzelnen Vorgehensschritte des CMA-ES Algorithmus. Im Anhang A.2 wird der CMA-ES Algorithmus zusammengefasst.

3.2.2.3 Stichprobenauswahl und Mittelwertberechnung

In CMA-ES wird eine Population neuer Individuen bzw. neue Suchpunkte durch Stichprobenentnahme der multivariaten Normalverteilung bestimmt. Die Gleichung, um die neuen Individuen für die Generation $t = 0, 1, 2, ...,$ zu erzeugen ist:

$$x_{k,t+1} \sim m_t + \sigma_t \mathcal{N}(0, C_t), \qquad k = 1, ..., \lambda, \tag{3.23}$$

wobei "$\sim$" die Gleichheit bei Verteilungen symbolisiert. Die Normalverteilung kann in verschiedenen Formen ausdrückt werden:

$$m + \mathcal{N}(0, C) \sim m + BDB^T \mathcal{N}(0, I) \sim m + BD\mathcal{N}(0, I), \tag{3.24}$$

wobei $C = BD^2B^T$ die Eigenwertzerlegung der Kovarianzmatrix C ist. Die Spalten der Orthogonalmatrix B stellen die Eigenvektoren von C dar, während die Diagonalelemente von D die Wurzeln der Eigenwerte von C sind. Mit Hilfe der Matrizen B und D kann die Verteilungsform der Individuen bestimmt werden. Hier ist der Mittelwert m_t der Verteilung ein gewichtetes Mittel der μ besten Punkte von $x_{1,k}, ..., x_{\lambda,k}$:

$$m_k = \sum_{i=1}^{\mu} w_i x_{i,k}, \tag{3.25}$$

sodass $\sum_{i=1}^{\mu} w_i = 1$ und $w_1 \geq w_2 \geq ... \geq w_\mu > 0$. Im biologischen Zusammenhang bedeutet das, dass die durch Selektion besten Individuen durch das gewichtete Mittel rekombiniert werden. Die Wahl von verschiedenen Wichtungen w_i kann wie folgt erfolgen:

$$w_i := \ln(\mu + 1) - \ln(i), \qquad i = 1, ..., \mu, \tag{3.26}$$

und wird auch als Selektionsmechanismus interpretiert. Die Gleichung $\mu_{eff} = (\sum_{i=1}^{\mu} w_i^2)^{-1}$ wird im Folgenden gebraucht und beschreibt eine Art gewichtete Selektionskonstante. Gilt $\mu_{eff} = \mu$, dann sind alle Gewichte w_i gleich.

3.2.2.4 Schrittweitenwahl

Die Kovarianz-Matrix-Adaption kontrolliert die Skalierung der Verteilung nicht explizit, sodass die optimale Schrittweite nicht gut approximiert werden kann. Um die Schrittweite σ_t zu kontrollieren, wird ein evolutionärer Pfad genutzt. Für ableitungsbasierte Optimierungsverfahren wird eine Abstiegsrichtung wählen, welche für evolutionäre Optimierungsprozesse jedoch nicht zur Verfügung steht.
Um die Schrittweite σ_{t+1} der nächsten Generation zu bestimmen, wird der evolutionäre Pfad $p_{\sigma,t+1}$ benutzt, welcher einen konjugierten Evolutionspfad repräsentiert. Der evolutionäre Pfad gibt die Länge und Richtung der Schritte der früheren Generationen an, um Informationen über

die Suchrichtung zu erhalten:

$$p_{\sigma,t+1} = (1 - c_\sigma)p_{\sigma,t} + \sqrt{c_\sigma(2 - c_\sigma)\mu_{eff}}C_t^{-\frac{1}{2}}\frac{m_{t+1} - m_t}{\sigma_t}, \tag{3.27}$$

wobei $c_\sigma \leq 1$. Je größer die Konstante ist, desto weniger Informationen von vorherigen Generationen wird bewahrt. Nun wird die Länge des nächsten Schrittes bestimmt. Die Größe der Kovarianz kann durch Multiplikation von σ_i auf diese variiert werden. Folglich sind die Individuen der nächsten Generation entweder näher zusammen bzw. weiter voneinander entfernt. Die Länge des evolutionären Pfades $\|p_{\sigma,t}\|$ wird mit der Länge des Erwartungswertes eines standardnormalverteilten Zufallsvektors verglichen, um die globale Schrittweite für die nächste Generation zu erhalten:

$$\sigma_{t+1} = \sigma_t \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|p_{\sigma,t+1}\|}{E\|\mathcal{N}(0, I)\|} - 1\right)\right), \tag{3.28}$$

wobei d_σ einen Dämpfungsparameter darstellt. Ist der evolutionäre Pfad länger als angenommen, so wird die Schrittweite vergrößert. Dadurch werden die zu kleinen Schritte durch längere Schrittweiten ersetzt und deshalb kann erwartet werden, dass die Konvergenz in Folge dessen beschleunigt wird. Die Schrittweite σ_t wird verkleinert, falls der evolutionäre Pfad kleiner als erwartet ist. In diesem Fall ist es möglich ein Optimum in der Nähe zu finden.

3.2.2.5 Anpassung der Kovarianzmatrix

Die Kovarianzmatrix gibt die Form der Verteilung an. Sie beschreibt einen Ellipsoiden, in dem sich alle Individuen befinden. Um die Effizienz des Algorithmus weiter zu verbessern wird ein weiterer evolutionärer Pfad eingeführt. Äquivalent zur Bestimmung der Gleichungen (3.27) und (3.28) bei der Berechnung der Schrittweite σ_t, wird der evolutionäre Pfad $p_{c,t+1}$ ermittelt:

$$p_{c,t+1} = (1 - c_c)p_{c,t} + \sqrt{c_c(2 - c_c)\mu_{eff}}\frac{m_{t+1} - m_t}{\sigma_t}. \tag{3.29}$$

Die neue Kovarianzmatrix C_{t+1} kann durch die Informationen des evolutionären Pfades $p_{c,t+1}$ sowie der alten Kovarianzmatrix berechnet werden:

$$\begin{aligned} C_{t+1} =& (1 - c_{cov})C_t + \frac{c_{cov}}{\mu_{eff}}p_{c,t+1}p_{c,t+1}^T \\ &+ c_{cov}(1 - \frac{1}{\mu_{eff}})\sum_{i=1}^{\mu} w_i \left(\frac{x_{i,t+1} - m_t}{\sigma_t}\right)\left(\frac{x_{i,t+1} - m_t}{\sigma_t}\right)^T, \end{aligned} \tag{3.30}$$

wobei c_{cov} und μ_{eff} gewichtete Konstanten sind. Die Lernrate der Kovarianzmatrix ist durch c_{cov} gegeben, welches den Anteil an neuen Informationen berechnet, die zu C_t während jeder Generation addiert werden. Für die Wahl der Parameter c_c, c_{cov}, c_σ, d_σ und μ_{eff}, um eine robustes Verfahren zu erhalten, wird auf [HMK 03] verwiesen. Die Populationsgröße λ und die Anzahl an Eltern μ wird standardmäßig auf $\lambda = 4 + \lfloor 3 \ln n \rfloor$ und $\mu = \lfloor \frac{\lambda}{2} \rfloor$ gesetzt.

3.2.2.6 Konvergenz

Ein Konvergenzbeweis zum CMA-ES Algorithmus, wie für die meisten evolutionären Algorithmen, existiert nicht. Aus Erfahrung kann gesagt werden, dass sich der Abstand zum Optimum in jeder Iteration durchschnittlich um einen konstanten Wert verringert. Des Weiteren führen kleine Populationsgrößen λ zu schnellerer Konvergenz, wobei größere Populationen dafür sorgen können, dass der Algorithmus nicht in ein lokales Optimum läuft.

3.2.3 BOBYQA

Der BOBYQA (Bound Optimization BY Quadratic Approximation) Algorithmus wurde 2009 von M. Powell vorgestellt. BOBYQA gehört zu der Gruppe der ableitungsfreien Verfahren, welche sich die ableitungsbasierte Trust-Region Technik zu nutze machen. Trust-Region Methoden fungieren, indem sie ein lokales Modell der Zielfunktion approximieren und dieses innerhalb eines Bereichs, dem Trust-Region, minimieren.

In diesem Abschnitt wird zunächst die grundlegende Idee von Trust-Region-Methoden vorgestellt sowie auf die Konvergenztheorie dieser eingegangen. Anschließend wird das allgemeine Konzept des ableitungsfreien BOBYQA Algorithmus vorgestellt, welcher das lokale Modell durch Interpolation konstruiert.

3.2.3.1 Trust-Region Konzept

Die folgenden Abschnitte zum Trust-Region Konzept und der Konvergenztheorie orientieren sich, soweit nicht anders gekennzeichnet, an [CGT 87] und [Alt 11].

Trust-Region-Verfahren sind iterative, ableitungsbasierte und robuste Verfahren, welche starken theoretischen Aussagen unterliegen. Es sind Methoden zur Minimierung von nichtlinearen Optimierungsproblemen ohne Nebenbedingungen. Diese Gruppe von Verfahren wurde zuerst von Levenberg [Lev 44] und Marquardt [Mar 63] im Bereich von nichtlinearen kleinste Quadrate-Problemen vorgestellt. Weiterentwickelt wurde die Methode von Goldfeldt et al. [GQT 66]. Die erste Konvergenztheorie für unbeschränkte Optimierungsprobleme in diesem Kontext wurde von Powell in [Pow 70] gemacht.

Betrachtet wird zunächst eine zweimal stetig differenzierbaren Zielfunktion $f : \mathbb{R}^n \to \mathbb{R}$, welche minimiert wird:

$$\min_{x \in \mathbb{R}^n} f(x). \tag{3.31}$$

Die Zielfunktion wird innerhalb einer bestimmten Umgebung $\mathcal{B}_k$ um die aktuelle Iterierte x_k durch eine quadratische Modellfunktion Q_k approximiert. Nun wird die Modellfunktion innerhalb dieser

Umgebung minimiert. Die Umgebung in der dem Modell „vertraut“ (englisch: trust) wird, wird als Trust-Region bezeichnet. Der Trust-Region ist für die k-te Iteration durch den Trust-Region-Radius $\mathcal{B}_{\rho_k}(x_k)$ mit $\rho_k \geq 0$ gegeben. Idealerweise soll der Trust-Region das Gebiet darstellen, in dem die Modellfunktion die Zielfunktion ausreichend gut approximiert. Die Minimierung der Modellfunktion innerhalb des Trust-Region erfolgt durch Lösen des folgenden Subproblems. Gesucht ist hier der Schritt $x_k + d_k$ als Lösung des Problemes:

$$\begin{aligned} &\min Q_k(x_k + d_k), \text{ so dass} \\ &\|x_k + d_k\| \leq \rho_k. \end{aligned} \tag{3.32}$$

Nach dem Satz von Weierstraß hat dieses Minimierungsproblem stets eine Lösung, da Q_k stetig und $\|d\| \leq \rho_k$ kompakt ist. Die Zielfunktion wird durch die quadratische Modellfunktion Q_k beispielsweise durch die Nutzung der Taylorreihe approximiert:

$$Q_k(x_k + d_k) = c + g_k^T d_k + \frac{1}{2} d_k^T H_k d_k, \tag{3.33}$$

mit $c = f(x_k)$, $g_k = \nabla f(x_k)$ und $H_k = \nabla^2 f(x_k)$ stellt die Hessematrix von f bzw. eine Approximation dieser dar. Durch das Lösen des Trust-Region-Subproblems (3.32) wird der Punkt $x_k + d_x$ bestimmt. Nun wird geprüft ob die Modellfunktion die Zielfunktion gut approximiert. Dazu wird der tatsächliche Abstieg mit dem vorhergesagten Abstieg der Modellfunktion verglichen. Die Bewertung der Qualität des berechneten Schrittes wird durch das Verhältnis:

$$r_k = \frac{f(x_k) - f(x_k + d_k)}{Q_k(x_k) - Q_k(x_k + d_k)}, \tag{3.34}$$

ausgedrückt. Der Iterationsschritt ist erfolgreich, wenn $r_k \geq \delta_1$, wobei δ_1 eine zu Beginn des Algorithmus gesetzte Konstante darstellt. Häufig gilt $\delta_1 = 0.1$. Für die neue Iterierte x_{k+1} gilt dann $x_{k+1} = x_k + d_k$. Ist das Verhältnis kleiner als δ_1, dann ist die Abnahme der Modellfunktion unbefriedigend, der Trust-Region-Radius wird verkleinert und es gilt $x_{k+1} = x_k$. Das Modell wird nun innerhalb des verkleinerten Trust-Region minimiert. Algorithmus A.1 im Anhang zeigt einen Basis Trust-Region Algorithmus.

3.2.3.2 Konvergenztheorie

Für den Zweck der globalen Konvergenz ist es ausreichend eine approximative Lösung zu finden, welche innerhalb des Trust-Region liegt und eine ausreichenden Abstieg des Modells garantiert. Dieser ausreichenden Abstieg lässt sich durch den Cauchy-Punkt beschreiben. Dazu wird ein Modell in der steilsten Abstiegsrichtung betrachtet:

$$Q_k(x_k - \lambda g_k) = f(x_k) - \lambda \|g_k\|^2 + \frac{\lambda^2}{2} g_k^T H_k g_k, \tag{3.35}$$

mit $g_k \neq 0$ und $\lambda \geq 0$. Die Modellfunktion (3.35) wird nun innerhalb des Trust-Region minimiert.

Die Lösung des Problems ist durch den Cauchyschritt $d_k^C = \lambda_k^C \frac{\rho_k}{\|g_k\|} g_k$ gegeben, wobei:

$$\lambda_k^C = \begin{cases} \frac{\|g_k\|^3}{\rho_k g_k^T H_k g_k} & \text{, falls } \frac{\|g_k\|^3}{g_k^T H_k g_k} \leq 0 \text{ und } g_k^T H_k g_k > 0, \\ 1 & \text{, sonst.} \end{cases} \tag{3.36}$$

Der Quotient aus tatsächlicher und vorhergesagter Reduktion erlaubt den Abstieg der Zielfunktion f zu beschreiben. Diese Beschreibung bildet die Basis für die globale Konvergenztheorie von Trust-Region Methoden.

Um das Problem (3.31) iterativ zu lösen, wird ein $x_{k+1} = x_k + d_k$ gesucht, so dass:

$$f(x_{x+1}) < f(x_k), \tag{3.37}$$

gilt. In Algorithmus A.1 ist ein Iterationsschritt erfolgreich, wenn die folgende Ungleichung gilt:

$$f(x_k) - f(x_{k+1}) \geq \delta_1(Q_k(x_k) - Q_k(x_{k+1})). \tag{3.38}$$

Es wird für $Q_k(x_k) - Q_k(x_{k+1})$ die folgende untere Grenze definiert, damit der gewünschte Abstieg der Zielfunktion, wie in (3.37) gewährleistet ist:

$$Q_k(x_k) - Q_k(x_{k+1}) > 0. \tag{3.39}$$

Der vorhergesagte Abstieg $Q_k(x_k) - Q_k(x_{k+1})$ kann mittels Cauchy-Abstiegsmethode analysiert werden. Durch den Vergleich des Wertes der Modellfunktion am Cauchypunkt $x_{k+1}^C = x_k + d_k^C$ und des Modellwertes des letzten Iterationspunktes genügt es zum Erfüllen von Gleichung (3.39), dass folgende Ungleichung gilt:

$$Q_k(x_k) - Q_k(x_k + d_k) \geq c[Q_k(x_k) - Q_k(x_k + d_k^C)], \tag{3.40}$$

wobei $c \geq 0$ ist. Nach Powell [Pow 75] gilt folgende Abschätzung der vom Modell vorhergesagten Verminderung des Zielfunktionswerts:

$$Q_k(x_k) - Q_k(x_k + d_k^C) \geq \frac{1}{2}\|g_k\| \min\{\rho_k, \frac{\|g_k\|}{\|H_k\|}\}. \tag{3.41}$$

Werden die Ungleichungen (3.38), (3.40) und (3.41) zusammengefügt, dann folgt:

$$f(x_{k+1}) \leq f(x_k) - \frac{\delta_1 c}{2}\|g_k\| \min\{\rho_k, \frac{\|g_k\|}{\|H_k\|}\}. \tag{3.42}$$

Ausgehend von Bedingung (3.42), welche einen ausreichenden Abstieg der Zielfunktion garantiert, kann die Konvergenz des Trust-Region Verfahrens mit quadratischen Modellfunktionen (3.33) mittels Theorem 3.43 gezeigt werden.

Der Beweis zum Theorem 3.43 findet sich beispielsweise in [NW 99].

Theorem 3.43. *Sei $f : \mathbb{R}^n \to \mathbb{R}$ zweimal stetig differenzierbar und die Nievaumenge $N_0 = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ kompakt. Endet Algorithmus A.1 nicht nach endlich vielen Schritten, dann gilt:*

$$\lim_{k\to\infty} \|g_k\| = 0. \tag{3.44}$$

Die Folge $\{x_k\}$ der Iterationspunkte besitzt mindestens einen Häufungspunkt. Für jeden Häufungspunkt $\hat{x}$ gilt:

$$\nabla f(\hat{x}) = 0. \tag{3.45}$$

3.2.3.3 BOBYQA Konzept

Der Algorithmus BOBYQA gehört zu den iterativen, ableitungsfreien Methoden, welche die folgenden durch Randbedingungen beschränkten Optimierungsprobleme löst:

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} f(x), \quad \text{sodass} \\ &a_i \leq x_i \leq b_i, \qquad i = 1, ..n. \end{aligned} \tag{3.46}$$

Die Zielfunktion $f(x)$ kann hier als Black-Box-Funktion betrachtet werden. Das Verfahren nutzt einen Trust-Region Algorithmus für Optimierungsprobleme, welche durch Box-Randbedingungen beschränkt sind. Der BOBYQA Algorithmus verwendet weder direkte Ableitungsinformationen der Zielfunktion, noch werden sie explizit approximiert. Stattdessen wird die Zielfunktion durch die Konstruktion von lokalen quadratischen Approximationen Q_k dieser mittels Interpolation realisiert. Auf die Polynominterpolation wird in Abschnitt 3.2.3.4 eingegangen. Generell werden quadratische Modelle konstruiert, um das Krümmungsverhalten der Zielfunktion mit zu betrachten. Für das Aufstellen eines kompletten quadratischen Modells werden im Allgemeinen $\frac{1}{2}(n+1)(n+2)$ Interpolationspunkte und damit auch genauso viele Funktionsauswertungen benötigt. BOBYQA kann jedoch auch mit weniger Punkten auskommen. Es ist dem Benutzer überlassen die Anzahl der Interpolationspunkte m aus dem Intervall $[n+2, \frac{1}{2}(n+1)(n+2)]$ zu wählen. Da numerische Ergebnisse zeigen, dass $m = 2n+1$ der effizienteste Wert für die Anzahl der Interpolationspunkte ist, wird von nun an $m = 2n+1$ gewählt. Durch die Reduktion der Interpolationspunkte wird für den Fall $m = 2n+1$ die Komplexität des Algorithmus von $\mathcal{O}(n^4)$ auf $\mathcal{O}(n^2)$ reduziert.

Vor dem Start des Algorithmus muss der Nutzer den Startwert $x_0 \in \mathbb{R}^n$, einen Start Trust-Region-Radius ρ_{beg}, den finalen Trust-Region Radius ρ_{end}, die Anzahl der Interpolationspunkte m und die Grenzen a und b des Problems definieren. Die erste Menge der Interpolationspunkte wird durch die Positionierung der Punkte am Rand des Trust-Region, d.h. im Abstand von $\rho_1 := \rho_{beg}$ zu x_0 und parallel zu den Koordinatenachsen konstruiert. Da alle Interpolationspunkte zulässig sein müssen, wird ein Fehler ausgegeben, falls $b_i - a_i \geq 2\rho_1$, für $i = 1, ..., n$, nicht erfüllt ist. Der Startwert x_0 wird eventuell automatisch verschoben um sicherzustellen, dass die Interpolationspunkte zulässig sind.

Zu Beginn der k-ten Iteration existiert bereits ein quadratisches Modell Q_k der Zielfunktion, sodass die folgenden Interpolationsbedingungen gelten:

$$Q_k(y_i) = f(y_i), \qquad i \in S, \tag{3.47}$$

wobei $S := \{1, ..., m\}$ die Indexmenge der Interpolationspunkte, y_i die Interpolationspunkte und $Y_k = \{y_i : i \in S\}$ sind. Des Weiteren existiert ein Trust-Region Radius ρ_k und der Punkt x_k mit $f(x_k) = \min\{f(y_i) : i \in S\}$.

Ist die maximale Anzahl an Funktionsauswertungen erreicht oder stimmt der Trust-Region Radius mit dem finalen Trust-Region Radius überein, so stoppt der Algorithmus und gibt x_k und $f(x_k)$ aus. Sonst wird ein neuer Interpolationspunkt berechnet und ersetzt einen der derzeitigen Punkte. Der neue Punkt wird durch das Finden des Schrittes d_k berechnet, sodass $x_k + d_k$ zulässig und nicht bereits einer der Interpolationspunkte y_i, $i \in S$, ist. Dieser Schritt wird durch das Ausführen einer Trust-Region-Iteration oder einer Alternativ-Iteration erzeugt. Eine Trust-Region-Iteration bestimmt d_k durch das Lösen des folgenden Subproblems:

$$\begin{aligned} &\min_{d_k \in \mathbb{R}^n} Q_k(x_k + d_k), \quad \text{sodass} \\ &a \leq x_k + d_k \leq b, \quad \|d_k\| \leq \rho_k. \end{aligned} \tag{3.48}$$

Dieses Problem wird durch die Bestimmung einer aktiven Menge-Version des abgeschnittenen konjugierten Gradientenverfahrens gelöst (vgl. [Pow 09]).

Die Alternativ-Iteration sucht ein d_k, welches die Geometrie der Interpolationspunkte verbessert. Es wird ein d_k gewählt, welches linear unabhängig zu den anderen Iterpolationspunkten ist, um das quadratische Modell zu optimieren. Ist die Trust-Region Iteration nicht erfolgreich, dann wird ein Punkt x_j, mit $|x_j - x_k| > 2\rho_k$, durch einen anderen Punkt ersetzt. Dieser neue Punkt geht aus der Maximierung des Lagrangepolynoms

$$\max_{d_k \in \mathbb{R}^n} |l_j(x_k + d_k)|, \quad \text{sodass} \tag{3.49}$$

$$\|d_k\| \leq \rho_k, \tag{3.50}$$

hervor. Lagrangepolynome spielen eine wesentliche Rolle, um Singulariäten eines Gleichungssystems zu vermeiden. Für weitere Details zur Prozedur wird auf [Pow 09] verwiesen.

Einer der alten Interpolationspunkte y_t, welcher am weitesten vom aktuellen Minimum entfernt ist, wird anschließend durch $x_k + d_k$ ersetzt, um eine neue Menge von Interpolationspunkten zu erzeugen. Diese Interpolationspunkte haben die Form:

$$\hat{y}_i = \begin{cases} y_i, & \text{falls } i \neq t \\ x_k + d_k, & \text{falls } i = t. \end{cases} \tag{3.51}$$

Vor der $k+1$-ten Iteration werden x_{k+1}, Q_{k+1} und ρ_{k+1} vor der neuen Iteration berechnet. x_{k+1}

wird wie folgt berechnet:

$$x_{k+1} = \begin{cases} x_k, & \text{falls } f(x_k) \leq f(x_k + d_k) \\ x_k + d_k, & \text{falls } f(x_k) > f(x_k + d_k). \end{cases} \tag{3.52}$$

Das neue quadratische Modell Q_{k+1} wird durch die Minimierung der Frobeniusnorm von $\nabla^2 Q_{k+1} - \nabla^2 Q_k$ unter den Nebenbedingungen $Q_{k+1}(\hat{y}_i) = f(\hat{y}_i)$, $i \in S$, realisiert. Dies wird durch das Lösen eines linearen Gleichungssystems verwirklicht. Die Herleitung ist in Abschnitt 3.2.3.5 erklärt.

Die guten Konvergenzaussagen von ableitungsbasierten Trust-Region Verfahren können auf den ableitungsfreien Fall übertragen werden.

Der Algorithmus ist in 3.1. Weitere Details zum BOBYQA Algorithmus finden sich in [Pow 09].

Algorithmus 3.1 : BOBYQA Algorithmus

Schritt 1: Initialisierung:
Seien x_0, $f(x_0)$ sowie die Trust-Region Radien ρ_{beg} und ρ_{end} gegeben. Wähle eine Start-Interpolationsmenge Y_1, welche x_0 und $2m$ Punkte um x_0 enthält. Setze $\rho_1 = \rho_{beg}$. Setze $k = 0$.
Berechne $x_1 \in Y_1$, sodass $f(x_1) = \min_{y \in Y_1} f(y)$.
Schritt 2: Abbruchkriterium:
Ist die maximale Anzahl an Funktionsauswertungen erreicht oder gilt $\rho_k = \rho_{end}$, dann STOP.
Schritt 3: Modellberechnung:
Konstruiere ein Modell $Q_k(x_k)$, welches die Funktion f an der Menge der Interpolationspunkte Y_k interpoliert, so dass die Interpolationsbedingungen (3.47) erfüllt sind. Die restlichen Freiheitsgrade werden durch die Minimierung der Frobeniusnorm von $\nabla^2 Q_{k+1} - \nabla^2 Q_k$ unter den Nebenbedingungen $Q_{k+1}(\hat{y}_i) = f(\hat{y}_i)$, $i \in S$ genommen.
Schritt 4: Schrittberechnung:
Berechne den Punkt $x_k + d_k$ durch das Lösen des Subproblems (3.48).
Bestimme $f(x_k + d_k)$ und den Quotienten r_k wie in (3.34).
Schritt 5: Update der Interpolationsmenge:
Erfolgreiche Iteration: Ist $r_k \geq 0.1$, dann wird $x_k + d_k$ in die Menge der Interpolationspunkte aufgenommen und ein bereits existierender Punkt entfernt.
Nicht erfolgreiche Iteration: Falls $r_k < 0.1$ ist und liegen die Punkt der Interpolationsmenge Y_k ungeeignet im Trust-Region, so wird die Geometrie der Interpolationspunkte verbessert, indem ein Punkt der Menge Y_k ersetzt wird.
Schritt 6: Trust-Region Update:
Setze

$$\rho_{k+1} = \begin{cases} \min\left[\frac{1}{2}\rho_k, \|d_k\|\right] & \text{, falls } r_k \leq 0.1 \\ \max\left[\frac{1}{2}\rho_k, \|d_k\|\right] & \text{, falls } 0.1 \leq r_k \leq 0.7 \\ \max\left[\frac{1}{2}\rho_k, 2\|d_k\|\right] & \text{, falls } r_k > 0.7. \end{cases}$$

Schritt 7: Update der Iterierten:

$$x_{k+1} = \begin{cases} x_k, & \text{falls } f(x_k) \leq f(x_k + d_k) \\ x_k + d_k, & \text{falls } f(x_k) > f(x_k + d_k). \end{cases}$$

Setze $k = k + 1$ und gehe zu Schritt 2.

3.2.3.4 Polynominterpolation

Unter einer Polynominterpolation wird ein Polynom, welches exakt durch vorgegebene Punkte verläuft, verstanden. Im ableitungsfreien Trust-Region Fall wird die Modellfunktion Q durch Interpolation an gegebenen Zielfunktionswerten, den Interpolationspunkten y_i, $i \in S$, mittels folgender Bedingung berechnet:

$$Q(y_i) = f(y_i) \quad \text{für alle } i \in S. \tag{3.53}$$

Eine Basis $\{\phi_1(x), \phi_2(x), ..., \phi_m(x)\}$ vom Raum der Polynome $\mathcal{P}_d^n$ vom Grad $\leq d$ im $\mathbb{R}^n$ ist eine Menge von m Polynomen vom Grad $\leq d$, welche den $\mathcal{P}_d^n$ aufspannt. Beispielsweise ergibt sich für ein Polynom zweiten Grades ($d = 2$) $m = \frac{1}{2}(n+1)(n+2)$ Unbekannte, um das Polynom im $\mathbb{R}^n$ eindeutig zu bestimmen. Es können verschiedenen Basen gewählt werden. Zu den bekanntesten zählen die Monome, Lagrange- oder Newton-Polynome. Für die Basis $\phi(x)$ kann ein Polynom $P(x) \in \mathcal{P}_d^n$ wie folgt dargestellt werden:

$$P(x) = \sum_{j=1}^{m} a_j \phi_j(x), \tag{3.54}$$

mit $a_j \in \mathbb{R}$. Sei die Menge der Interpolationspunkte y_i für alle $i \in S$ gegeben. Die Koeffizienten a_j, $j = 1, ..., m$ können durch das lösen des Gleichungssystems:

$$M(\phi, Y) a_\phi = f(Y), \tag{3.55}$$

$$M(\phi, Y) = \begin{pmatrix} \phi_1(y_1) & \phi_2(y_1) & \dots & \phi_q(y_1) \\ \phi_1(y_2) & \phi_2(y_2) & \dots & \phi_q(y_2) \\ \vdots & \vdots & & \vdots \\ \phi_1(y_m) & \phi_2(y_m) & \dots & \phi_q(y_m) \end{pmatrix}, \; a_\phi = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{pmatrix}, \; f(Y) = \begin{pmatrix} f(y_1) \\ f(y_2) \\ \vdots \\ f(y_m) \end{pmatrix}, \tag{3.56}$$

bestimmte werden. Um das Gleichungssystem zu lösen muss, darauf geachtet werden, dass die Matrix nicht singulär wird. Die richtige Wahl der Menge Y, damit dies nicht zum Problem wird, ist eine Schwierigkeit, auf die im nächsten Abschnitt eingegangen wird. Beim BOBYQA Algorithmus wird es dem Benutzer überlassen die Anzahl der Interpolationspunkte m aus dem Intervall $m \in [n+2, \frac{1}{2}(n+1)(n+2)]$ zu wählen. Die Matrix der Interpolationsbedingungen $M(\phi, Y)$ hat in diesem Fall mehr Spalten als Zeilen und die Koeffizienten des Interpolationspolynoms sind durch:

$$Q(y_i) = \sum_{k=1}^{q} a_k \phi_k(y_i) = f(y_i), \quad i = 1, ..., m, \tag{3.57}$$

definiert. Durch die geringere Anzahl an Interpolationspunkten ist das Interpolationsmodell nicht eindeutig lösbar. Um trotzdem eine eindeutige Lösung zu erhalten, wird ein weiteres unterbestimmtes Modell aufgestellt auf das im nächsten Abschnitt eingegangen wird.

3.2.3.5 Frobeniusnorm-Updating Prozedur

Die untere Schranke für den Fehler der Funktion und des Gradienten des quadratischen Interpolationsmodells hängt von der Norm der Hessematrix der Modellfunktion ab. Damit liegt es nahe ein unterbestimmtes Modell zu bauen, so dass die Hessematrix angemessen ist. Nun können die restlichen Freiheitsgrade von (3.57) so gewählt werden, dass die Frobeniusnorm der Hessematrix der Modellfunktion minimiert wird.

Powell führte eine Variante dieser Methode in [Pow 03] und [Pow 04a] ein und verwendet diese auch in seinem Algorithmus BOBYQA [Pow 09]. Hier wird nicht die Hessematrix direkt minimiert, sondern die Differenz der Hessematrizen der aktuellen und der vorherigen Iteration, d.h. $\|\nabla^2 Q_{k+1} - \nabla Q_k\|_F$, wobei die Frobeniusnorm einer Matrix wie folgt definiert ist:

$$\|\Theta\|_F = \sqrt{\sum_{i,j=1}^{n} \Theta_{ij}^2}, \qquad \Theta \in \mathbb{R}^{n \times n}. \tag{3.58}$$

Das quadratische Modell Q_{k+1} wird durch das Vorgängermodell Q_k bestimmt, wobei Q_{k+1} die Interpolationsbedingungen (3.47) für die neue Menge an Iterationspunkten erfüllen muss. Die restlichen Freiheitsgrade von Q_{k+1} werden durch die Minimierung der Frobeniusnorm von $\nabla^2 Q_{k+1} - \nabla^2 Q_k$ mit den Nebenbedingungen $Q_{k+1}(\hat{y}_i) = f(\hat{y}_i)$, für $i \in S$ genommen. So entsteht ein lineares Gleichungssystem, welches im Folgenden aufgestellt wird. Durch das Lösen dieses Systems wird das nächste quadratische Modell bestimmt.

Dazu sei $D(x)$ das folgende quadratische Modell:

$$D(x) = Q_{k+1}(x) - Q_k(x) = p + (x - x_0)^T q + \frac{1}{2}(x - x_0)^T F (x - x_0), \tag{3.59}$$

mit $p = Q_{k+1}(x_0) - Q_k(x_0)$, $q = \nabla Q_{k+1}(x_0) - \nabla Q_k(x_0)$ und $F = \nabla^2 Q_{k+1} - \nabla^2 Q_k$. Mit diesen Notationen lässt sich das Modell durch die Minimierung von

$$\frac{1}{4}\|\nabla^2 D\|_F^2 = \frac{1}{4}\|F\|_F^2 \tag{3.60}$$

unter den Interpolationsbedingungen (3.47) aktualisieren. Der Faktor $\frac{1}{4}$ wurde aus Annehmlichkeit hinzugefügt und beeinflusst die Lösung des Problems nicht.

Das Updatingproblem des quadratischen Modells wird nun zu einem konvexen, quadratischen Problem mit der Lagrangefunktion:

$$\mathcal{L} = \frac{1}{4}\sum_{i,j=1}^{n} F_{ij}^2 - \sum_{j=1}^{m} \lambda_j [D(\hat{y}_j) - f(\hat{y}_j) + Q_k(\hat{y}_j)] \tag{3.61}$$

transformiert, wobei λ_j die Lagrangemultiplikatoren sind. Für die Lösung des quadratischen Problems werden die partiellen Ableitungen von $\mathcal{L}$ nach den Parametern von $D(x)$ alle gleich Null

gesetzt. Dadurch werden die folgenden Beziehungen für die Lagrangemultiplikatoren gefolgert:

$$\sum_{j=1}^{m} \lambda_j = 0, \qquad \sum_{j=1}^{m} \lambda_j (\hat{y}_j - x_0) = 0, \tag{3.62}$$

$$F = \sum_{j=1}^{m} \lambda_j (\hat{y}_j - x_0)(\hat{y}_j - x_0)^T. \tag{3.63}$$

Diese Gleichungen entstehen durch das Differenzieren von $\mathcal{L}$ nach p, den Elementen von q und F. Mit Hilfe von (3.59) kann $Q_{k+1}(x)$ wie folgt bestimmt werden:

$$Q_{k+1}(x) = Q_k(x) + p + (x - x_0)^T q + \frac{1}{2}(x - x_0)^T F (x - x_0), \tag{3.64}$$

wobei F die Form (3.63) annimmt. Es zeigt sich, dass sich die Berechnung von Q_{k+1} auf die Berechnung von p, q und λ reduziert hat und damit auf die Bestimmung von $m+n+1$ Unbekannten. Durch das Hinzufügen von (3.64) in die Interpolationsbeschränkungen (3.47) folgt:

$$p + (\hat{y}_i - x_0)^T q + \frac{1}{2} \sum_{j=1}^{m} \lambda_j [(\hat{y}_i - x_0)^T (\hat{y}_j - x_0)]^2 = f(\hat{y}_i) - Q_k(\hat{y}_i), \quad i = 1, ..., m, \tag{3.65}$$

und damit das lineare Gleichungssystem mit den Parametern p, q und λ:

$$\left(\begin{array}{c|c} A & Y^T \\ \hline Y & 0 \end{array}\right) \left(\begin{array}{c} \lambda \\ \hline p \\ q \end{array}\right) = \left(\begin{array}{c} r \\ \hline 0 \end{array}\right), \qquad \begin{array}{l} \}m \\ \}n+1 \end{array} \tag{3.66}$$

mit $r_i = f(\hat{y}_i) - Q(\hat{y}_i)$, $A \in \mathbb{R}^{m \times m}$ ist eine symmetrische Matrix mit den Elementen:

$$A_{ij} = \frac{1}{2}[(\hat{y}_i - x_0)^T (\hat{y}_i - x_0)]^2, \qquad i, j \in S \tag{3.67}$$

und $Y \in \mathbb{R}^{(n+1) \times m}$ hat die Form:

$$Y = \begin{pmatrix} 1 & 1 & ... & ... & 1 \\ \hat{y}_1 - x_0 & \hat{y}_2 - x_0 & ... & ... & \hat{y}_m - x_0 \end{pmatrix}. \tag{3.68}$$

Die ersten m Zeile des Gleichungssystems sind in (3.65), die $(m+1)$-te Zeile und die letzten n Zeilen sind durch die Gleichungen in (3.62) gegeben. Dies vervollständigt die Herleitung des linearen Gleichungssystems.
Da p, q und λ die Unbekannten des linearen Gleichungssystems sind, ist die Matrix mit der tatsächlich gearbeitet wird die Inverse H mit der Form:

$$H = \left(\begin{array}{c|c} A & Y^T \\ \hline Y & 0 \end{array}\right)^{-1} = \left(\begin{array}{c|c} \Omega & \Xi^T \\ \hline \Xi & 0 \end{array}\right). \tag{3.69}$$

Die Submatrizen können mit dem Gauß-Jordan-Verfahren berechnet werden.

3.2.4 Coordinate-Descent Methode

Coordinate-Descent Algorithmen lösen Optimierungsaufgaben, indem sie sukzessive entlang einer Koordinatenrichtung bzw. einer Hyperebene optimieren. Das Verfahren ist eng verwandt mit dem Gauss-Seidel und dem SOR-Verfahren für das Lösen von linearen Gleichungssystemen. Hier wird nun auf die fundamentale Idee der Coordinate-Descent Methode eingegangen und schließlich die in dieser Arbeit verwendete Variante beschrieben. Des Weiteren wird die Konvergenzanalyse kurz behandelt.

3.2.4.1 Konzept des Coordinate-Descent Verfahrens

Das Verfahren ist eine iterative Methode, welche eine neue Iterierte durch die Fixierung der meißten Komponenten des Vektors $x \in \mathbb{R}^n$ und die Optimierung der Zielfunktion in Bezug auf die restlichen Komponenten bestimmt. Im nächsten Iterationsschritt werden andere Komponenten festgehalten, sodass die verbliebenen unfixierten Komponenten optimiert werden können. Sind alle Minima für die einzelnen Kompontenten berechnet, wird der Schritt wiederholt bis das Abbruchkriterium greift. Jedes dieser Subprobleme ist von der Dimension kleiner als das ursprüngliche Optimierungsproblem und somit üblicherweise einfacher lösbar.

Im einfachsten Fall werden alle bis auf eine Komponente festgehalten, während die freie Komponente optimiert wird. Algorithmus 3.2 verdeutlicht das Vorgehen.

Algorithmus 3.2 : Coordinate-Descent Methode [Tse 01]

Schritt 1: Initialisierung:
Wähle einen Startwert $x_0 = (x_1^0, ..., x_n^0)$
Schritt 2: Komponentenwahl:
Sei $x_k = (x_1^k, ..., x_n^k)$ gegeben. Wähle einen Index $j \in 1, ..., n$.
Schritt 3: Iteration:
Berechne die neue Iterierte $x_{k+1} = (x_1^{k+1}, ..., x_n^{k+1})$, sodass
$x_j^{k+1} = \arg\min_{y \in \mathbb{R}} f(x_1^{k+1}, ..., x_{j-1}^{k+1}, y, x_{j+1}^k, ..., x_n^k)$,
$x_s^{k+1} = x_s^k, \qquad$ für alle $s \neq j$.
Gehe zu 2.

Im zweiten Schritt wird die zu optimierende Komponente gewählt. Für diese Wahl gibt es verschiedenen Methoden. Zum einen können die Komponenten zyklisch nacheinander optimiert werden. Alternativ bietet sich auch eine zufällige Wahl der zu optimierenden Komponente an. Das Verfahren bricht durch das Erreichen einer bestimmten Genauigkeit bzw. der maximal gegebenen Anzahl an Funktionsauswertungen ab.

Das Coordinate-Descent Verfahren, welches in dieser Arbeit verwendet wird, optimiert mehrere Variablen gleichzeitig. Es wird ein einfacher zyklischer Durchlauf gewählt und die Coordinate-Descent Methode wird in den Optimierungprozess von BOBYQA und CMA-ES einbezogen. Als Abbruchkriterium für die Coordinate-Descent Methode dienen die Abbruckkriterien der Algorithmen BOBYQA und CMA-ES.

3.2.4.2 Konvergenz

Schon für ein einfaches dreidimensionales Beispiel kann gezeigt werden, dass das Verfahren nicht immer konvergiert. Powell zeigt in [Pow 73] für eine nicht-konvexe Funktion $f : \mathbb{R}^3 \to \mathbb{R}$ für ein zyklisches exaktes Coordinate-Descent Verfahren, dass die Methode nicht gegen einen stationären Punkt konvergiert. Sei die Funktion f gegeben durch:

$$f(x_1, x_2, x_3) = -x_1x_2 - x_2x_3 - x_1x_3 + \sum_{i=1}^{3}(|x_i| - 1)_+^2, \tag{3.70}$$

mit

$$(|x_i| - 1)_+ = \begin{cases} |x_i| - 1, & \text{falls } |x_i| \geq 1 \\ 0, & \text{falls } |x_i| < 1. \end{cases} \tag{3.71}$$

Die beiden Optima befinden sich an den Eckpunkten des Einheitswürfels $[-1, -1, -1]$ und $[1, 1, 1]$. Fällt die Wahl des Startpunktes beispielsweise auf $[-1.02, 1.01, -1.005]$, dann springt das Verfahren zwischen den sechs Ecken des Einheitswürfels, welche nicht optimal sind ([Pow 73]). Ist die Funktion nicht differenzierbar, so kann es auch passieren, dass das Verfahren in einem nicht-stationären Punkt stecken bleibt, auch wenn f konvex ist (vgl. [Tse 01]). Dennoch kann die Konvergenz des Verfahrens für den Fall, dass die Zielfunktion glatt und konvex ist, gezeigt werden (siehe zum Beispiel [Wri 15]).

3.2.5 Multilevel-Optimierung

In der Multilevel-Optimierung kommt eine Modellhierarchie der Zielfunktion, wobei die Modelle unterschiedliche Komplexitätsgrade aufweisen, zur Anwendung. Der Vorteil von Multilevelansätzen in der Optimierung ist, dass durch Vereinfachungen der Zielfunktion die Kosten für die Funktionsauswertungen reduziert werden. So können beispielsweise diskretisierte Differentialgleichungen mittels Multilevel-Optimierung gelöst werden, indem diese zunächst auf einem groben Gitter berechnet werden und dieses nach und nach feiner gewählt wird. Es besteht auch die Möglichkeit, dass zwischen den verschiedenen Gittertypen hin und her gesprungen wird. Da die Funktionsauswertung auf einem feinen Gitter sehr teuer ist, bietet ein grobes Gitter den Vorteil einer schnelleren Funktionsauswertung.

Der Hauptaufwand in der ableitungsfreien Optimierung liegt nicht in der Schrittberechnung des Verfahrens, sondern in der Funktionsauswertung. Da dies auf die beiden ausgewählten Verfahren zutrifft, wäre eine Strategie die Informationen von verschiedenen Level zu nutzten und damit das Modell zu bauen.

Sei $\{f_i\}_{i=1}^r$ eine Menge von verschiedenen Beschreibungen der Zielfunktion, wobei für jedes $f_i : \mathbb{R}^n \to \mathbb{R}$ gilt. Jede Funktion f_{i-1} ist einfacher auszuwerten als die Funktion f_i, für alle

$i = 1, ..., r$. Zur Vereinfachung werden hier nur zwei Level $i = 1, 2$ betrachtet. Die Zielfunktionen f_i beinhalten die Lösungen der linearen Gleichungssyteme der Spiegelfacette auf verschiedenen Gittervarianten. Sei x_1^* die Lösung des Gleichungssystems $A_1 x_1 = b_1$ resultierend aus dem FE-Modell mit grobem Gitter und x_2^* die Lösung des Systems $A_2 x_2 = b_2$ mit feinem Gitter. Der Fehler zwischen gröbem und feinem Gitter ergibt sich durch $e := x_1^* - x_2^*$. Wird die Lösung auf dem groben Gitter gefunden und ist der Fehler e bekannt, so kann die gesuchte Lösung x_2^* des feinen Gleichungssystems einfach durch $x_2^* = x_1^* - e$ bestimmt werden. Die FE-Verformungsberechnung der Spiegelfacette ist der ausschlaggebende Zeitfaktor für die Optimierung. Da die Auswertung des Spiegels auf einem gröberen Gitter schneller erfolgt, wird die Optimierung mit Hilfe dieses Verfahrens beschleunigt, indem der Optimierungsdurchlauf zunächst mit dem gröberen Modell erfolgt und anschließend auf das feine Gitter gesprungen wird. Die detaillierte Vorgehensweise in Bezug auf die genaue Aufgabenstellung in Kaptiel 5 beschrieben.

4 Stand der Technik

Um die lokalen Steigungsabweichungen einer Spiegelfacette im Labor oder in einem Solarfeld zu ermitteln, kann die Deflektometrie genutzt werden. Das Verfahren wird im Folgenden vorgestellt und erläutert. Ferner existieren bereits FE-Modelle der Innen- und Außenspiegel, um Faktoren, welche die Formgenauigkeit eines Parabolrinnenspiegels beschreiben, zu charakterisieren. Diese FE-Modelle werden im zweiten Teil des Kapitels vorgestellt. In dieser Arbeit werden diese Modelle genutzt, um von gemessenen Spiegelformabweichungen auf die Fehlstellungen der Padpositionen zu schließen.

4.1 Deflektometrische Messung

Deflektometrische Messungen werden genutzt, um Gestaltinformationen spiegelnder Oberflächen zu erhalten. Mit Hilfe dieses Verfahrens werden geometrische Fehler von reflektierenden Gegenständen identifiziert. Es kann damit die Krümmung eines zu untersuchenden Objektes ermittelt werden. Die Vorteile des Messprinzips im Vergleich zu anderen Verfahren liegen in seiner hohen Auswertegeschwindigkeit, Präzision und optischen Auflösung sowie der Möglichkeit der Anwendung in der laufenden Produktion. Im Bereich der Solarthermie wird das Verfahren der Deflektometrie überwiegend zur hochgenauen Vermessung der Formabweichung von Spiegelfacetten genutzt (siehe [Lü et al. 07a]). Im Test- und Qualifizierungszentrum für konzentrierende Solartechnik (QUARZ®) des DLR wurde eine solche deflektomertische Messmethode entwickelt.

Deflektometrische Messungen arbeiten mit der Projektion von phasenverschobenen sinusförmigen Linienmustern auf eine Leinwand. Das verzerrte Bild, dass durch die Reflexion dieser Muster an der Spiegeloberfläche entsteht, wird von einer Digitalkamera erfasst. Ein Algorithmus wertet das Bild aus, sodass es über die Kenntnisse des unverzerrten Musters möglich ist Aussagen über die Oberfläche der Spiegelfacette zu treffen (siehe [Wil 09]). Der Aufbau und das Prinzip einer Deflektometriemessung wird in Abbildung 4.1 verdeutlicht. Da die Position der Projektionswand, der Spiegelfacette und der Kamera bekannt sind, kann aufgrund des Reflexionsgesetzes auf die Flächennormalenvektoren geschlossen werden. Mittels eines Kodierungssystems kann jeder Kamerapixel, welcher einen bestimmten Punkt auf dem Spiegel beschreibt, einem Punkt auf der Leinwand zugeordnet werden, sodass es möglich ist die lokalen Steigungsabweichungen in x und in y für die ganze Spiegelfacette zu bestimmen. Diese können mit einer Genauigkeit von $\pm 0.5\,\mathrm{mrad}$ ermittelt werden.

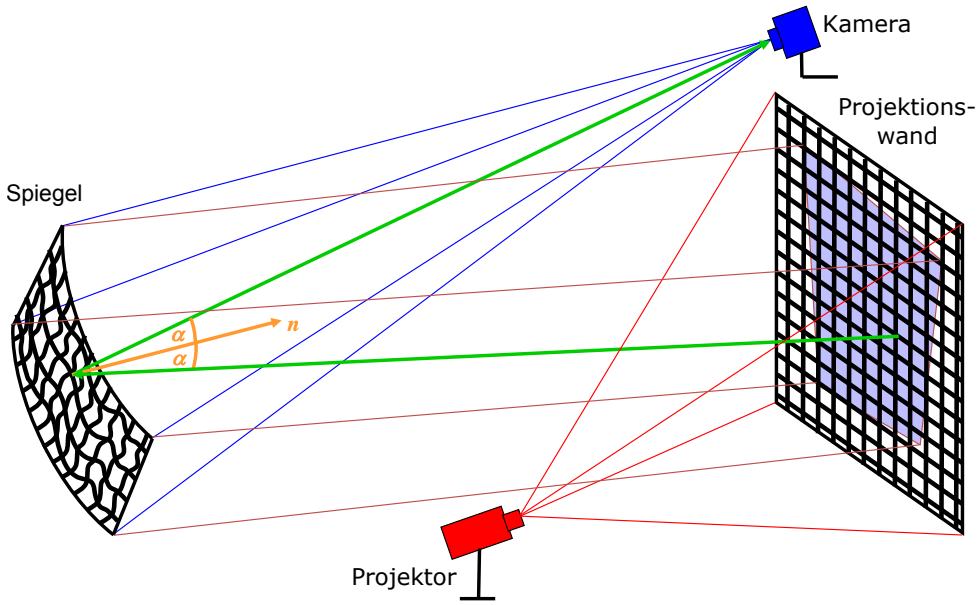


Abbildung 4.1: Grundprinzip der Deflektometrie zur Bestimmung von Spiegelformabweichungen [Lüp 12]

Abbildung 4.2 zeigt exemplarisch ein Resultat einer Deflektometriemessung eines RP3 Innenspiegels. Es ist ein Plot der lokalen Slope Deviation-Werte in gekrümmter x- und in ungekrümmter y-Richtung dargestellt. Die Abbildung zeigt eine Ansicht von oben auf die Spiegeloberfläche und stellt eine Projektion dar. Jedem Koordinatenpunkt auf der Oberfläche wird der entsprechende Wert der lokalen Steigungsabweichungen, welcher mittels Formel (2.3) bestimmt wird, zugeordnet und mit Hilfe der Plots grafisch veranschaulicht. Rote Bereiche beschreiben eine Rotation der Normalenvektoren nach außen im Vergleich zum Idealnormalenvektor und werden mit einem positiven Wert belegt, während blaue Bereiche auf Drehungen nach innen hinweisen und negativ

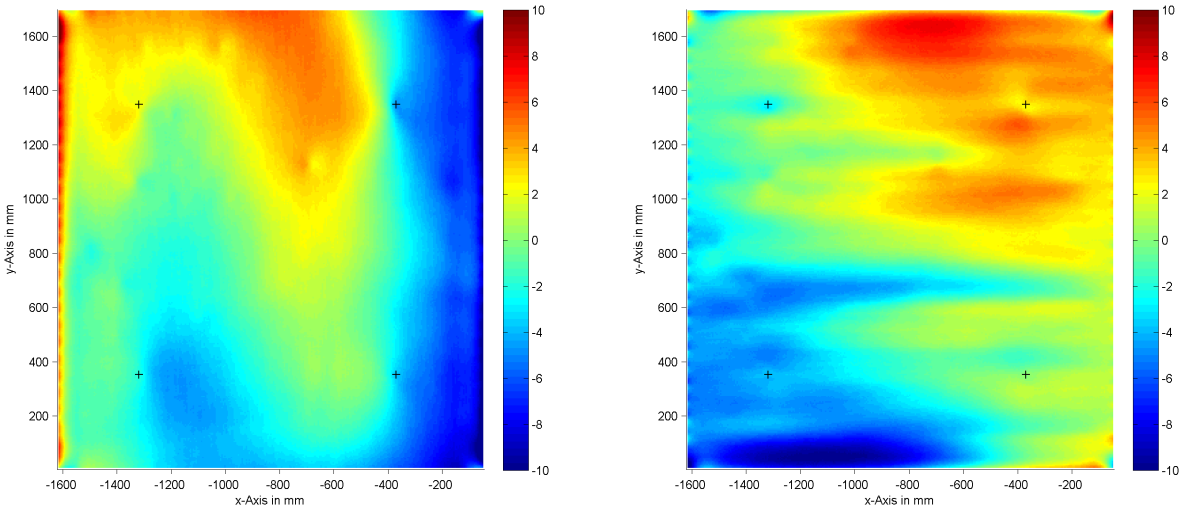


Abbildung 4.2: Ergebnisplots der lokalen Slope Deviation-Werte in x-Richtung (links) und in y-Richtung (rechts) einer Deflektometriemessung eines RP3-Innenspiegels mit SDx =3.5639 mrad

definiert sind, wie bereits in Abschnitt 2.2.2 erläutert und in Abbildung 2.7 dargestellt. Des Weiteren kann mittels der bekannten Werte der lokalen Steigungsabweichung die Slope Deviation SDx wie in Formel (2.4) berechnet werden. Für diesen Fall gilt SDx =3.564 mrad.

Wird nur das Ergebnis einer deflektometrischen Messung ausgewertet, wird deutlich, dass dieses ein Eindeutigkeitsproblem birgt. Durch Betrachten des vom Spiegel reflektierten Strahls, wie beispielsweise in Abbildung 4.1, zeigt sich, dass es entlang dessen hypothetisch unendlich viele Normalenvektoren gibt. Da nur durch die Normalenvektoren die Krümmung des Spiegels beschrieben wird, ist die genaue Lage der Spiegelfacette unklar. Folglich bleibt ein freier Parameter in den Ergebnissen. Um dennoch ein dreidimensionales Bild der Spiegeloberfläche zu erhalten, werden zusätzliche Informationen benötigt, welche durch die Lagepositionen von Projektionswand, Spiegel und Kamera zueinander gegeben werden. Es kann die Ortsinformation eines Punktes auf dem Spiegel betrachtet und von diesem ausgehend die Oberflächennormalenvektoren aufintegriert werden. Dadurch entsteht das dreidimensionale Spiegeloberflächenbild. Bei dieser Methode ist zu beachten, dass sich der Fehler bei der Integration vervielfacht je weiter die Entfernung vom Startpunkt ist.

4.2 FE-Modelle

Die FE-Spiegelmodelle, welche in dieser Arbeit genutzt werden, basieren auf den Modellierungen von Siw Meiser in [Mei 13] und der Weiterentwicklung dieser Modelle sowie Aufbereitung für die Optimierung durch Simon Schneider. Die Verformungsanalyse wird mittels des statischen Struktursimulationswerkzeugs der Finite-Elemente-Software Ansys ausgeführt. Diese Methode wird in Kapitel 3 vorgestellt. In [Mei 13] wird mit Hilfe der Analyse der Einfluss der Spiegelmontage und der Spiegelorientierung auf die Verformung des Spiegels hin erforscht und die resultierenden Formgenauigkeiten verglichen. In dieser Arbeit wird von einer vorhandenen Spiegelform auf die vorliegenden Bedingungen an den Aufhängepunkten des Spiegels geschlussfolgert. Insbesondere werden Rückschlüsse auf Fehlstellungen der Aufhängeelemente gemacht.

Die nächsten Unterkapitel beschreiben die für diese Arbeit verwendeten FE-Spiegelmodelle. Es wird kurz auf die gegebenen Daten und die verwendeten Annahmen eingegangen. Anschließend werden die verschiedenen Ansys-Spiegelmodelle vorgestellt. In dieser Arbeit dienen sie den Algorithmen als Optimierungsmodell. Die Optimierungsparameter werden als externe Verschiebungen in Ansys ausgewertet und die dadurch erzeugten Verformungen mittels Matlab ausgewertet. Es wird der RMS-Wert der Steigungsabweichungsdifferenzen berechnet, welcher die Zielfunktion des Optimierungsprozesses ist. Die folgenden Abschnitte orientieren sich soweit nicht anders vermerkt an [Mei 13].

4.2.1 Annahmen

Für die statische FE-Analyse wird die lineare Elastitzitätstheorie unter stationären Bedingungen berücksichtigt. Folglich werden nur linear elastische und kleine Deformationen in das Modell einbezogen. Des Weiteren werden reale Materialeigenschaften genutzt, jedoch Inhomogenitäten des Materials und der Geometrie vernachlässigt. Damit wird ein idealer Spiegel modelliert. Ferner werden keine Schrauben und sonstige Verbindungselemente im Modell integriert. Weitere Vereinfachungen werden in den folgenden Unterkapiteln diskutiert.

4.2.2 Modellbeschreibung

In dieser Arbeit fällt der Fokus der Betrachtung auf den Innenspiegel des Kollektors. Analoge Untersuchungen und Aussagen können auf den Außenspiegel übertragen werden. In Abschnitt 2.1.2 kann der Aufbau eines Kollektors nachgelesen werden. Abbildung 4.3 zeigt verschiedene Perspektiven des Innenspiegels mit den vier Aufhängepunkten sowie die Diskretisierung der Spiegelfacette, welche einer Modellierung mittels Ansys zugrunde liegt.

Die Geometrie des FE-Modells, das heißt die Geometrie der Spiegel, des Klebers und der Pads, wird direkt mit der Anwendung Ansys DesignModeler konstruiert. Die Spiegelfacette wird als mechanische Schale modelliert. Des Weiteren wird angenommen, dass die reflektierende Silberbeschichtung und die Schutzbeschichtungen keinen Einfluss auf das Verformungsverhalten des Spiegels haben. Aufgrund dessen bleiben diese im Modell unberücksichtigt, so dass der Spiegel nur aus 4 mm dickem Floatglas besteht. Die Aufhängepads werden durch ellipsenförmige, 12 mm dicke Körper modelliert, wobei die Metallhülsen mit Innengewinde, welche dafür sorgen, dass der Spiegel mit der Unterstruktur befestigt wird, nicht modelliert wird. Abbildung 4.4 zeigt die Position des Koordinatensystems für ein Parabolrinnenkollektormodul, wie bereits in Abschnitt 2.1.2 beschrieben.

In der Mitte der Padunterseiten wird jeweils ein Padkoordinatensystem definiert. Diese sind durch die lokale Steigung $\frac{\partial y}{\partial z}$ entlang der y-Achse geneigt. Die Positionen der Padkoordinatensysteme des RP3 Innenspiegels sind in Tabelle 4.1 dargestellt.

Tabelle 4.1: Positionen der Padkoordinatensysteme des RP3 Innenspiegels [Mei 13]

	x [mm]	y [mm]	z [mm]	$\frac{\partial y}{\partial z}$ [°]
Pad 1	−371.33	352	7.69	6.17
Pad 2	−371.33	1348	7.69	6.17
Pad 3	−1322.46	352	242.39	21.08
Pad 4	−1322.46	1348	242.39	21.08

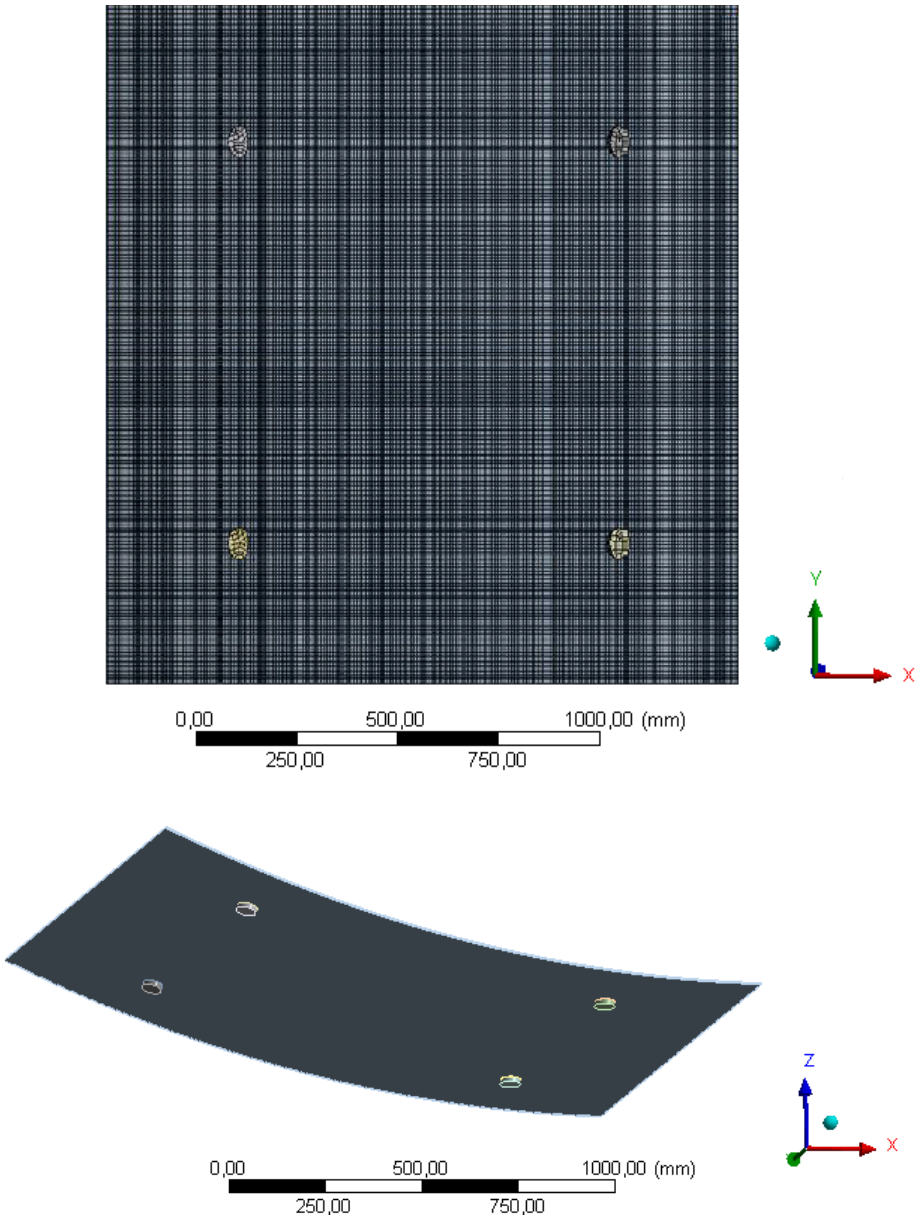


Abbildung 4.3: RP3-Innenspiegel mit vier Aufhängepunkten aus verschiedenen Perspektiven

Des Weiteren wird die Dichte, der Elastizitätsmodul und die Poissonzahl für die Materialien Glas, Keramik, Baustahl, Aluminium, sowie für den Silikonkleber mit in das Modell einbezogen. Die genauen Daten für die Materialeigenschaften sind in [Mei 13] aufgeführt.

Die Diskretisierung der Modelle geschieht in ANSYS und ist für die Spiegelfacette in Abbildung 4.3 veranschaulicht. Je feiner die Diskretisierung des FE-Modells, desto besser wird die Lösung approximiert, jedoch müssen zusätzliche Knoten gelöst werden. Dies ist mit einer höheren Rechenzeit verbunden. Eine Untersuchung der Diskretisierungskonvergenz führt zu einer Unterteilung von 200×200 Elementen für die Länge und Breite der Spiegelfacette (siehe [Kle 11]). Ferner ist die Definition eines strukturierten Netzes für den Spiegel wichtig, um bei der Berechnung des RMS-Wertes der Steigungsabweichungsdifferenzen von zwei Spiegelfacetten die gleichen Koordinatenpunkte miteinander zu vergleichen.

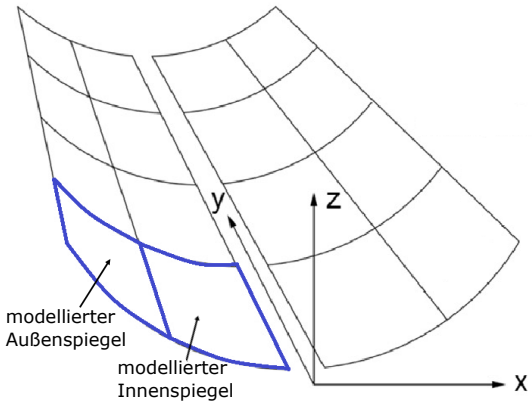


Abbildung 4.4: Koordinatensystem eines Parabolrinnenkollektormoduls [MLSP 14]

Außerdem spielt die Gravitation als Randbedingung für die Modelle eine wichtige Rolle, um das Eigengewicht der Spiegel zu berücksichtigen. Da die Spiegel im Lauf eines Tages der Sonne nachgeführt werden, ändert sich die Position der Spiegelfacetten über den Tag gesehen. Statt das Spiegelmodell in Bezug zum Koordinatensystem zu ändern, werden stattdessen die Gravitationsvektorkomponenten in Relation zur jeweiligen Spiegelstellung, welche mittels Winkel δ beschrieben wird, angepasst. Die Gravitation wird in $\frac{\text{mm}}{\text{s}^2}$ ausgedrückt, da die Spiegelgeometrie in mm gegeben ist. Da die Spiegel der Sonne nur einachsig nachgeführt werden, spielen hier lediglich die Gravitationskomponenten in x und z eine Rolle. Für Spiegel, welche sich auf der linken Seite der Parabel befinden, das heißt in negativer x-Richtung, gilt:

$$g_x = -9810\frac{\text{mm}}{\text{s}^2} \cdot \sin\delta, \qquad g_z = 9810\frac{\text{mm}}{\text{s}^2} \cdot \cos\delta. \tag{4.1}$$

Befindet sich der Spiegel rechts vom Ursprung, so ist die x-Komponente der Gravitation mit einem positven Vorzeichen versehen. Tabelle 4.2 zeigt die Gravitationskomponenten g_x und g_z für einige Winkel. Mittels Formel (4.1) kann jedoch auch jede andere Spiegelposition bestimmt werden. Hierbei ist mit der 0 °-Kollektorstellung die Zenitstellung des Kollektors gemeint. Das heißt, die Kollektorposition mit der Kollektoröffnung nach oben. Die Gravitationskomponenten werden in einem Gravitationsvektor $g = [g_x, g_y, g_z]$ zusammengefasst.

Tabelle 4.2: Gravitationskomponenten für verschiedene Spiegelwinkel des Innenspiegel rechts vom Scheitelpunkt [Mei 13]

Winkel [°]	g_x $[\frac{\text{mm}}{\text{s}^2}]$	g_z $[\frac{\text{mm}}{\text{s}^2}]$
0	0	9810
45	6936.7	6936.7
90	−9810	0
horizontal: 13.86	−2350	9524

4.2.2.1 Spiegelmodelle für die Optimierung

In [Mei 13] werden die Spiegelmodelle mit Hilfe von fixierten Lagerungen an den Padunterseiten festgehalten. In der vorliegenden Arbeit werden statt der fixierten Lagerungen externe Verschiebungen (Remote Displacement) an der Klebstoff- bzw. Padunterseite definiert. Mit Hilfe von externen Verschiebungen als Randbedingungen ist auch eine Fixierung des FE-Modells gewährleistet. Durch diese Randbedingungen werden Translationen in Richtung x, y und z möglich, sowie Rotationen um diese Achsen berücksichtigt. Durch die Translationen und Rotationen der vier Aufhängepunkte ergeben sich 24 Freiheitsgrade. Des Weiteren besteht die Möglichkeit einzelne Parameter zu fixieren, um verschiedene Szenarien zu testen. Mit Hilfe dieses Modells kann auch eine Lösungsreferenz für ein Testproblem konstruiert werden, welche der Optimierer finden muss. Das FE-Modell kann aber auch als Optimierungsmodell dienen, welches so lange verformt wird, bis es mit der Referenz übereinstimmt. So ist es möglich Verformungen, welche durch den Einfluss der Kollektorunterstruktur auftreten, wie beispielsweise Montageungenauigkeiten nachzuformen.

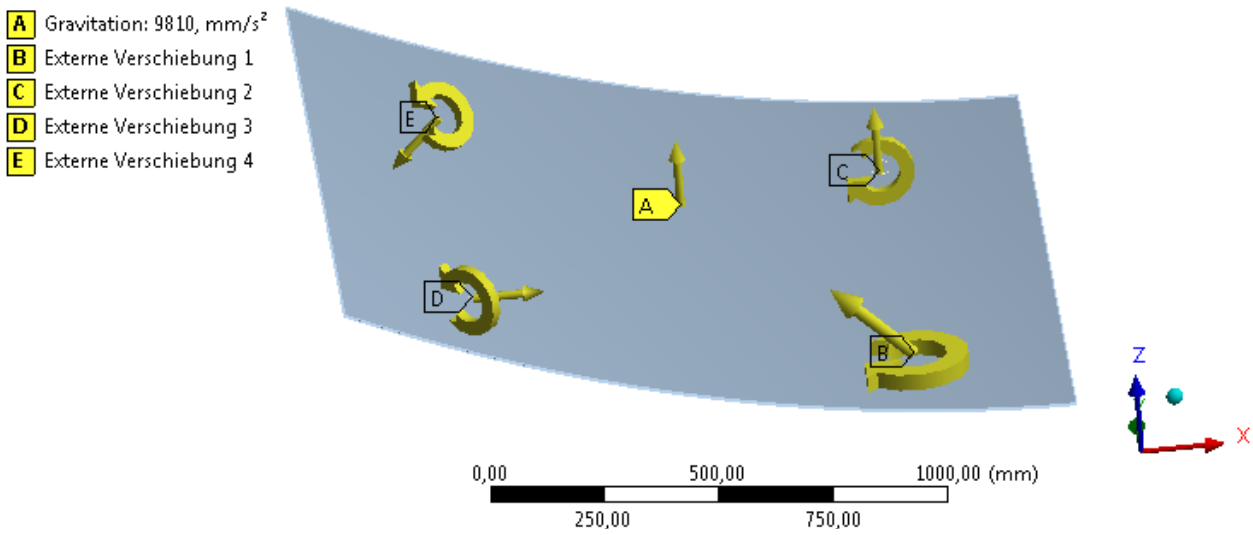


Abbildung 4.5: Innenspiegel mit externen Verschiebungen, d.h. Translationen und Rotationen

Die externen Verschiebungen in x-, y- und z-Richtung werden in mm angegeben. Die Translationen um die x-, y- und z-Achse sind in ANSYS in rad gegeben. Es ist jedoch gewünscht, dass die in MATLAB ein- und ausgegebenen Tranlationen in mrad betrachtet werden, da die Verzerrungen an den Spiegelaufhängungen in der Wirklichkeit Verkippungen im mrad-Bereich unterliegen. Die Verschiebungswerte der Spiegelrückseite werden in Form einer Textdatei exportiert und diese anschließend von MATLAB eingelesen. Hier werden darauf folgend die Oberflächennormalenvektoren berechnet und mit den Idealnormalenvektoren verglichen, so dass sich die lokalen Differenzen ergeben.

Klebstoff-Spiegelmodell

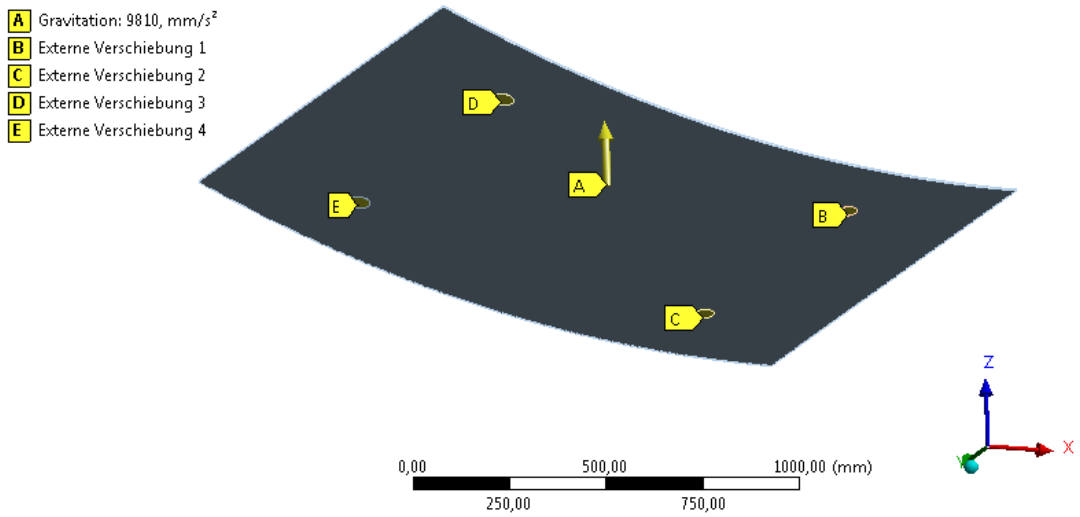


Abbildung 4.6: Rückseite des FE-Modells des RP3 Innenspiegels mit Klebstoffschicht, sowie externen Verschiebungen

Für die ersten Anwendungen der Algorithmen auf die Problemstellung wird das Spiegelmodell mit der Klebstoffschicht betrachtet. An den Unterseiten des Klebers werden die externen Verschiebungen, wie in Abbildung 4.6 verdeutlicht, definiert. Weitere Vereinfachungen für erste Tests werden durch eine gröbere Diskretisierung der Spiegelfacette realisiert. Durch die Reduktion der Knotenanzahl des Spiegels ist eine schnellere Funktionsauswertung möglich.

Pad-Spiegelmodell

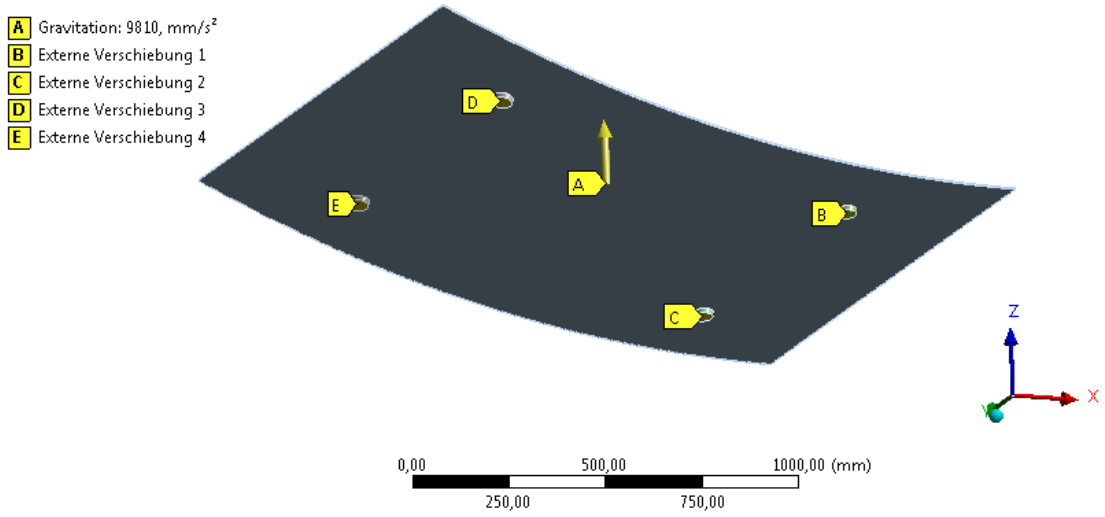


Abbildung 4.7: Rückseite des FE-Modells des RP3 Innenspiegels mit Pads, sowie externen Verschiebungen

In diesem FE-Spiegelmodell werden Schrägstellungen und Verschiebungen der Pads mittels externer Verschiebungen an den jeweiligen Padunterseiten modelliert. Dies ist in Abbildung 4.7 veranschaulicht. Hier dient das Pad als Schnittstelle zur nicht-modellierten Trägerkonstruktion. So werden die externen Verschiebungen, die durch die Unterkonstruktion verursachten Verzerrungen, welche auf das Pad und somit auch auf die Spiegelfacette wirken, simuliert.

4.2.2.2 Modell mit Laborunterstruktur

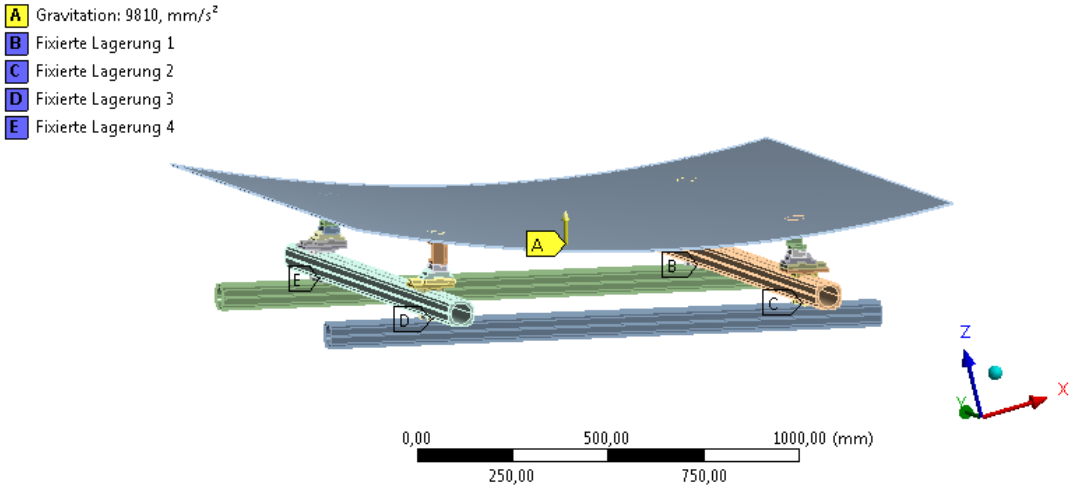


Abbildung 4.8: FE-Modell des RP3 Innenspiegels fixiert an die Laborunterkonstruktur

Die Laborunterstruktur dient als Halterung für den Spiegel bei deflektometrischen Messungen im QUARZ® Labor des Institutes für Solarforschung und wird in [Mei 13] modelliert. Der Tragerahmen besteht aus zwei Paaren parallel verlaufender Aluminiumbalken, welche in gekrümmter und ungekrümmter Richtung des Spiegels ausgerichtet sind. Sie sind durch Montage-Kreuzverbindungen miteinander verbunden. Präzise gefertigte Winkelklammern aus Stahl, welche wesentlich steifer als die Klammern im Kollektor sowie anderes geformt sind, sorgen für einen gezielt konstruierten Neigungswinkel der zu testenden Spiegel an den vier Aufhängepunkten. Die Winkelklammern sind durch Verbindungsstücke und Präzisionsschlitten mit den Balken verbunden. Unter Verwendung der Laborunterstruktur kann eine Verstellung bzw. falsche Ausrichtung des Stützgerüstes ausgeschlossen werden, so dass die Ursachen möglicher Deformationen der getesteten Spiegel auf Designfehler oder anderer Faktoren zurückzuführen sind. Abbildung 4.8 zeigt das FE-Modell des RP3 Innenspiegels, welcher an die Laborunterkonstruktion fixiert ist. Des Weiteren liegt der Spiegel horizontal auf dem Tragerahmen. Die Randbedinungen für das Modell sind durch fixierte Lagerungen an den Kreuzverbindungen gegeben. Das FE-Modell dient in dieser Arbeit als Referenzlösung. In [Mei 13] wird für das Spiegelmodell mit Laborunterstruktur gezeigt, dass das FE-Modell das reale Verformungsverhalten nachbildet, indem reale Deflektometriedaten mit den simulierten Steigungsabweichungen verglichen werden. Der Optimierungsalgorithmus wird in dieser Arbeit auf das validierte Modell angewendet. Dadurch wird ein realistisches Testproblem konstruiert, welches mit Hilfe der Optimierungsverfahren gelöst wird.

5 Methoden

Zuerst wird der allgemeine Optimierungsprozess, welcher von Spiegelformmessungen auf die ursächlichen Translationen und Rotationen an den Spiegelaufhängungen schließt, beschrieben. Des Weiteren werden verschiedene Sensitivitätsanalysen, eine Untersuchung bzgl. der Diskretisierung sowie ein Hybridalgorithmus aus CMA-ES und BOBYQA vorgestellt. Es wird auf eine Variation der Zielfunktion, Reduktion der Optimierungsparameter und auf die richtige Skalierung dieser eingegangen. Ferner wird das Coordinate-Descent Verfahren im Zusammenhang mit der Problemstellung behandelt. Schließlich werden die Vorgehensweisen in Bezug auf die Laborunterstruktur und die reale Spiegelmessung präsentiert.

5.1 Optimierungsprozess

In diesem Abschnitt wird zunächst mittels Optimierungsmodell und Referenz das Ziel der Optimierung sowie das allgemeine Optimierungsprozedere mit der Ansteuerung von Ansys mittels Matlab erläutert. Anschließend wird die Zielfunktion mathematisch aufgestellt, die Wahl der Optimierer und die Implementierung beschrieben.

5.1.1 Optimierungs- und Referenzmodell

Um die ursächlichen Verschiebungen und Verkippungen an den Spiegelaufhängungen aus Spiegelformmessungen herzuleiten, werden die in [Mei 13] konstruierten und in Abschnitt 4.2.2.1 vorgestellten FE-Spiegelmodelle verwendet. Die untere Ellipsenfläche des jeweiligen Montagepads bzw. der jeweiligen Klebstoffschicht ist die Schnittstelle zur nicht-modellierten Unterkonstruktion und dient als Stelle, an der die externe Verschiebung greift. Details dazu finden sich in Abschnitt 4.2. Abbildung 5.1 verdeutlicht die allgemeine Idee der Optimierung. Rechts ist das Ergebnis einer deflektometrischen Messung in Form eines Plots der Slope Deviation in x aufgezeigt. Es sind starke Verformungen im oberen Teil sowie im rechten Bereich des Spiegels zu erkennen. Auf der linken Seite ist der Slope Deviation Plot in x einer unverformten Spiegelfacette dargestellt. An den Aufhängepunkten werden die externen Verschiebungen angelegt und variiert, bis der Slope Deviation Plot auf der linken Seite mit dem Plot rechts übereinstimmt. Aufgrund dessen wird der Slope Deviation Plot links als Optimierungsmodell und der Plot auf der rechten Seite als Referenz, welches das Optimierungsmodell nachbildet, bezeichnet. Als Referenz kann

nicht nur das Resultat einer Deflektometriemessung dienen. Es besteht auch die Möglichkeit ein Testproblem aus einem FE-Spiegelmodell zu konstruieren. Beim erstellten Testproblem sind die Translationen und Rotationen bekannt, sodass untersucht werden kann, wie genau das Optimierungsmodell die Referenz annähert. △SDx bezeichnet das gewichtete quadratische Mittel der Steigungsabweichungsdifferenzen von Optimierungsmodell und Referenz (siehe Formel (2.5)). Das quadratische Mittel wird als Zielfunktion für die Optimierung gewählt. Je ähnlicher sich die beiden Spiegelfacetten sind, desto kleiner ist der △SDx-Wert.

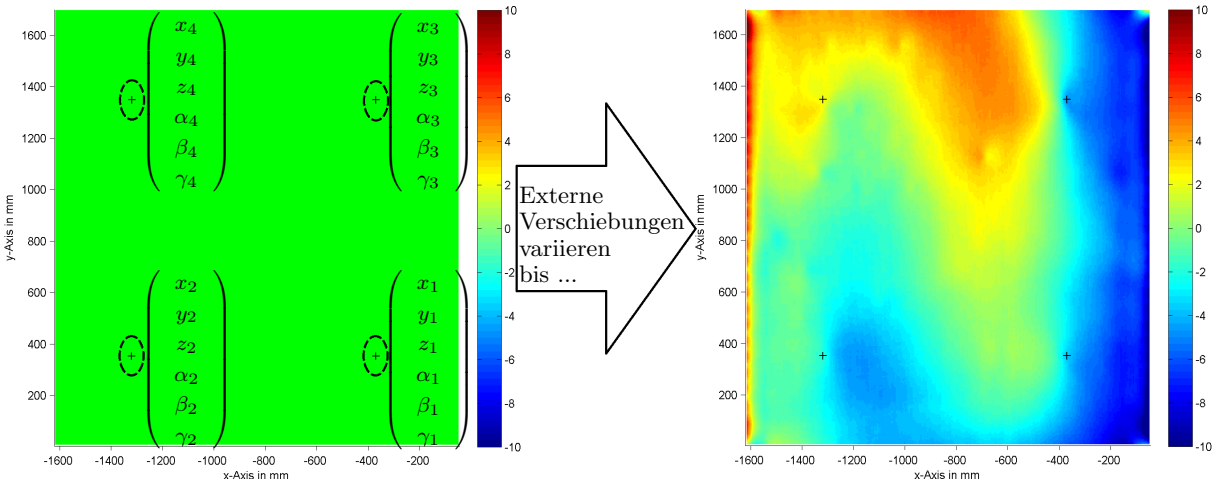


Abbildung 5.1: Grundsätzliche Methodik der Optimierung: Anlegen und variieren von externen Verschiebungen an das Optimierungsmodell (links), um die Referenz (rechts) zu erhalten

Anzumerken ist, dass neben den externen Verschiebungen auch die Gravitation im Optimierungsmodell berücksichtigt wird. Die vier Aufhängungen werden der Übersicht halber durchnummeriert, wie in Abbildung 5.1 gezeigt. Montagepad Nummer 1 befindet sich bei den Slope Deviation Plots rechts unten. Pad 2 liegt links unten, rechts oben befindet sich Aufhängepunkt 3. Links oben ist das Montagepad 4 angebracht. Der Vektor der externen Verschiebungen an jedem Pad ist durch $[x_i, y_i, z_i, \alpha_i, \beta_i, \gamma_i]^T$, für Pad $i = 1, 2, 3, 4$ gegeben. Hier sind x_i, y_i und z_i die Verschiebungen entlang der x-, y- und z-Achse sowie α_i, β_i und γ_i jeweils die Rotationen um die x-, y- bzw. z-Achse von Pad i, $i = 1, 2, 3, 4$. Die externen Verschiebungen an den Aufhängungen sind die Eingangsparameter für die Simulation einer Spiegelfacette mittels Ansys und dienen als Optimierungsparameter für die Optimierung. Ferner ist das FE-Spiegelmodell mit Hilfe der externen Verschiebungen durch extern vorgegebene Randbedingungen fixiert.

5.1.2 Grundprinzip des Optimierungsprozesses

Die Idee der Nachbildung der Spiegelform mittels Optimierungsalgorithmen entsteht im Rahmen der Dissertation von Simon Schneider, welche voraussichtlich 2016 erscheint.

Nachfolgend wird der allgemeine Optimierungsprozess erläutert. In ANSYS werden Parameter von existierenden FE-Modellen eines RP3-Innenspiegels bestimmt, um parametrische Studien zu realisieren. Diese Parameter sind durch Verschiebungen und Rotationen an den vier Spiegelaufhängungen gegeben und dienen als Unbekannte für die Optimierung. Sie werden in einem Parameterset zusammengefasst. Die Übergabe des Parametersets in ANSYS und Ansteuerung dieser Simulationssoftware erfolgt mittels durch Simon Schneider erstellte Schnittstelle in MATLAB. Damit kann die Verformung der Spiegelfacette mit Hilfe der FE-Analyse in ANSYS bestimmt werden. Die berechneten Verschiebungen der Knoten in x-, y- und z-Richtung des FE-Spiegelmodells werden MATLAB übergeben, sodass im Anschluss durch eine bereits bestehende MATLAB-Routine aus den verformten Knoten die Berechnung der lokalen Steigungsabweichungen folgt. Damit kann die Abweichung zwischen Optimierungsmodell und Referenz, wie in Abschnitt 5.1.1 beschrieben, bestimmt werden. Der berechnete △SDx-Wert von Optimierungs- und Referenzmodell wird dem Optimierer als Zielfunktionswert für das vorher übergebene Parameterset zugewiesen. Durch den Optimierungsprozess wird ein neues Set von Parametern bestimmt, welches dann wieder ANSYS übergeben wird. Der Vorgang wird im Idealfall solange weitergeführt, bis die Spiegelverformungen des Optimierungsmodells mit denen des Referenzmodells übereinstimmen. Folglich, wenn der RMS-Wert der Differenzen der lokalen Slope Deviations △SDx gegen null läuft. Der Algorithmus bricht mit einer gegebenen Genauigkeit ab, bzw. stoppt mit dem Erreichen der maximalen Anzahl an Iterationen. Das beschriebene Schema ist in Abbildung 5.2 gezeigt.

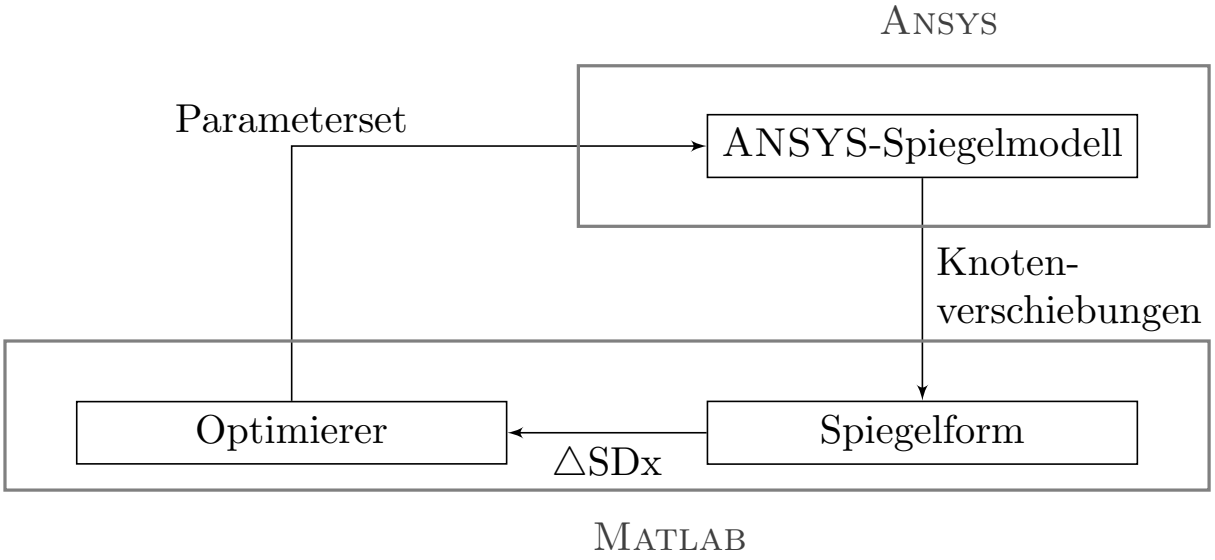


Abbildung 5.2: Grundprinzip des Optimierungsprozesses

5.1.3 Wahl der Optimierungsalgorithmen

Es existieren ableitungsbasierte und ableitungsfreie Verfahren für die Lösung von Optimierungsproblemen. Bei dem in dieser Arbeit untersuchten Problem steht kein analytischer Wert des Gradienten zur Verfügung, da die Modellierung und Diskretisierung der Spiegelfacette in Ansys erfolgt. Außerdem sind die für das automatische Differenzieren benötigten Informationen durch das Softwarepaket Ansys nicht gegeben, da das kommerzielle Simulationsprogramm keinen expliziten Zugriff auf dessen Komponenten erlaubt.

Die numerische Berechnung von Ableitungen beispielsweise mittels Differenzenquotienten ist sehr rechenaufwändig. Für die Gradientenberechnung werden zusätzliche Funktionsauswertungen benötigt. Für den Gradienten g in einem n-dimensionalen Raum gilt:

$$g_i(x) = \frac{f(x + \delta x_i) - f(x - \delta x_i)}{2\delta_i}, \qquad i = 1, ..., n, \tag{5.1}$$

wobei $\delta x_i = [0...0\,\delta_i\,0...0]^T \in \mathbb{R}^n$ ist. Damit müssen für die Berechnung eines einzelnen Gradienten $2n$ Funktionsauswertungen gemacht werden. Die Berechnung der Hessematrix würde weitere Funktionsauswertungen benötigen. Aufgrund der hier vorliegenden teuren Auswerteprozedur wird dieses Verfahren abgelehnt.

Darüber hinaus soll es dem Anwender möglich sein mechanisches Vorwissen in dem Optimierungsprozess zu berücksichtigen, sodass die Suche auf einen logischen Bereich eingegrenzt werden kann. Deshalb sollen die Algorithmen mit Box-Beschränkungen $a \leq x \leq b$ umgehen können. Aus diesen Gründen fällt die Wahl auf zwei ableitungsfreie Standardverfahren aus dem Bereich der modellbasierten und populationsbasierten Verfahren. Der BOBYQA Algorithmus (siehe Abschnitt 3.2.3) aus der Klasse der modellbasierten Methoden approximiert das Optimierungsmodell mittels eines quadratischen Modells und der populationsbasierte CMA-ES Algorithmus aus Abschnitt 3.2.2 macht sich die Mechanismen der biologischen Evolution zunutze. Diese beiden Verfahren werden auf die Problemstellung angewendet.

5.1.4 Zielfunktion

Die oben erläuterte Idee wird nun mathematisch beschrieben und als Optimierungsproblem formuliert. Es sind die 24 Parameter für die Optimierung gegeben, welche die Translationen und Rotationen an den vier Aufhängepunkten beschreiben. Damit ergibt sich:

$$\begin{aligned} f : \mathbb{R}^{24} &\to \mathbb{R} \\ (x_1, y_1, z_1, \alpha_1, \beta_1, \gamma_1, x_2, \cdots, \gamma_2, x_3, \cdots, \gamma_3, x_4, \cdots, \gamma_4)^T &\mapsto f(x_1, y_1, z_1, \alpha_1, \beta_1, \gamma_1, \cdots, \gamma_4), \end{aligned} \tag{5.2}$$

wobei f die Zielfunktion des Problems repräsentiert und hier eine Black-Box Funktion darstellt, da die genaue Berechnung dieser unbekannt ist. Die Optimierungsparameter werden an Ansys übergeben. Dort wird die Verformung der Spiegelfacette bestimmt und anschließend werden die

Knotenpunkte wieder an MATLAB übergeben. Mit Hilfe der Knoten des Optimierungsmodells und der bereits bekannten Lage der Knoten des Referenzmodells werden nun die lokalen Slope Deviations in x berechnet. Anschließend wird, wie bereits in Formel (2.5) beschrieben, der RMS-Wert der Steigungsabweichungsdifferenzen ermittelt:

$$f(x_1, \cdots, \gamma_4) = \triangle\text{SDx} := \sqrt{\sum_{k=1}^{p} \left((sdx_{2_k} - sdx_{1_k})^2 \frac{a_k}{A_{tot}} \right)}, \tag{5.3}$$

wobei sdx_{1_k} und sdx_{2_k} die lokalen Slope Deviation an den Punkten $k = 1, ..., p$ von Referenz und Optimierungsmodell darstellen. Die Gleichheit der beiden Spiegelfacetten soll erreicht werden. Das heißt, das Optimierungsmodell soll die Referenz so gut wie möglich nachahmen. Um dies zu gewährleisten muss die Funktion (5.3) minimiert werden:

$$\begin{aligned} &\min_{(x_1,\cdots,\gamma_4)^T \in \mathbb{R}^{24}} f(x_1, \cdots, \gamma_4), \quad \text{sodass} \\ &a_1 \leq x_1 \leq b_1 \\ &a_2 \leq y_1 \leq b_2 \\ &a_3 \leq z_1 \leq b_3 \\ &\vdots \\ &a_{23} \leq \beta_4 \leq b_{23} \\ &a_{24} \leq \gamma_4 \leq b_{24}, \end{aligned} \tag{5.4}$$

wobei $a_i, b_i \in \mathbb{R}$ für $i = 1, ..., 24$ die Grenzen der Optimierungsvariablen darstellen. Da die maximalen Translationen im Bereich $\pm 5\,\text{mm}$ und die maximalen Rotationsgrenzen zwischen $\pm 5\,\text{mrad}$ liegen, ist eine geeignete Wahl für $a_i = -5$ und $b_i = 5$, $i = 1, ..., 24$. Jedoch ist die Wahl der Grenzen für jeden Optimierungsdurchlauf neu wählbar.

Es besteht für erste Tests die Möglichkeit das Problem für weniger Dimensionen zu betrachten, indem ganze Aufhängepunkte oder einzelne Rotations- bzw. Translationsparameter fixiert werden und damit nicht in den Optimierungsprozess eingehen. Mit Hilfe dieser Methode werden die Algorithmen zunächst an einfachen Problemen getestet und eventuell angepasst.

5.1.5 Implementierung

Da sich bereits zu Beginn der Masterarbeit herausgestellt hat, dass eine Auswertung in ANSYS-Workbench, der graphischen Benutzeroberfläche von ANSYS, eine lange Zeit in Anspruch nimmt, wurde von Simon Schneider ein Skript zur Ansteuerung von ANSYS-Classic, der (M)APDL - ANSYS Parametric Design Language, in MATLAB geschrieben. Dadurch reduziert sich die Zeit für eine Funktionsauswertung eines FE-Modells mit einer Diskretisierung von 50×50 Elementen von 90 auf weniger als 7 sec. Durch die schnellere Auswerteprozedur ist eine Optimierung mit 24 Parametern in angemessener Zeit möglich.

Für die Optimierung werden zwei Algorithmen ausgewählt. Eine MATLAB-Implementierung des CMA-ES Verfahrens findet sich auf der Internetseite [Han 12] von Hansen selbst. Für den BOBYQA Algorithmus wurde eine C++ Implementierung aus der Open-Source-Software-Library `dlib` von D. King verwendet, welche mittels MATLAB-Funktion `mex` und `bobyqa.m` von U. Römer für eine Anwendung in MATLAB zur Verfügung gestellt wird (vergleiche [KR 14]). Die Funktion `bobyqa.m` ruft die `mex`-Funktion auf, welche eine Gateway-Funktion ist, die einen C++ Code in eine `Mex`-Datei kompiliert. Diese `Mex`-Datei ist dann auch von MATLAB abrufbar. Des Weiteren werden die Zielfunktionswerte in `bobyqa.m` und damit in MATLAB berechnet. Die MATLAB-Files, die `dlib` Library sowie eine Erklärung zur Vorbereitung für die erste Verwendung finden sich in [KR 14].

Die betrachteten ANSYS-Spiegelmodelle werden zunächst aufbereitet, indem diese in ANSYS Workbench betrachtet und externe Verschiebungen an den Pad- bzw. Klebstoffunterseiten definiert werden. Anschließend wird eine Eingabedatei mit der Endung `.dat` erzeugt. In diesem File befinden sich die Geometriedaten, die Gravitationskomponenten sowie die externen Verschiebungen. Direkt in der Eingabedatei können nun die für die Optimierung verwendeten Parameter gekennzeichnet werden. Dies geschieht durch eine fortlaufende Nummerierung der Gravitationskomponenten und Optimierungsparameter. Die Gravitationskomponenten werden mittels der willkürlichen Werte 9991, 9992 und 9993 kodiert, während die zu optimierenden Parameter die nachfolgenden Zahlen zugewiesen bekommen. Nicht zu optimierende Translationen und Rotationen werden auf null bzw. auf einen in Zusammenhang des jeweiligen Optimierungsproblems sinnvollen Wert gesetzt. Des Weiteren wird eine Excel-Tabelle erzeugt, welche die zu optimierenden Parameter, die entsprechende Einheit und die in der Eingabedatei zugewiesenen willkürlichen Zahlenwerte enthält.

Die Algorithmen werden in ein MATLAB-Skript eingefügt. In diesem findet unter anderem das Einlesen einer Konfigurationsdatei statt. Ferner wird die Konfigurationsdatei `configfile.m` so erweitert, dass die Gravitation g, der Startwert x_0, die Optimierungsgrenzen a und b, die Wahl des Optimierers und die zugehörigen Parameter, wie beispielsweise die Abbruchkriterien, berücksichtigt werden. Nach dem Start des Skripts wird mit dem gewählten Algorithmus verfahren. Den Algorithmen wird der Startwert x_0, die Grenzen a und b, die Startschrittweite σ im CMA-ES-Fall und ρ_{beg} beim BOBYQA Algorithmus, die maximale Anzahl an Funktionsauswertungen und der Parameter *handles*, der unter anderem die Daten aus der Konfigurationsdatei und die Werte der Excel-Tabelle enthält, übergeben. Für BOBYQA wird außerdem ein finaler Trust-Region Radius gefordert, welcher als Abbruchkriterium dient. Des Weiteren besteht die Möglichkeit die Anzahl der Interpolationspunkte zu variieren. Beim CMA-ES Algorithmus kann die Populationsgröße λ verändert sowie als Abbruchkriterium eine untere Grenze für den Zielfunktionswert definiert werden. Nach der Optimierung werden in beiden Fällen das berechnete Optimum x_* sowie der zugehörige Zielfunktionswert $f(x_*)$ ausgegeben. Genauere Angaben zu den Ein- und Ausgabewerten können im Anhang in Abschnitt A.3 nachgelesen werden.

Die Funktionsauswertung der Optimierung findet mit Hilfe der MATLAB-Funktion `Funktionsauswertung.m` (siehe 5.1) statt. Als Eingabewerte dienen die aktuelle Iterierte vX_opt so-

Algorithmus 5.1 : Funktionsauswertung.m

```
Funktion: [ nF_opt, handles ] = Funktionsauswertung( vX_opt, handles )
  for i = 1 : length(vX_opt) do
    handles.ConfigParams.ANSYS.Parameter(3 + i).value = mat2str(vX_opt(i));
  end
  handles = FiniteElementAnalysis_CE(handles);
  handles = MirrorShape_CE(handles);
  switch (handles.ConfigParams.comparisonType)
  case 'CompareTwoCases':
    nF_opt = abs(handles.Variables.SDx.FEM2toFEM1(:, end));
  case 'AddMeasData2FEMResultsAndCompareWithMeasurement':
    nF_opt = abs(handles.Variables.SDx.MeasPlusFEMtoMeas2);
  end
```

wie der Parameter *handles*. Der berechnete Zielfunktionswert $nF_opt = \triangle$SDx zur Iterierten vX_opt und die *handles*-Struktur sind die Ausgabewerte der Funktion. Die Ansteuerung von Ansys mittels Matlab existiert bereits und wird nun in den Optimierungsprozess integriert, sodass die zu optimierenden Parameter automatisch an Ansys übergeben werden. Die eingelesenen Werte aus der Excel-Tabelle werden nun durch die Optimierungsparameter an der Stelle $handles.ConfigParams.ANSYS.Parameter$ überspeichert. Mittels der Matlab-Funktion $FiniteElementAnalysis\_CE(handles)$ werden die Parameter anschließend Ansys übergeben, indem eine Suche der Zahlenwerte in der Ansys-Eingabedatei stattfindet. Dann wird eine zweite Eingabedatei erzeugt, welche die Werte durch das aus der Optimierung hervorgegangene Parameterset ersetzt. Auch die Gravitationskomponenten werden aus der Konfigurationsdatei ausgelesen und an den entsprechenden Stellen gespeichert. Der Aufruf der Ansys-Eingabedatei erzeugt eine Datei, welche die Knotenverschiebungen der Spiegelfacette enthält. Diese wiederum wird in Matlab eingelesen und es findet die FE-Analyse der Spiegelfacette statt. Im Anschluss werden mit der Funktion $MirrorShape\_CE(handles)$ die lokalen Steigungsabweichungen sowie der RMS-Wert der lokalen Steigungsabweichungsdifferenzen von zwei Spiegelmodellen bestimmt. Nachfolgend wird für die Referenz zwischen FE-Modell und Resultat einer Deflektometriemessung unterschieden, da diese verschiedene Auswerteprozeduren beinhalten. Ist die Referenz durch ein FE-Spiegelmodell gegeben, wird der Fall $'CompareTwoCases'$, andernfalls $'AddMeasData2FEMResultsAndCompareWithMeasurement'$ betrachtet. Das gewichtete quadratische Mittel der Steigungsabweichungsdifferenzen $\triangle$SDx= nF_opt wird in beiden Fällen als aktueller Zielfunktionswert bestimmt. Des Weiteren werden die aktuelle Iterierte und der zugehörige Zielfunktionswert sowie die entsprechende Iterationszahl und Nummer der aktuellen Funktionsauswertung in einer Matrix gespeichert. Anschließend wird mit Hilfe der Matrix die beste Iterierte bestimmt, statt wie in der ursprünglichen Version von BOBYQA und CMA-ES die letzte Iterierte. Zum Schluss werden Plots der Slope Deviation in x und y sowie der lokalen Slope Deviation Differenzen von Optimierungs- und Referenzmodell erstellt. Auch ein Plot der Koordinatenabweichung in z, die z-Deviation, wird erzeugt.

5.2 Diskretisierungsuntersuchung

Die Idee bei der Diskretisierungsuntersuchung ist es die Optimierung zu beschleunigen, indem die Zeit für eine Funktionsauswertung verringert wird. In [Kle 11] wird eine Netzstudie durchgeführt, um die Unabhängigkeit der Simulationsergebnisse von der verwendeten Diskretisierung sicherzustellen. [Kle 11] zeigt, dass die Wahl von 200×200 Elementen für die Länge und Breite der Spiegelfacette ein genügend genaues Ergebnis liefert (siehe auch Abbildung A.2 im Anhang). Da der Fokus dieser Arbeit in der Untersuchung der Algorithmen und der damit verbundenen Exaktheit und Rechenzeit liegt, wird hier erneut eine Diskretisierungsanalyse in Bezug auf die Optimierungsalgorithmen durchgeführt. Zumal bei der Optimierung für jede Iteration eine Funktionsauswertung erfolgt, die teuer ist, wird versucht diese Zeit für die Auswerteprozedur zu verringern indem ein gröberes Netz gewählt wird. Es ist zu beachten, dass eine gröbere Diskretisierung der Spiegelfacette zu einer schnelleren Auswertung der Zielfunktion führt, während durch eine feinere Netzwahl ein genaueres Ergebnis erreicht wird. Erste Untersuchungen zeigen, wie fein die Diskretisierungen gewählt werden muss, um sowohl noch eine gute Aussagekraft über das Problem zu erhalten, als auch eine schnelle Konvergenz des Verfahrens zu gewährleisten. Dazu werden verschiedene Diskretisierungen des Spiegels gewählt. Es finden Diskretisierungen der Spiegelfacette von 10×10, 25×25, 50×50, 75×75, 100×100 sowie 200×200 Elementen statt, indem das Spiegelmodell in Ansys-Workbench betrachtet wird. Hier wird die jeweilige Anzahl an Elementen gewählt und in der Eingabedatei gespeichert sowie eine Zuweisung der zu optimierenden Parametern in dieser Datei getätigt. Dadurch entstehen die FE-Spiegelmodelle mit verschiedenen Diskretisierungen. Die beiden Algorithmen werden auf die Problemstellung mit unterschiedlicher Netzwahl angewendet und die Ergebnisse in Abschnitt 7.1 verglichen. Für das Modell mit Laborunterstruktur sowie für die realen Messergebnisse wird die von [Kle 11] empfohlene Diskretisierung gewählt.

5.3 Sensitivitätsanalyse

In der Sensitivitätsanalyse wird die Auswirkung eines variierenden Parameters, bei Konstanthaltung aller anderen, auf die optimale Lösung der Optimierungsaufgabe untersucht. Von Interesse ist der Einfluss auf das System durch die Veränderung des Parameters (siehe [BZ 12]).

In dieser Arbeit werden die Eingabewerte der beiden Algorithmen CMA-ES und BOBYQA variiert. Zunächst findet eine Untersuchung der Startschrittweite statt. Im Fall von BOBYQA ist die Startschrittweite durch ρ_{beg} und beim CMA-ES Algorithmus durch σ gegeben. Die Ergebnisse für die Wahl von verschiedenen Schrittweiten für unterschiedliche konstruierte Probleme finden sich in den Abschnitten 7.1 sowie 7.2. Des Weiteren ist es im Fall des BOBYQA Algorithmus möglich den finalen Trust-Region Radius ρ_{end} zu wählen.Der Einfluss dieses Parameters auf das Ergebnis wird untersucht und die Resultate in Abschnitt 7.2 und 7.5 vorgestellt. Da das CMA-ES Verfahren stochastisch arbeitet, können für den gleichen Versuch mit identischen

Startbedingungen unterschiedlich gute Ergebnisse berechnet werden. Aufgrund dessen werden für den CMA-ES Algorithmus jeweils zwei Durchläufe für jedes Modell gemacht. Ferner besteht die Option den Einfluss des Startwertes auf die Lösung zu untersuchen. Die Ergebnisse dieser Analyse finden sich in 7.3.

5.4 Komma- und Plus Strategie

Für das CMA-ES Verfahren werden unterschiedliche Vererbungsarten getestet. Der CMA-ES Algorithmus arbeitet durch die Vererbung der Eigenschaften der Eltern an die nächste Generation. Beim bisher verwendeten CMA-ES Verfahren wird nur die letzte berechnete Population als Eltern für die nächste Generation betrachtet. Diese Variante von Evolutionsstrategien ist unter der Komma-Strategie bekannt und wird mit CMA($\lambda\,,\mu$) bezeichnet. Werden nun statt der letzten Generation, alle bisher berechneten Lösungen für die Berechnung neuer Individuen betrachtet, so ist dies eine Plus-Strategie und wird als CMA($\lambda + \mu$) ausgedrückt. Der Vorteil liegt in einer stärkeren Fokussierung des Algorithmus. Dieser Fall wird durch eine Switch-Abfrage in den Code eingebaut. Wird der Plus-Fall betrachtet, dann werden die zwischengespeicherten Punkte sowie zugehörigen Zielfunktionswerte der Größe nach geordnet, wobei mit dem kleinsten Zielfunktionswert begonnen wird. Die μ besten Zielfunktionswerte aller Generationen werden Eltern der neuen Generation.

5.5 Hybridalgorithmus

Da durch die externen Verschiebungen an allen vier Aufhängungen eine gegenseitige Beeinflussung der Pads stattfindet, wird damit gerechnet, dass die Zielfunktion in ein lokales Optimum läuft. Aufgrund dessen wird ein Hybridalgorithmus aus dem CMA-ES und dem BOBYQA Algorithmus erzeugt und auf ein Problem mit 12 Variablen angewendet. Es werden die Translationen in x, y und z an allen vier Aufhängepunkten, also x_i, y_i und z_i für $i = 1, 2, 3, 4$, betrachtet. Der CMA-Algorithmus generiert zunächst zufällige Startwerte, welche vom BOBYQA Algorithmus ausgewertet werden. Dem CMA-Algorithmus werden 3 Iterationen mit jeweils nur 4 Funktionsauswertungen erlaubt. Während die maximale Anzahl an Funktionsauswertungen bei BOBYQA pro Durchlauf 1000 beträgt. Bei der Optimierung mit dem CMA-ES Algorithmus wird statt der MATLAB-Funktion `Funktionsauswertung.m`, das BOBYQA Verfahren mit der von CMA-ES berechneten Iterierten als Startwert für BOBYQA aufzurufen. Das berechnete Optimum wird nun wieder CMA-ES als Zielfunktionswert zur Iterierten, welche für BOBYQA als Startwert fungierte, übergeben. Durch die Optimierung sollen eventuelle lokale Optima aufgezeigt werden. Gleichzeitig wird dadurch eine Sensitivitätsanalyse bzgl. des Startwertes durchgeführt. Es werden unterschiedlich exakte Lösungen durch die verschiedenen zufälligen Startwerte generiert. Dadurch kann mit Hilfe der Lösungen eine Empfehlung für die Wahl des Startwertes gemacht werden.

Während der oben beschriebene Ansatz eine Verschachtelung der beiden Algorithmen darstellt, ist auch eine Hintereinanderausführung von CMA-ES und BOBYQA möglich. Zunächst generiert der CMA-ES Algorithmus vielversprechende Lösungen, das heißt mit kleinem Zielfunktionswert. Die besten Lösungen nutzt anschließend BOBYQA als Startwert und optimiert diese weiter.

5.6 Variation der Zielfunktion

Die geeignete Wahl der Zielfunktion hat einen großen Einfluss auf die Optimierung und kann diese beschleunigen und zu einer Verbesserung der Genauigkeit des Ergebnisses führen. Aufgrund dessen werden hier Variationen dieser getestet. Beispielsweise besteht die Möglichkeit die z-Deviation mit in die Optimierung aufzunehmen. Zum anderen kann auch die Zielfunktion (5.3) modifiziert werden.

5.6.1 Optimierung der z-Deviation

Da die genaue Position der Spiegelfacette im Raum unklar ist, können die gleichen Slope Deviation Plots bei unterschiedlichen Verschiebungsvektoren entstehen. Damit nicht unendlich viele gleiche Lösungen aufkommen, kam die Idee auf die z-Deviation mit in den Optimierungsprozess zu integrieren. Die z-Deviation stellt die Abweichung der z-Werte von der idealen Spiegelposition dar. Ein Voroptimierer, welcher die z-Komponenten des Lösungsvektor vorab in die richtige Position rückt, ist vorstellbar. Hier fungiert die Differenz der lokalen Werte der z-Deviation als Zielfunktion. Folglich wäre die Lage des Spiegels im Raum bekannt und damit der zuvor freie Parameter durch die Optimierung der z-Deviation bestimmbar. Zu beachten ist, dass die z-Deviation bei realen Deflektometriemessungen fehlerbehaftet ist. Kleine Messfehler oder Messungenauigkeiten werden durch die Integration aufsummiert. Denkbar ist auch die z-Deviation mit einer Wichtung versehen in die Zielfunktion zu involvieren. Durch die zusätzlichen Informationen für die z-Komponenten könnte der Optimierungsprozess beschleunigt werden.

Zunächst wird die Optimierung mit der z-Deviation für einen Testlauf untersucht. Damit gilt: $f : \mathbb{R}^4 \to \mathbb{R}$, $(z_1, z_2, z_3, z_4)^T \mapsto f(z_1, z_2, z_3, z_4)$. Es werden nur die Translationen in z der vier Montagepads als Optimierungsvariablen gewählt. Die Zielfunktion ist durch den RMS-Wert der Differenzen der lokalen z-Deviation von zwei Spiegelfacetten wie folgt gegeben:

$$f(z) = \triangle Devz = \sqrt{\sum_{k=1}^{p} ((devz_{2_k} - devz_{1_k})^2)}. \tag{5.5}$$

Analog zur Berechnung des RMS der Steigungsabweichungsdifferenzen in x, kann auch der Mittelwert der Differenzen der z-Deviation in MATLAB bestimmt werden. Mittels Switch-Anweisung wird zwischen Optimierung der z-Deviation und Slope Deviation in x unterschieden. Da nur die Optimierung der z-Werte erfolgt, werden nur diese in die Excel-Tabelle eingefügt, alle anderen

Rotations- und Translationsparameter werden innerhalb der Ansys-Eingabedatei auf null gesetzt. Die Ergebnisse der Optimierung finden sich in Abschnitt 7.3.

5.6.2 Zielfunktion ohne Wichtungsfaktor

Bei der Betrachtung der Zielfunktion, also des RMS-Wertes der lokalen Steigungsabweichungsdifferenzen, in Formel (5.3):

$$\triangle \mathrm{SDx} := \sqrt{\sum_{k=1}^{p} \left((sdx_{2_k} - sdx_{1_k})^2 \frac{a_k}{A_{tot}} \right)},$$

fällt auf, dass dieser mit einem Wichtungsterm $\frac{a_k}{A_{tot}}$ versehen ist. Es besteht die Möglichkeit, dass diese Wichtung der lokalen Slope Deviation-Werte in x einen Einfluss auf den Optimierungsprozess hat. Es ist denkbar, dass dem Algorithmus durch die stärkere Wichtung bestimmter Bereiche der Optimierungsprozess erschwert wird, da beispielsweise eine Verzerrung oder Streckung des Suchraums stattfinden kann. Deshalb findet in Abschnitt 7.5 eine Untersuchung eines Testproblems mit und ohne Wichtung der Zielfunktion statt.

5.7 Parameterreduktion

Es wird gezeigt, dass verschiedene externe Verschiebungen der Montagepads zu den gleichen Slope Deviation Plots sowohl in x als auch in y und damit zu den gleichen Lösungen führen können. Damit kann ein Problem unendlich viele äquivalente Lösungen besitzen. Folglich ist die Funktion f nicht injektiv. Eine Möglichkeit damit umzugehen ist es, ein Pad während des kompletten Optimierungslaufes zu fixieren und die anderen für die Optimierung freizugeben. Durch das Festhalten eines Pads ist die Lage des Spiegels im Raum fest definiert und es wird das Auftauchen von mehreren Lösungen vermieden. Um dennoch Änderungen der lokalen Steigungsabweichungen an der Stelle des in x, y und z festgehaltenen Montagepads zu beschreiben, bleiben die Rotationen an diesem Pad variabel. Aufgrund dessen verringert sich die Optimierung um 3 Parameter. Durch die geringere Anzahl an Variablen wird eine schnellere Optimierung und damit eine kürzere Laufzeit erhofft. Das fixierte Pad wird für die Translationen in x, y und z in der Ansys-Eingabedatei auf null gesetzt. Die Ergebnisse werden ab Abschnitt 7.3 vorgestellt.

5.8 Skalierung

Der Ansys-Solver verwendet intern für die Rotationsparameter um die x-, y- und z-Achse immer die Einheit rad. Selbst wenn die auszuführende Datei in Ansys-Workbench erzeugt und die Rotationen in Grad gegeben sind, findet intern eine Umrechnung in das Bogenmaß rad

statt. Dabei entspricht 1 rad $= 57.3°$. Damit haben die Optimierungsparameter unterschiedliche Größenordnungen. Während die Translationen im Bereich von $\pm$ 5 mm liegen, erstrecken sich die Rotationen im Intervall von $\pm$ 0.005 rad. Solche Probleme werden als schlecht konditionierte Probleme bezeichnet. Sie sind schwer zu lösen, da zum einen die Konditionszahl des Problems groß ist. Dies führt dazu, dass mehr Iterationen bis zur Konvergenz des Verfahrens benötigt werden. Bei der teuren Funktionsauswertung des hier vorliegenden Problems ist es wünschenswert zusätzliche Funktionsauswertungen zu vermeiden. Außerdem ist es schwer ein angemessenes Abbruchkriterium für die unterschiedlichen Optimierungsparameter zu finden. Ein für die Translationen geeignetes Abbruchkriterium ist für die Rotationen zu groß gewählt. Für die Rotationen wird eine höhere Genauigkeit gefordert, um ein genügend genaues Optimum zu erreichen. Durch die richtige Skalierung der Parameter kann das beschriebene Problem leicht gelöst werden. Dazu wird die Verformung der Spiegelfacette statt in rad in der Einheit mrad betrachtet. Da maximale Rotationen von $\pm$ 0.005 rad $= \pm$ 5 mrad angemessen sind, stimmt die Wahl der Einheit mrad mit der Translationsgrößenordnung überein. Dazu wird die MATLAB-Funktion `Funktionsauswertung.m` in Algorithmus 5.1 durch eine switch-Abfrage erweitert. Die Parameter, die ANSYS übergeben werden, werden an die Posititon *handles.ConfigParams.ANSYS.Parameter.value* gespeichert. *handles.ConfigParams.ANSYS.Parameter.unit* listet die für die Optimierungsvariable verwendete Einheit. Wird die Einheit des Parameters in mrad betrachtet, so wird der zu optimierende Parameter nochmal durch 1000 geteilt. Dadurch wird die Umrechnung von rad zu mrad erreicht. Ist die Einheit mm gegeben, folgt nur die Übergabe der Parameter ohne Skalierungsfaktor. Durch die Skalierung der Parameter wird eine schnellere Optimierung erreicht. Es finden eine Gegenüberstellung der Optimierung in rad und mrad in Abschnitt 7.5 statt.

5.9 Coordinate-Descent

Das Coordinate-Descent Verfahren zeichnet sich dadurch aus, dass die meisten Parameter festgehalten werden, während nur einige wenige optimiert werden, wie in Abschnitt 3.2.4 beschrieben. Anschließend werden die zuvor optimierten Parameter als fest definiert, wobei andere für die Optimierung freigegeben werden. In dieser Arbeit werden die Variablen der einzelnen Pads als Teilparameterset für die Coordinate-Descent Optimierung genutzt. Damit springt das Coordinate-Descent Verfahren zwischen den Pads $(x_i, y_i, z_i, \alpha_i, \beta_i)^T$ für $i = 1, 2, 3, 4$ und optimiert jeden Aufhängepunkt einzeln. Für den Optimierungstest des Coordinate-Descent Verfahrens wird die Rotation um die z-Achse zunächst nicht betrachtet. Die Hinzunahme dieser Parameter ist jedoch einfach umsetzbar. Es kann für den Optimierungsprozess zwischen CMA-ES und BOBYQA gewählt werden.

Die Funktion 5.2 beschreibt in groben Zügen die Implementierung der Coordinate-Descent Methode. Die grundlegende Idee besteht in der Verwendung der Modulo-Funktion. Die drei Gravitationskomponenten und die Optimierungsparameter $x_1, y_1, z_1, \alpha_1, \beta_1, x_2, \cdots, \beta_4$ sind in dieser Reihenfolge in *handles.ConfigParams.ANSYS.Parameter* gespeichert. Ein Laufindex p eingeführt, der die Nummer des Durchlaufes der einzelnen Coordinate-Descent Teiloptimierungen

Algorithmus 5.2 : CoordinateDescent.m

```
...
for p = 1 : n do
  for j = 1 : 5 do
    vX(j) =
    str2num(handles.ConfigParams.ANSYS.Parameter(3 + j + mod(p − 1, 4) ∗ 5).value);
  end
  [nF_opt, vX_opt] = bobyqa(vX, a, b, ...)
  for j = 1 : 5 do
    handles.ConfigParams.ANSYS.Parameter(3 + j + mod(p − 1, 4) ∗ 5).value =
    mat2str(vX_opt(j));
  end
end
...
```

beschreibt und indirekt, durch die Modulo-Funktion, auch die Padnummern. Dadurch ist ein einfacher zyklischer Durchlauf für die Coordinate-Descent Methode möglich. Mit Hilfe der Modulo-Funktion werden die ersten fünf Variablen, also alle Parameter des ersten Pads, im Vektor vX gespeichert. $(p-1)mod4$ für $i = 1, 2, \cdots$ stellt sicher, dass das Verfahren zwischen den vier Montagepads springt. Anschließend wird der Rest, welcher aus der Berechnung der Modulo-Funktion folgt, mit dem Faktor 5 multipliziert, da hier an jedem Pad fünf Unbekannte existieren. Die drei Gravitationskomponenten, die an den ersten Stellen des Strukturparameters stehen werden durch das Aufaddieren des Wertes 3 zum restlichen Term weggelassen. Die Variablen werden dem BOBYQA Algorithmus übergeben, sodass dieser statt mit 20 Optimierungsvariablen nur mit 5 arbeitet. Die maximale Anzahl an Funktionsauswertungen sowie Wahl der Grenzen für die Teiloptimierungen sind dem Benutzer überlassen. Innerhalb der Funktion `bobyqa.m` findet der Aufruf der `Funktionsauswertung.m` statt. Hier folgt die umgekehrte Vorgehensweise. Die Optimierungsparameter werden wieder so gespeichert, dass sie an ANSYS übergeben werden. Die Teiloptimierung stoppt mit den für BOBYQA gewählten Abbruchbedingungen. Nachdem der Optimierer das berechnete Optimum zurückgegeben hat, wird dieses an die richtige Stelle gesetzt, der Laufindex p wird um eins erhöht und die Optimierung schreitet solange voran, bis die maximale Anzahl an Durchläufen für das Coordinate-Descent Verfahren erreicht ist.

Algorithmus 5.3 : Funktionsauswertung.m

```
(Coordinate-Descent)
Funktion: [ nF_opt, handles ] = Funktionsauswertung( vX_opt, handles )
  for i = 1 : length(vX_opt) do
    if strcmp(char(handles.ConfigParams.ANSYS.Parameter(3 + i).unit),'mrad') then
      handles.ConfigParams.ANSYS.Parameter(3 + i + mod(p − 1, 4) ∗ 5).value =
      mat2str(vX_opt(i)/1000);
    else
      handles.ConfigParams.ANSYS.Parameter(3 + i + mod(p − 1, 4) ∗ 5).value =
      mat2str(vX_opt(i));
    end
  end
  ...
```

5.10 Vorgehensweise bei der Laborunterstruktur

Die Nachbildung des FE-Spiegelmodells mit Laborunterstruktur aus Kapitel 4.2.2.2 mit Hilfe der Optimierer hat zunächst einige Probleme bereitet. Dies wurde durch die Slope Deviation Plots in x verdeutlicht. Die Plots von Optimierungsmodell und Referenz zeigen eine große Diskrepanz. Es wurden eine Vielzahl von Optimierungsdurchläufen absolviert um den Grund für das schlechte Optimierungsergebnis zu finden. Dazu wurde der Startwert variiert mit der Hoffnung einen besseren Ausgangspunkt für die Optimierer zu finden. Ferner wurde die z-Deviation in die Zielfunktion eingefügt. Da vermutet wurde, dass mittels externer Verschiebungen an den vier Pads das Referenzmodell der Laborunterstruktur nicht nachgebildet werden kann, wurden die Klammern, welche den Spiegel mit der Unterkonstruktion verbinden, mit in das FE-Optimierungsmodell aufgenommen. Durch die Veränderung des Modells wurde ein ähnlicheres Verformungsverhalten erwartet. Des Weiteren wurde über eine Wichtung der Randbereiche des Spiegels nachgedacht, da diese besonders schlecht von den Optimierern nachgeahmt wurden.

Es stellte sich heraus, dass beim Vergleich der lokalen Slope Deviation-Werte von Optimierungs- und Referenzmodell ein Fehler auftrat. Da bei der Modellierung der Spiegelfacette mit Laborunterkonstruktion, in diesem Fall das Referenzmodell, kein strukturiertes Netz für den Spiegel verwendet wurde, kam es bei der Gegenüberstellung beider Modelle zum Vergleich von unterschiedlich gelegenen Knotenpunkten. Dies führte zu einem verzerrten Bild der Zielfunktion und machte es den Optimierern unmöglich in die Nähe des Optimums zu kommen. Obwohl stochastische Verfahren im Allgemeinen besser mit einem zerklüfteten Suchraum umgehen können als modellbasierte Verfahren, war hier die Tatsache, dass BOBYQA bessere Ergebnisse als CMA-ES erzielte, auffällig. Nach der Behebung des Fehlers wurden erfolgversprechende Ergebnisse berechnet, welche in Abschnitt 7.5 vorgestellt werden.

5.11 Vorgehensweise bei der realen Spiegelmessung

Zum Schluss wird der Algorithmus BOBYQA auf ein reales Messergebnis einer Deflektometriemessung angewendet. Details zum verwendeten Spiegel und der Vermessung des Innenspiegels finden sich in Abschnitt 6.2.

Da eine reale Spiegelfacette nicht ideal hergestellt werden kann, sondern Spiegelfehler aufgrund von Fertigungstoleranzen entstehen, müssen diese im Optimierungsmodell berücksichtigt werden. Die hier betrachtete Spiegelfacette wurde von Meiser [Mei 13] in einer vertikalen Position vermessen. Dadurch kann der Gravitationseinfluss auf den Spiegel vernachlässigt werden und der Plot der Slope Deviation zeigt nur die herstellungsbedingten Fehler der reflektierenden Oberfläche. Auf die lokalen Slope Deviations des Optimierungsmodells werden für jede Iteration noch die herstellungsbedingten Fehler aufaddiert. Erst anschließend folgt ein Vergleich mit der Referenz über den RMS-Wert der Steigungsabweichungsdifferenzen. Die Addition

der lokalen Slope Deviations von Optimierungsmodell und herstellungsbedingten Fehlern ist bereits in der MATLAB-Routine enthalten. Wie bereits in Abschnitt 5.1.5 beschrieben, wird zwischen den verschiedenen Referenztypen unterschieden. Wird in der Konfigurationsdatei der Fall $'AddMeasData2FEMResultsAndCompareWithMeasurement'$ gewählt, dann handelt es sich bei der Referenz um eine reale Deflektometriemessung. Hier werden auch die Slope Deviation Plots der herstellungsbedingten Fehler bei der Berechnung des RMS-Wertes der lokalen Steigungsabweichungsdifferenzen mit berücksichtigt. Die Knoten aus dem Optimierungsmodell werden automatisch auf die Ergebnisse der Deflektometriemessung übertragen, sodass ein Vergleich der beiden Modelle möglich ist. Die Ergebnisse des Optimierungslaufes finden sich in Abschnitt 7.6.

5.11.1 Multilevelansatz

Da die Auswertung der realen Messung durch die feine Diskretisierung von 200×200 Elementen viel Zeit in Anspruch nimmt, wird der Optimierungsprozess beschleunigt, indem die Spiegelfacette zunächst auf einem gröberen Gitter betrachtet wird. Nach einer Anzahl an Iterationen wird anschließend das feine Gitter mit für die Optimierung verwendet. Es existiert bereits eine MATLAB-Routine, welche die Knotendaten von simulierten Spiegelmodellen mit den gemessenen Daten vergleicht und mit einer variablen Anzahl an Knoten umgehen kann. Hier wird die Wahl der Diskretisierung automatisch auf die Messdaten übertragen. Folglich ist es möglich unterschiedliche Diskretisierungen beim Optimierungsprozess zu wählen und mit der gemessenen Referenz zu vergleichen. Zu beachten ist, dass verschiedene Diskretisierungen eine unterschiedliche Güte und Genauigkeit der Lösung bringen. Für die Verwendung einer Multilevelvariante muss ein Zusammenhang zwischen den Zielfunktionswerten der unterschiedlichen Netzwahl gefunden werden. Wird der Tatsache, dass zwischen den unterschiedlichen Diskretisierungsmodellen ein Fehler herrscht, wie bereits in Abschnitt 3.2.5 dargestellt, keine Beachtung geschenkt, dann stoppt der Optimierungsprozess frühzeitig und findet keine akkurate Lösung. Da die Zielfunktionswerte für den gleichen Punkt jedoch unterschiedliche Diskretisierungsmodelle mit steigender Elementenanzahl größer wird, wird der Optimierer den Sprung von einer groben Netzwahl zu einer feineren Diskretisierung nicht meistern und damit kann es dem Optimierer nicht gelingen, die Lösung weiter zu verbessern. In [Kle 11] wird der prozentuale Fehler unterschiedlicher Netzqualitäten untersucht. Dieser ist im Anhang in Abbildung A.2 dargestellt. Die Netzstudie stellt die Unabhängigkeit des Simulationsergebnisses vom gewählten Netz sicher. Die Darstellung zeigt für eine Netzteilung von 100×100 Elementen eine maximale Abweichung zum Netz mit 200×200 Elementen von unter 7 %. Diese Information wird genutzt, um von einer groben Diskretisierung zu einer feineren Vernetzung zu springen.

Seien $sdx_{1_k}^{200}$ und $sdx_{2_k}^{200}$ die lokalen Slope Deviation-Werte der Referenz, sowie des Optimierungsmodells für eine Diskretisierung von 200×200 Elementen, sowie $sdx_{1_k}^{100}$ und $sdx_{2_k}^{100}$ für 100×100 Elemente. Damit ergibt sich folgende Ungleichung:

$$|sdx_{2_k}^{200} - sdx_{1_k}^{200}| \leq 1.07|sdx_{2_k}^{100} - sdx_{1_k}^{100}|. \tag{5.6}$$

Der Zusammenhang zwischen den Zielfunktionswerten der beiden Diskretisierungen ist wie folgt gegeben:

$$\triangle \mathrm{SDx}_{200} = \sqrt{\sum_{k=1}^{p} \left((sdx_{2_k}^{200} - sdx_{1_k}^{200})^2 \frac{a_k}{A_{tot}} \right)} \leq \sqrt{\sum_{k=1}^{p} \left(1.07^2 (sdx_{2_k}^{100} - sdx_{1_k}^{100})^2 \frac{a_k}{A_{tot}} \right)} \tag{5.7}$$

$$= 1.07 \sqrt{\sum_{k=1}^{p} \left((sdx_{2_k}^{100} - sdx_{1_k}^{100})^2 \frac{a_k}{A_{tot}} \right)} = 1.07 \triangle \mathrm{SDx}_{100}. \tag{5.8}$$

Durch die Addition des Faktors 1.07 auf den Zielfunktionswert des groben Modells kann später auf die feinere Diskretisierung gewechselt werden, sodass sichergestellt ist, dass der Zielfunktionswert weiter sinkt. Bei einer realen Messung werden darüber hinaus noch herstellungsbedingte Fehler auf das Optimierungsmodell addiert. Hier kann es durch den Wechsel des Netzes bzw. durch kleine Abweichungen zwischen Knotenkoordinaten und Messwerten zu Fehlern kommen (vgl. [Kle 11]). Aufgrund dessen wird die Zielfunktion des gröberen Modells mit einem Korrekturfaktor von 1.10 multipliziert. Für die Optimierung werden die ersten 350 Evaluationen mit einer Netzteilung von 100×100 Elementen durchgeführt und nachfolgend wird mit einer Diskretisierung von 200×200 Elementen gearbeitet. Die Resultate sind in Abschnitt 7.6.1 präsentiert.

6 Testprobleme und Messergebnisse

Es werden verschiedene Testprobleme konstruiert und als Referenzlösung genutzt, um die Algorithmen CMA-ES und BOBYQA aus Kapitel 3, Abschnitt 3.2.3 und 3.2.2 auf ihre Performance hin zu untersuchen. Des Weiteren werden im QUARZ® Labor des DLR deflektometrische Messungen durchgeführt. Das Ergebnis einer realen Spiegelmessung eines RP3-Innenspiegels dient als letzte Optimierungsaufgabe für den erfolgversprechendsten Algorithmus und zeigt, ob es auch möglich ist reale Spiegelmessungen nachzubilden.

6.1 Reduzierte Testprobleme mit variabler Parameterzahl

Es werden zunächst niedrigdimensionale Probleme erstellt und schrittweise Dimensionen hinzugefügt, sodass immer komplexere Testprobleme entstehen. Zuerst werden nur Translationen an einem Montagepad bzw. zwei Aufhängepunkten erzeugt. Ferner werden die Translationen an allen vier Pads betrachtet und schließlich auch Rotationsvariablen in die Optimierung eingebaut. Zum Schluss wird das FE-Spiegelmodell mit Laborunterstruktur als Testproblem aufbereitet.

6.1.1 Dreidimensionales Testproblem

Zunächst soll nur ein Aufhängepunkt, in diesem Fall das Pad 1, in Richtung x, y und z verschoben werden. Die Aufhängepunkte 2, 3 und 4 sind fixiert. Damit handelt es sich um ein dreidimensionales Testproblem. Es werden zwei verschiedene Testprobleme für den dreidimensionalen Fall konstruiert.

6.1.1.1 Testproblem 1

Das erste Testproblem besteht zunächst aus dem Padmodell. In diesem Fall werden die Spiegelfacette, die vier Aufhängepads und die Klebstoffschicht, welche Spiegel und Pads miteinander verbinden, simuliert. Der Spiegel ist in 0°-Zenitstellung ausgerichtet, das heißt für die Gravitationskomponenten gilt $g = [0,\ 0,\ 9810]^T$. Beim ersten erstellten Testproblem liegt das konstruierten Optimum des Referenzmodells bei $[x_1, y_1, z_1]^T = [0.5,\ 1.5,\ 1.0]^T$, welches mit Hilfe der Optimierer gefunden werden soll. Das heißt, das Pad wurde um 0.5 mm nach rechts entlang der x-Achse verschoben, 1.5 mm entlang der y-Achse und 1.0 mm nach oben verrückt und der Spiegel entsprechend

verformt. Der flächengewichtete Mittelwert der lokalen Slope Deviation ist für eine Diskretisierung der Spiegelfacette bei einem Netzteilung von 50×50 Elemente durch $\mathrm{SDx} = 1.386\,\mathrm{mrad}$ gegeben. In Abschnitt 7.1.1.1 folgt eine Gegenüberstellung der Slope Deviation Plots für die verschiedenen Diskretisierungen. Abbildung 6.1 zeigt den Plot der Slope Deviation in x auf der linken Seite

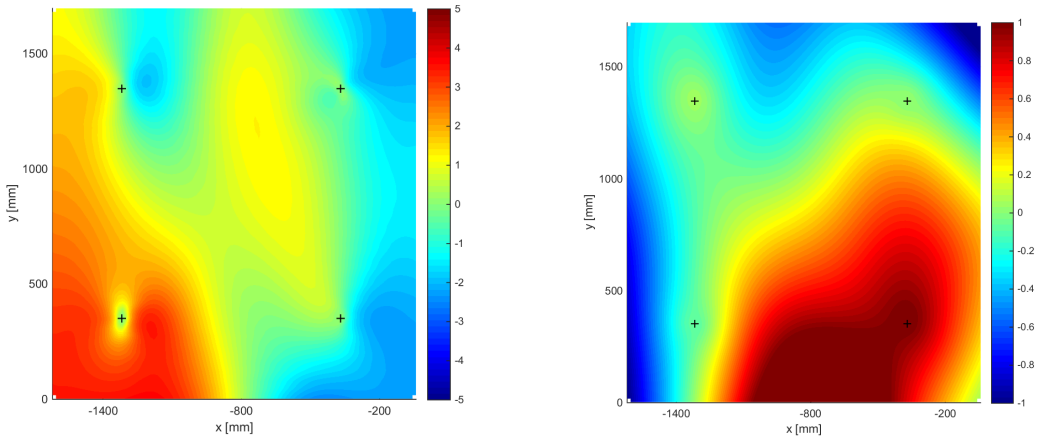


Abbildung 6.1: Slope Deviation Plot (links) und z-Deviation Plot (rechts) des Padmodells von Testproblem 1

sowie rechts den der z-Deviation, also die Höhenlinien der Spiegelfacette. Anhand des z-Deviation Plots ist gut erkennen, dass Pad 1 unten links nach oben verschoben wurde. Mit Hilfe dieses Testproblems erfolgt eine Diskretisierungsanalyse bezüglich der Spiegelfacette, wie in Abschnitt 5.2 beschrieben. Des Weiteren findet eine erste Sensitivitätsanalyse bezüglich der Startschrittweite statt. Außerdem werden die gleichen Verschiebungen am ersten Aufhängepunkt für das Klebstoff-Spiegelmodell betrachtet. Die zugehörigen Slope Deviation Plots sind in Abbildung 6.2 dargestellt. Hier liegt der RMS-Wert der lokalen Slope Deviation bei $\mathrm{SDx} = 2.006\,\mathrm{mrad}$. Die Ergebnisse werden in Abschnitt 7.1 präsentiert.

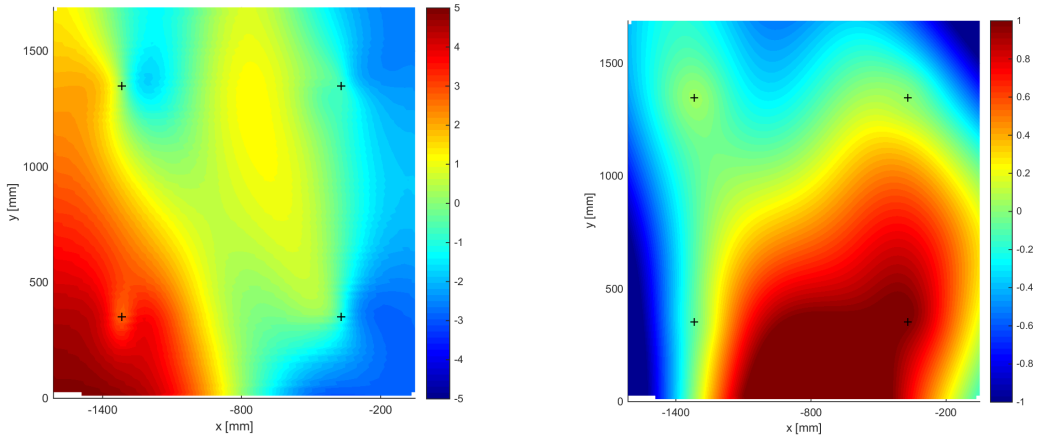


Abbildung 6.2: Slope Deviation Plot (links) und z-Deviation Plot (rechts) des Klebstoffmodells von Testproblem 1

6.1.1.2 Testproblem 2

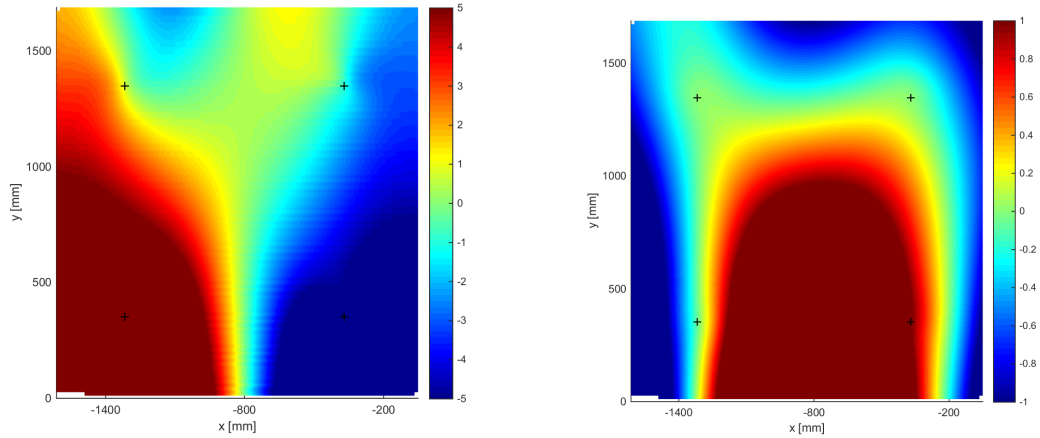


Abbildung 6.3: Slope Deviation Plot (links) und z-Deviation Plot (rechts) von Testproblem 2

Testproblem 2 besteht aus der Spiegelfacette und der Klebstoffschicht, ohne Berücksichtigung der Pads. Ebenso wie der Spiegel in Testproblem 1 ist auch dieser Spiegel in 0°-Zenitstellung ausgerichtet. Der Lösungsvektor des Optimierungsproblems ist durch $[x_1, y_1, z_1]^T = [1,\ 1,\ 1]^T$ gegeben. Abbildung 6.3 zeigt links den Slope Deviation Plot in x und auf der rechten Seite den z-Deviation Plot für das konstruierte Problem, welches durch die beiden Optimierungsalgorithmen gelöst wird. Für den RMS-Wert der Slope Deviation gilt SDx = 4.601 mrad. Die Entscheidung fällt auf eine Netzteilung von 50×50 Elementen. Ebenfalls finden sich die Ergebnisse in Abschnitt 7.1.

6.1.2 Sechsdimensionales Testproblem

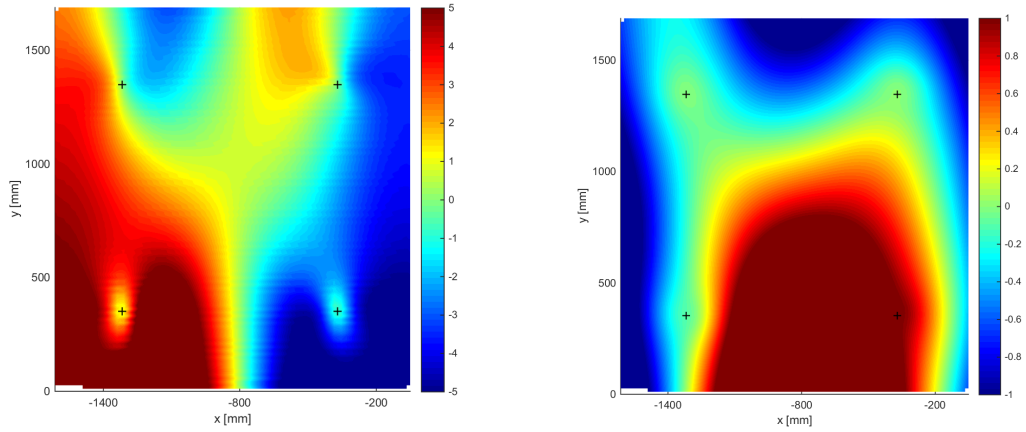


Abbildung 6.4: Slope Deviation Plot in x (links) und z-Deviation Plot (rechts) des sechsdimensionalen Testproblems

Für die sechsdimensionale Optimierung wird nur mit der Klebstoffschicht, ohne die Pads, simuliert, um eine schnelle Optimierung zu erreichen. Hier wird die Gravitation testweise auf

$g = [0,\ 0,\ 18000]^T$ gesetzt. Die Wahl der Gravitation hat keinen Einfluss auf den Optimierungsprozess, da sowohl das Optimierungsmodell als auch die Referenz den gleichen Gravitationskomponenten unterliegen. Genau wie bei Testproblem 2 im Dreidimensionalen, wird Pad 1 um den Vektor $[1,\ 1,\ 1]^T$ verschoben und Pad 3 bleibt unverformt, soll jedoch anderes als im vorhergegangenen Fall mit in die Optimierung eingehen. Das Optimum befindet sich somit bei $[x_1, y_1, z_1, x_3, y_3, z_3]^T = [1,\ 1,\ 1,\ 0,\ 0,\ 0]^T$. Abbildung 6.4 zeigt die Slope Deviation in x und die z-Deviation dieses Testproblems. Es gilt SDx = 3.432 mrad. Obwohl der gleiche Verschiebungsvektor wie im dreidimensionalen Fall gewählt wurde, unterscheiden sich die beiden Abbildungen 6.3 und 6.4. Dies liegt an der unterschiedlichen Wahl der Gravitation g. Die Optimierer sollen in diesem Fall auf ihre Genauigkeit und Laufzeit untersucht, verifiziert und optimiert werden. Dies geschieht durch die Variation verschiedener Eingabewerte für die Optimierung wie beispielsweise der Startschrittweite σ oder der Populationsgröße λ. Die Ergebnisse sind in 7.2 dargestellt.

6.1.3 12-dimensionales Testproblem

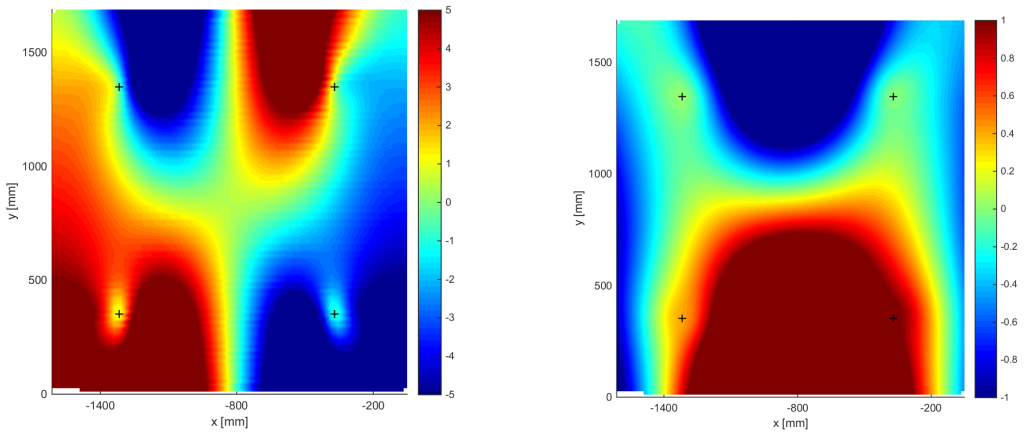


Abbildung 6.5: Slope Deviation Plot (links) und z-Deviation Plot (rechts) des 12-dimensionalen Testproblems

Ein Testproblem mit dem Optimum $[x_1, y_1, z_1, x_2, ..., z_4]^T = [1,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ \frac{1}{2},\ \frac{1}{2},\ 0,\ 0]^T$ wird konstruiert. Auch hier gilt für die Gravitation $g = [0,\ 0,\ 18000]^T$. Es wird erneut das Klebstoffmodell für einen schnelleren Optimierungslauf gewählt. Slope Deviation Plot in x und z-Deviation Plot sind in Abbildung 6.5 abgebildet. Der RMS-Wert der lokalen Slope Deviation beträgt SDx = 4.180 mrad. Es ist damit zu rechnen, dass durch die Verschiebungen der vier Aufhängepunkte eine gegenseitige Beeinflussung der Pads stattfindet. Deshalb erfolgt eine Anwendung des Hybridalgorithmus aus 5.5. Des Weiteren wird dadurch indirekt eine Sensitivitätsanalyse bezüglich des Startwertes abgeleitet. Es wird die Optimierung der z-Deviation, wie in Abschnitt 5.6.1 beschrieben, und eine Parameterreduktionsversuch (siehe 5.7) durchgeführt. Die Resultate der Optimierungsdurchläufe werden in Abschnitt 7.3 vorgestellt.

6.1.4 20-dimensionales Testproblem

Es wird das Klebstoffmodell mit Gravitationsvektor $g = [0,\ 0,\ 9810]^T$ betrachtet. Im Punkt $[x_1, y_1, z_1, \alpha_1, \beta_1, x_2, ..., \beta_4]^T = [0.5, 0, 0, 0, 0,\ 0, 0, 0, 0, 0,\ -0.2, 0, 0, 0, 0,\ -0.2, 0, 0, 0, 0]^T$ befindet sich das konstruierte Optimum. Dadurch gilt SDx $= 2.878$ mrad. Abbildung 6.6 zeigt auf der linken Seite den Slope Deviation Plot in x und rechts den z-Deviation Plot. Hier wird getestet, ob der Optimierer auch durch das Hinzufügen von Rotationen das Optimum findet. Des Weiteren findet die Coordinate-Descent Methode aus Abschnitt 5.9 ihre Anwendung. Die Ergebnisse werden in 7.4 präsentiert.

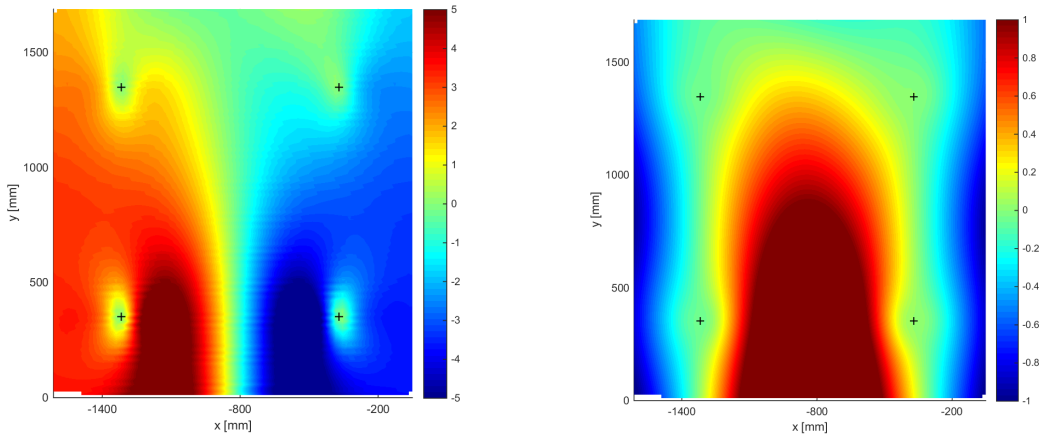


Abbildung 6.6: Slope Deviation Plot (links) und z-Deviation Plot (rechts) des 20-dimensionalen Testproblems

6.1.5 Laborunterstruktur

Das realistische Modell der Spiegelfacette aus Abschnitt 4.2.2.2, welche auf der Laborunterstruktur befestigt ist dient als Testproblem für die Optimierer und verifiziert, dass die Optimierer CMA-ES und BOBYQA auch in der Lage sind für diese Aufgabenstellung wirklichkeitsnahe Probleme zu lösen. Da die Spiegelfacette horizontal auf der Unterstruktur liegt, gilt für die Gravitation $g = [2350.4,\ 0,\ 9524.3]^T$. Durch die Gravitation sowie die Unterstruktur wirken Kräfte auf die Spiegelaufhängepunkte, sodass dies kleine Verschiebungen und Verkippungen zur Folge hat. Die Positionen der verschobenen Pads können in Tabelle 6.1 abgelesen werden.

Tabelle 6.1: Translationen der Pads für die Laborunterstruktur

	Pad 1	Pad 2	Pad 3	Pad 4
x [mm]	$-4.314 \cdot 10^{-2}$	$-3.942 \cdot 10^{-2}$	$3.934 \cdot 10^{-2}$	$4.292 \cdot 10^{-2}$
y [mm]	$-1.230 \cdot 10^{-3}$	$-1.970 \cdot 10^{-3}$	$2.113 \cdot 10{-}3$	$1.312 \cdot 10^{-3}$
z [mm]	$-1.437 \cdot 10^{-3}$	$-1.711 \cdot 10^{-3}$	$-1.994 \cdot 10^{-3}$	$-1.592 \cdot 10^{-3}$

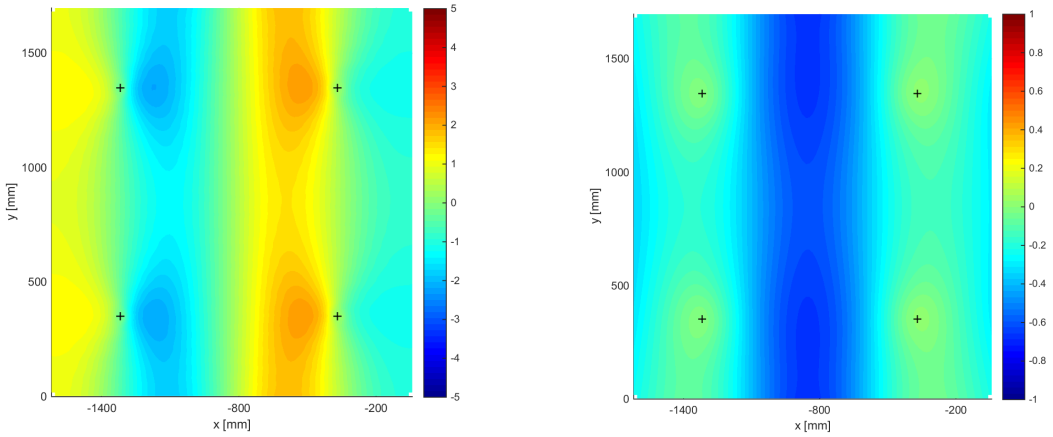


Abbildung 6.7: Slope Deviation Plot in x (links) und z-Deviation Plot (rechts) der Spiegelfacette mit Laborunterstruktur

Die Rotationen an den Padpositionen konnten nicht abgelesen werden, da diese zu klein sind. Die Plots der Slope Deviation in x und z-Deviation sind in Abbildung 6.7 dargestellt und es gilt SDx $= 1.137\,\text{mrad}$. Die Optimierungsresultate zu den Translations-, Skalierungs- und Parameterreduktionsuntersuchungen sind in Abschnitt 7.5 aufgezeigt.

6.2 Reale Spiegelmessung

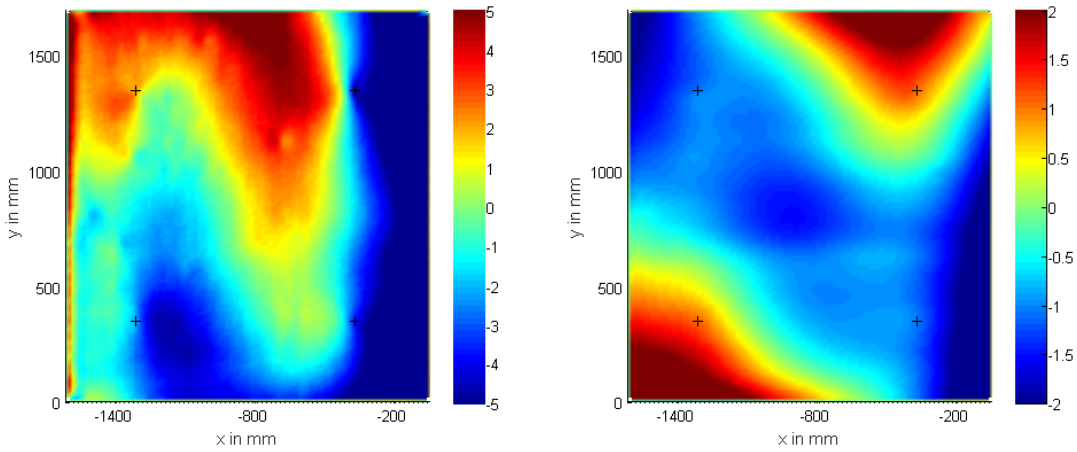


Abbildung 6.8: Slope Deviation Plot in x (links) und z-Deviation Plot (rechts) des RP3 Innenspiegels

Zuletzt folgt eine Anwendung des erfolgversprechendsten Algorithmus auf eine reale Messung. Dazu wurde eine deflektometrische Messung eines RP3 Innen- und Außenspiegels der Firma Flabeg im QUARZ® Labor in 0°-Zenitstellung durchgeführt. Die Spiegelfacetten wurden für eine unveröffentlichte interne Arbeit des DLR von Timo Effertz und Simon Schneider vermessen. Die zu vermessenden Innen- und Außenspiegel wurden auf eine Trägerkonstruktion angebracht.

Diese besteht aus der Torque-Box, welche eine Rahmenstruktur zur Stabilisierung und Torsionsverhinderung der Konstruktion darstellt. Ausgehend von der Torque-Box erstrecken sich zwei Cantilever-Arme, das heißt zwei Trägerarme, an denen die Spiegel angebracht wurden. Aus Symmetriegründen wurde nur eine Hälfte der Parabolrinne im Labor aufgebaut. Anschließend wurden die Spiegelfacetten deflektometrisch vermessen. Es ist zu beachten, dass sich die Klammern bei der Messung nicht in Idealposition befanden, da die Unterstruktur nicht ideal eingestellt wurde. Aufgrund dessen ist es zu relativ starken Verformungen der Spiegelfacetten gekommen. Abbildung 6.8 zeigt die Plots der Slope Deviation in x und der z-Deviation des Innenspiegels mit der Identifikationsnummer 110355717. Für die Slope Deviation gilt $\mathrm{SDx} = 3.564\,\mathrm{mrad}$. Die Ergebnisse des Optimierungslaufes mit dem BOBYQA Algorithmus sowie ein erster Test mit einem Multilevelansatz werden in Abschnitt 7.6 vorgestellt.

7 Ergebnisse

Die beiden Algorithmen BOBYQA und CMA-ES werden mit den beschriebenen Variationen und Verfahren aus Kapitel 5 auf die in Kapitel 6 vorgestellten Testprobleme angewendet und hinsichtlich der Exaktheit der Lösung, Anzahl an Evaluationen und der Laufzeit untersucht.

7.1 Dreidimensionales Problem

Für den Fall der dreidimensionalen Testprobleme wird der Klebstoff bzw. die Pads 2, 3 und 4 festgehalten und nur die Klebstoffschicht oder das Montagepad 1 optimiert. Damit handelt es sich um ein dreidimensionales Optimierungsproblem mit $f : \mathbb{R}^3 \to \mathbb{R}, (x_1, y_1, z_1)^T \mapsto f(x_1, y_1, z_1)$.

7.1.1 Diskretisierungsanalyse und Sensitivitätsanalyse bzgl. der Startschrittweite

Die Algorithmen werden auf das Testproblem 1 aus Abschnitt 6.1.1.1 mit dem Startwert $x_0 = [0,\ 0,\ 0]^T$ angewendet. Die Lösung des Problems ist durch $[0.5,\ 1.5,\ 1.0]^T$ gegeben. Für die Grenzen a und b gilt $a_i = -5$ sowie $b_i = 5$ für $i = 1, 2, 3$. Der Algorithmus CMA-ES stoppt bei Erreichen der Genauigkeit $\epsilon_{end} = 10^{-4}$. Für BOBYQA ist der finale Trust-Region mit $\rho_{end} = 10^{-5}$ als Abbruchkriterium gegeben. Der Spiegel wird in 0 °-Zenitstellung getestet. Es werden für jeden Optimierungstest 250 Funktionsauswertungen erlaubt. Diese werden hinsichtlich verschiedener Diskretisierungen der Spiegelfacette, wie in Abschnitt 5.2 beschrieben, untersucht.

Für die erste Untersuchung wird ein Spiegelmodell mit einer Klebstoffschicht und den vier Pads betrachtet. Da der BOBYQA Algorithmus indirekt Ableitungen benutzt, ist in diesem Fall eine Diskretisierungsanalyse sinnvoll. Für das CMA-ES Verfahren hingegen ist eine solche Analyse eher ungeeignet. Da es sich um ein ableitungsfreies Verfahren handelt, wird nicht davon ausgegangen, dass eine bestimmte Diskretisierung dem Verfahren ein Vorteil bringt. Es werden dennoch einige Tests mit dem CMA-ES Algorithmus bezüglich verschiedener Diskretisierungen des Spiegels durchgeführt, um diese Aussage zu untermauern. Des Weiteren wird eine Sensitivitätsanalyse bzgl. der Startschrittweite gemacht. Für BOBYQA ist die Startschrittweite durch ρ_{beg} und für den CMA-ES Algorithmus durch σ gegeben.

7.1.1.1 Slope Deviation Plots verschiedener Diskretisierungen

Die Diskretisierungsanalyse findet, wie in Abschnitt 5.2 beschrieben, statt. Tabelle 7.1 zeigt die Slope Deviation Plots und die SDx-Werte für verschiedene Diskretisierungen. Es ist zu erkennen, dass die ersten beiden Slope Deviation Plots mit einer Netzteilung von 10×10 und 25×25 Elementen wenig Ähnlichkeit mit dem Vergleichsmodell, welches 200×200 Elemente besitzt, aufweisen. Eine Diskretisierung mit mehr als 50×50 Elementen bildet eine gute Annährung zum feinsten betrachteten Netz. Des Weiteren fällt auf, dass je feiner die Diskretisierung gewählt wird, desto größer wird der Wert SDx. Für den Plot mit nur 10×10 Elementen ergibt sich ein Wert von SDx $= 0.953$ mrad, während SDx $= 1.587$ mrad für eine Diskretisierung von 200×200 gilt. Je feiner die Diskretisierung gewählt wird, desto stärker zeigt sich die Verformung und desto mehr Verformungsdetails werden sichtlich. Die Gegenüberstellung der Slope Deviation Plots gibt einen ersten Eindruck über die Wahl der verschiedenen Diskretisierungen.

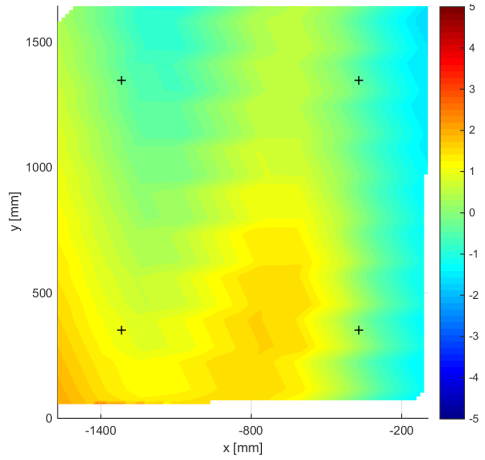
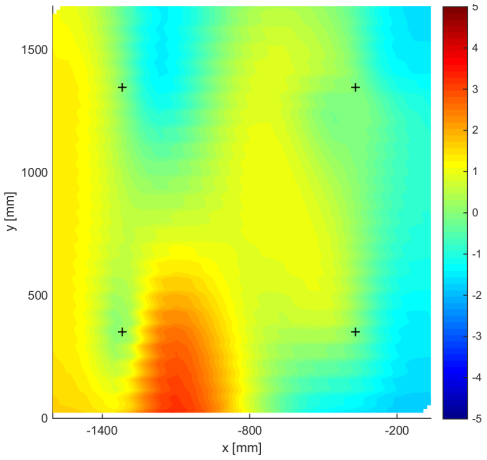
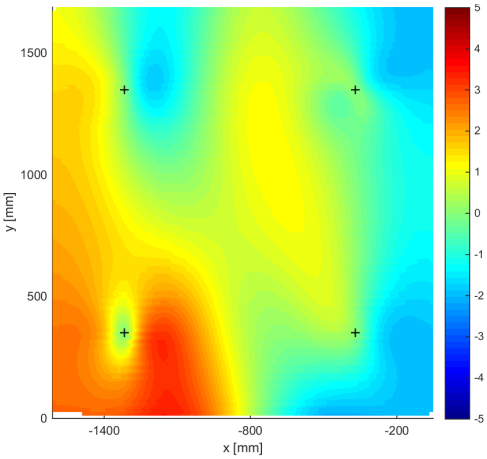
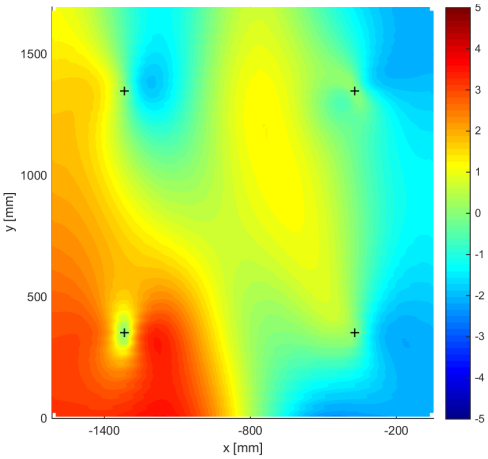
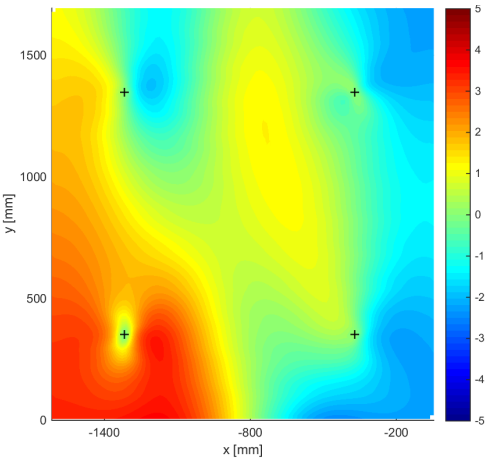
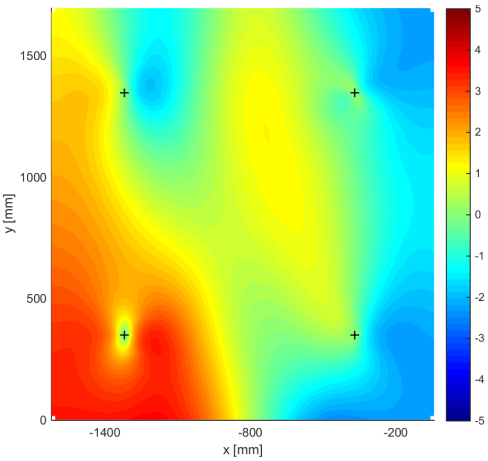
7.1.1.2 BOBYQA

Zunächst findet eine Anwendung des BOBYQA Algorithmus auf das Testproblem 1 statt. Das Ergebnis wird für verschiedene Diskretisierungen der Spiegelfacette verglichen. Des Weiteren wird der Test auch für unterschiedliche Startschrittweiten durchgeführt. Die Resultate dazu werden im Folgenden präsentiert. Die Tabellen A.1, A.2 und A.3 im Anhang zeigen die Ergebnisse der Netzanalyse von BOBYQA für eine Startschrittweite von $\rho_{beg} = 0.01$, $\rho_{beg} = 1.0$ und $\rho_{beg} = 2.0$.

Es zeigt sich klar, dass BOBYQA für jede Diskretisierungswahl und jede Startschrittweite das Optimum $[0.5, 1.5, 1.0]^T$ findet. Im dreidimensionalen Fall werden 7 Evaluationen benötigt, um das erste quadratische Modell zu approximieren. Im Anschluss findet die erste Iteration statt. Aufgrund dessen ist die Anzahl der Funktionsauswertungen im Vergleich zu den Iterationen immer um 7 erhöht. Außerdem stellt sich für jeden der Durchläufe bei der Wahl von verschiedenen Startschrittweiten heraus, dass der Algorithmus für eine Wahl von 50×50 Elementen eine schnellere Konvergenz aufweist, als für eine Netzteilung von 10×10 und 25×25. Den Durchläufen mit 50×50 Elementen gelingt es sowohl die geringste Anzahl an Iterationen als auch die kürzeste Laufzeit zu erreichen. Beispielsweise stoppt der Algorithmus im Fall $\rho_{end} = 0.01$ bereits nach 38 min. Am längsten, mit 122 min, benötigt der Test mit 200×200 Elementen und für $\rho_{end} = 1.0$.

Durch die erhöhte Anzahl an Elementen ist es möglich eine glattere Spiegelfacette zu simulieren. Da BOBYQA ein quadratisches Modell der Zielfunktion approximiert und dieses dann mittels konjugiertem Gradientenverfahren löst, verwendet dieser indirekt Ableitungsinformationen. Deshalb hat eine glattere Funktion einen positiven Einfluss auf den Optimierungsprozess beim BOBYQA Algorithmus. Bei einer Wahl von mehr als 50×50 Elementen kommt es wieder zu mehr benötigten Iterationen. Ferner fällt auf, dass der Algorithmus für 200×200 Elemente mit einer durchschnittlichen Zeit von 107 min besonders lange braucht. Dies liegt an der Feinheit der Diskretisierung der Spiegelfacette und der damit verbundenen langen Auswerteprozedur von Ansys.

Tabelle 7.1: Slope Deviation Plots von verschiedene Diskretisierungen des dreidimensionalen Testproblems

10×10 Elemente	25×25 Elemente
SDx = 0.953 mrad	SDx = 1.092 mrad
	
50×50 Elemente	75×75 Elemente
SDx = 1.386 mrad	SDx = 1.511 mrad
	
100×100 Elemente	200×200 Elemente
SDx = 1.556 mrad	SDx = 1.587 mrad
	

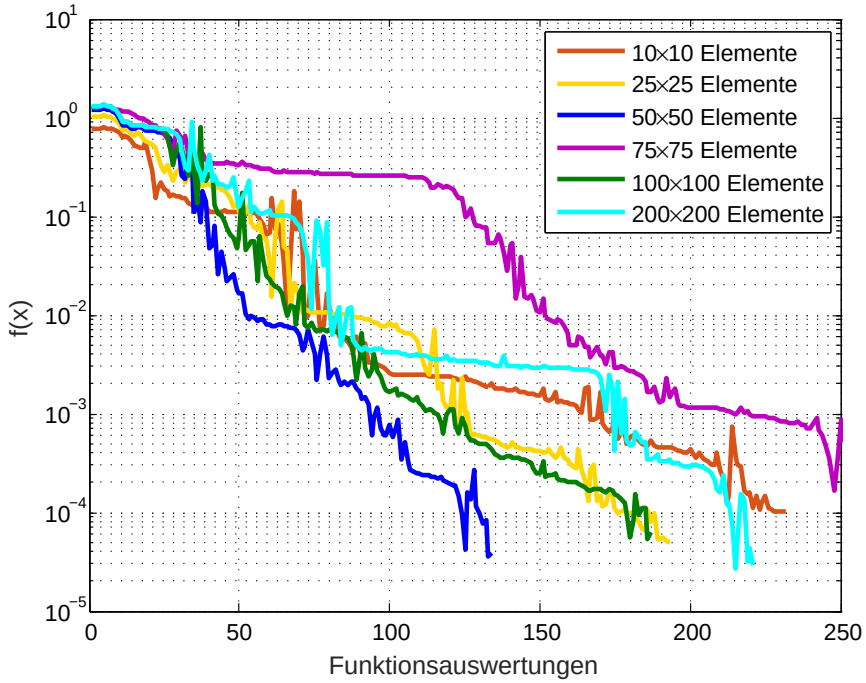


Abbildung 7.1: Konvergenz von BOBYQA für verschiedene Diskretisierungen mit $\rho_{beg} = 0.01$

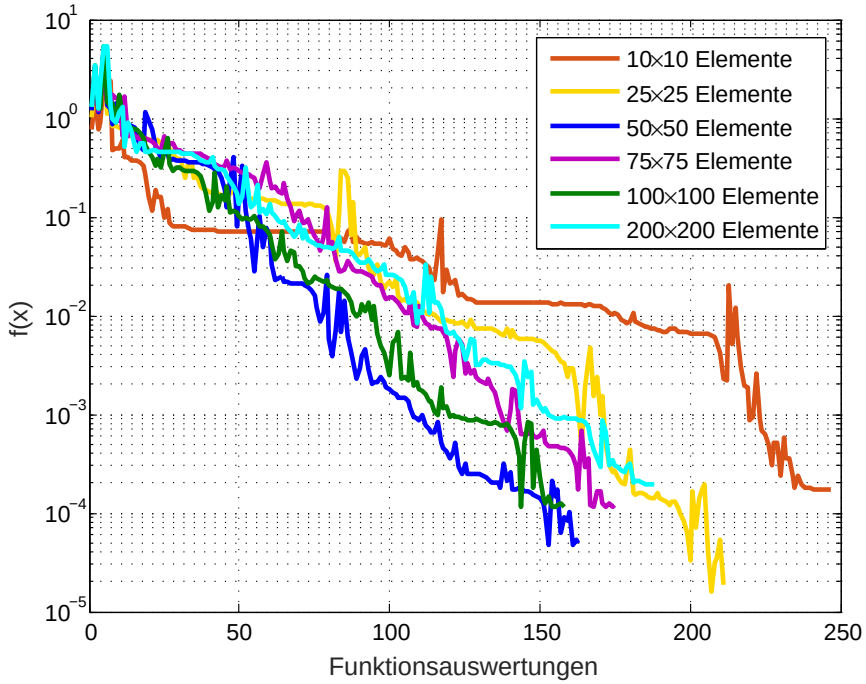


Abbildung 7.2: Konvergenz von BOBYQA für verschiedene Diskretisierungen mit $\rho_{beg} = 1.0$

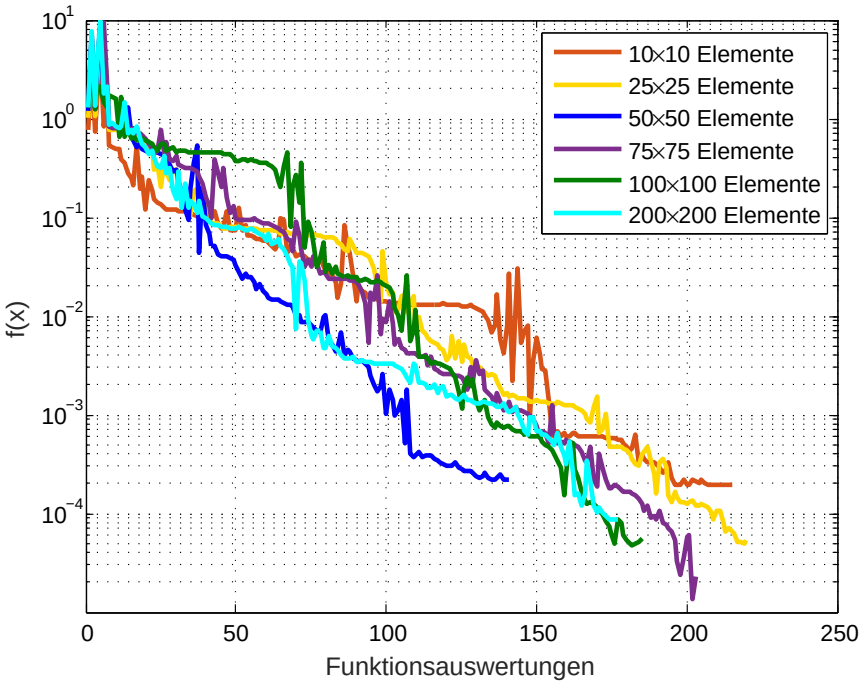


Abbildung 7.3: Konvergenz von BOBYQA für verschiedene Diskretisierungen mit $\rho_{beg} = 2.0$

Zur Veranschaulichung der Konvergenz der unterschiedlichen Testläufe wird der Zielfunktionswert logarithmisch über der aktuellen Funktionsauswertungszahl aufgetragen. Die Ergebnisse für die verschiedenen Startschrittweiten ρ_{beg} werden in den Abbildungen 7.1, 7.2 und 7.3 veranschaulicht. Auch hier zeigt sich nochmals die schnelle Konvergenz bei der Wahl von 50×50 Elementen. Darüber hinaus zeigt eine Diskretisierung von 100×100 ein vergleichsweise gutes Verhalten in Bezug auf Konvergenzgeschwindigkeit und Genauigkeit. Da für die unterschiedlichen Startschrittweiten die Diskretisierung von 50×50 für die Länge und Breite der Spiegelfacette immer gute Ergebnisse erzielt und auch die Zielfunktion gut annähert, wird für weitere Tests dieses Diskretisierung gewählt. Für die Startschrittweitenwahl ρ_{beg} hat sich keine eindeutige Wahl herauskristallisiert.

7.1.1.3 CMA-ES

Zuerst wird gezeigt, dass die Wahl der Diskretisierung keinen Einfluss auf die Lösung von CMA-ES hat. Dazu werden zwei Tests mit einer Netzteilung von jeweils 50×50 und 200×200 Elementen gemacht. Als Startschrittweite wird $\sigma = 1.0$ gewählt. Tabelle A.4 im Anhang zeigt die Ergebnisse der vier Durchläufe. Es fällt auf, dass diese mit einer Wahl von 200×200 Elementen zeitlich mehr als doppelt so lange für die Auswertung brauchen als die Tests mit einer Netzteilung von 50×50. Während die Test mit 50×50 Elementen eine Zeit von 70 min benötigen, stoppen die Durchläufe mit einer Netzteilung von 200×200 nach 160 min. In allen Fällen stoppt der Algorithmus mit der maximalen Anzahl an Funktionsauswertungen von 250. Die Exaktheit der Lösugen liegt zwischen $5.352 \cdot 10^{-2}$ und $1.090 \cdot 10^{-2}$ mrad. Der Algorithmus nähert sich dem Optimum

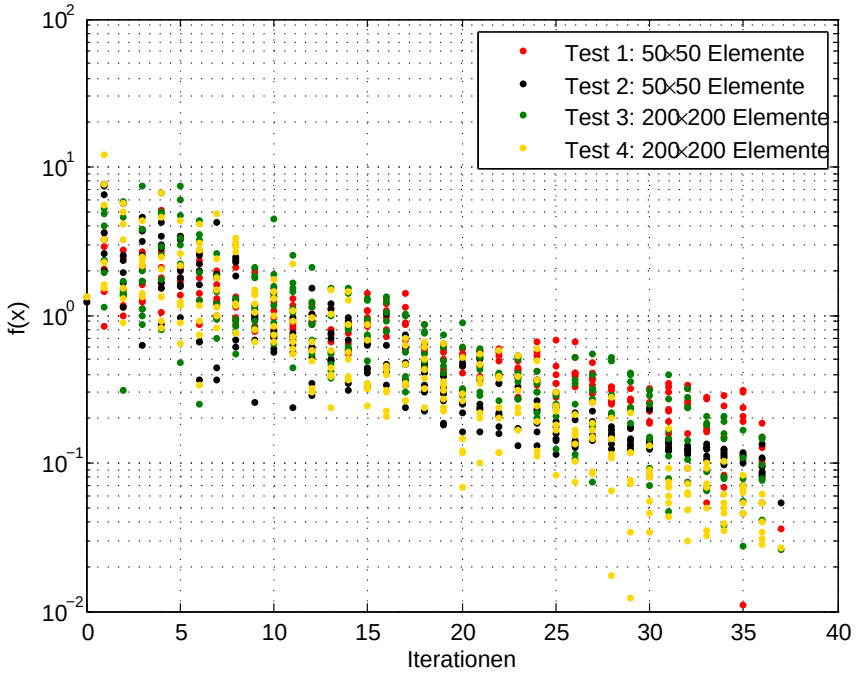


Abbildung 7.4: Konvergenz von CMA-ES für verschiedene Diskretisierungen mit $\sigma = 1.0$

$[0.5, 1.5, 1.0]^T$ für alle Versuche. Des Weiteren ist zu erkennen, dass Test 1 mit einer Teilung von 50×50 zwar das genauste Ergebnis liefert, dafür hat der zweite Test mit dieser Diskretisierung das schlechteste Resultat gebracht. Hier zeigt sich das stochastische, auf Wahrscheinlichkeiten beruhende Verhalten des Verfahrens. Für einige Durchläufe erreicht der Algorithmus gute Werte, wobei er für den gleichen Test mit identischen Startbedingungen auch vergleichsweise schlechte Ergebnisse produzieren kann. Es ist jedoch keine Tendenz für ein bestimmtes Netz erkennbar. Abbildung 7.4 verdeutlicht die sprunghafte, stochastische Suche in verschiedene Richtungen des Suchraums. Deutlich zu erkennen ist, dass sich keine eindeutige Diskretisierung für das CMA-ES Verfahren ableiten lässt. Beispielsweise zeigt Testlauf 4 die kleinsten Zielfunktionswerte über den ganzen Optimierungsprozess im Vergleich zu den anderen Durchläufen, dennoch gelingt es Test 1 mit einer Netzteilung von 50×50 Elementen die beste Lösung zu generieren.

Nachfolgend werden verschiedene Startschrittweiten für den CMA-ES Algorithmus getestet. Tabelle A.5 und Abbildung 7.5 zeigen die Ergebnisse für $\sigma = 0.01$. Tabelle A.5 im Anhang zeigt, dass für den ersten Test der y_1-Wert mit $y_1 = 0.603$ mit einer prozentualen Abweichung von 60 % weit vom Optimum $y_1 = 1.5$ entfernt ist. Dies macht sich auch im Funktionswert von $\triangle\text{SDx} = 3.391 \cdot 10^{-1}$ mrad bemerkbar. Der hohe Wert lässt auf eine relativ schlechte Lösung schließen. Test 2 fällt deutlich besser aus. Hier wird ein minimaler Zielfunktionswert von $9.390 \cdot 10^{-2}$ mrad erreicht. Dies wird in dem deutlich besseren Lösungvektor sichtbar. Der Algorithmus bricht in beiden Fällen mit der maximalen Anzahl an Funktionsauswertungen ab. Durch die Wahl der kleinen Startschrittweite sucht der Algorithmus nur in einer kleinen Umgebung um den Startwert nach besseren Lösungen, weshalb er relativ lange für das Finden einer akkuraten Lösung braucht. Abbildung 7.5 verdeutlicht dieses Verhalten. Zu Beginn des

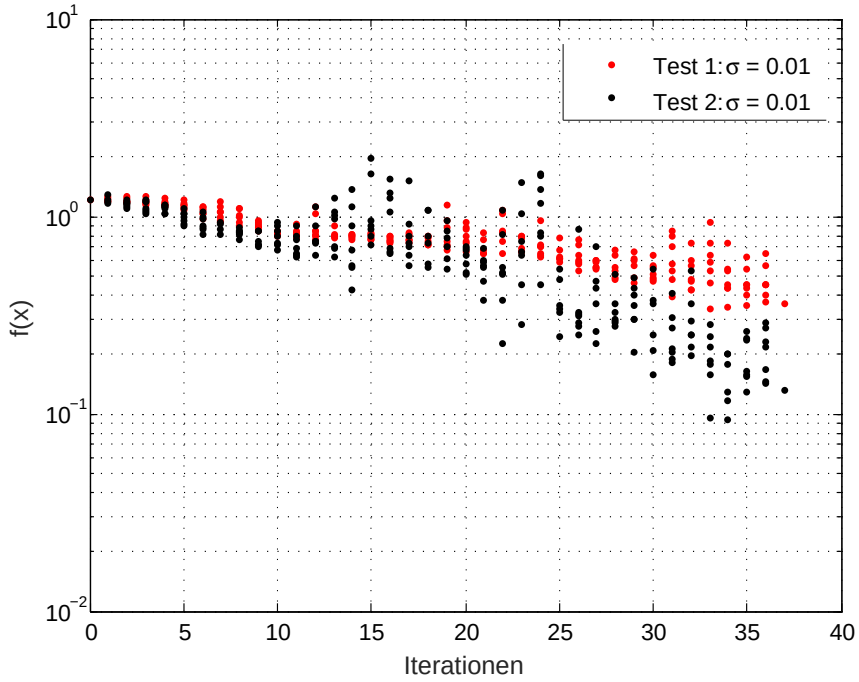


Abbildung 7.5: Konvergenz von CMA-ES mit $\sigma = 0.01$

Tests sind die Zielfunktionswerte noch beisammen, statt wie für einen evolutionären Algorithmus typisch eine große Vielfalt an Lösungsvorschlägen zu präsentieren. Erst nach einigen Iterationen ist erkennbar, dass die Zielfunktionswerte ein breites Spektrum an Werten annehmen. Aufgrund der zu kleinen Wahl der Startschrittweite zeigt der Algorithmus hier ein vergleichsweise schlechtes Verhalten. Deshalb soll die Startschrittweite auf $\sigma = 1.0$ hochgesetzt werden, um dem Algorithmus dadurch eine bessere Ausgangssituation zu liefern und den Optimierungsprozess zu verbessern.

Die Resultate des Durchlaufes finden sich in Tabelle A.6 im Anhang und Abbildung 7.6 wieder. Diese werden bereits bei der Diskretisierungsanalyse in Abschnitt 7.1.1 vorgestellt. Der Algorithmus bircht mit dem Erreichen der maximalen Anzahl an Evaluationen ab. Es wird deutlich, dass die beiden minimalen Zielfunktionswerte kleiner und somit genauer sind als für den Fall $\sigma = 0.01$. Das Optimum wird in beiden Fällen mit einer Exaktheit von $1.090 \cdot 10^{-2}$ mrad beziehungsweise $5.352 \cdot 10^{-2}$ mrad gefunden. Abbildung 7.6 zeigt, dass der Algorithmus bereits nach dem Start eine Bandbreite an verschiedenen Zielfunktionswerten ermittelt. Auffällig ist, dass obwohl Test 2 einen insgesamt besseren Verlauf präsentiert, im ersten Test der kleinste Zielfunktionswert und damit der exaktere Lösungsvektor ermittelt wird.

Zuletzt wird die Startschrittweite $\sigma = 2.5$ gewählt. In Tabelle A.7 und Abbildung 7.7 werden die Ergebnisse präsentiert. Auch hier stoppt der Algorithmus mit der maximalen Anzahl an Auswertungen und einer Zeit von 72 min. Damit bleibt die geforderte Genauigkeit erneut unerreicht. Hier zeigt sich wieder eine relativ große Spannung in der Güte der Ergebnisse. Während Test 2 eine Genauigkeit von $\triangle \mathrm{SDx} = 1.892 \cdot 10^{-2}$ erreicht, zeigt der erste Test einen um ein Zehntel schlechteren Zielfunktionswert. Dies ist auch in der Abbildung 7.7 dargestellt. Durchlauf

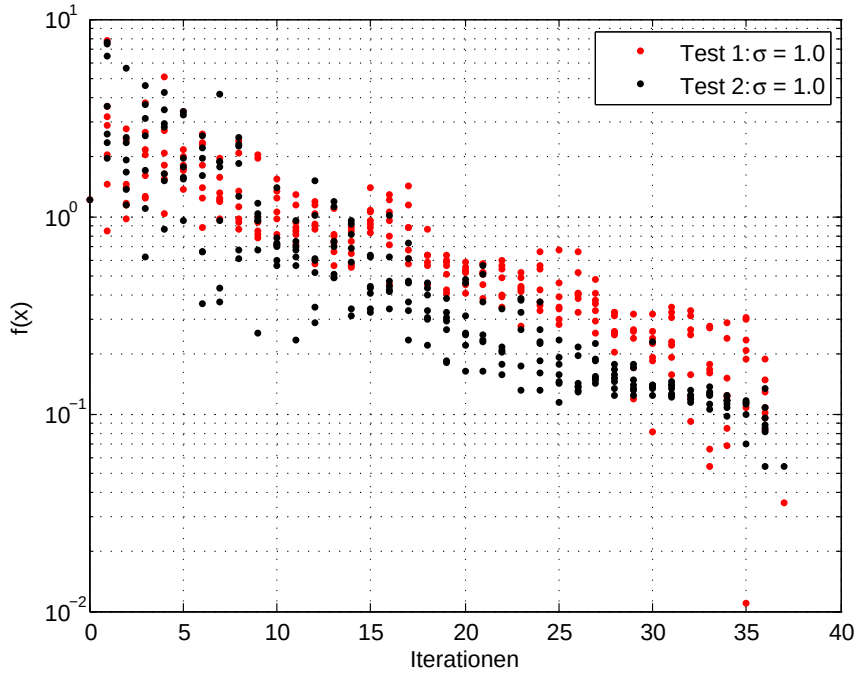


Abbildung 7.6: Konvergenz von CMA-ES mit $\sigma = 1.0$

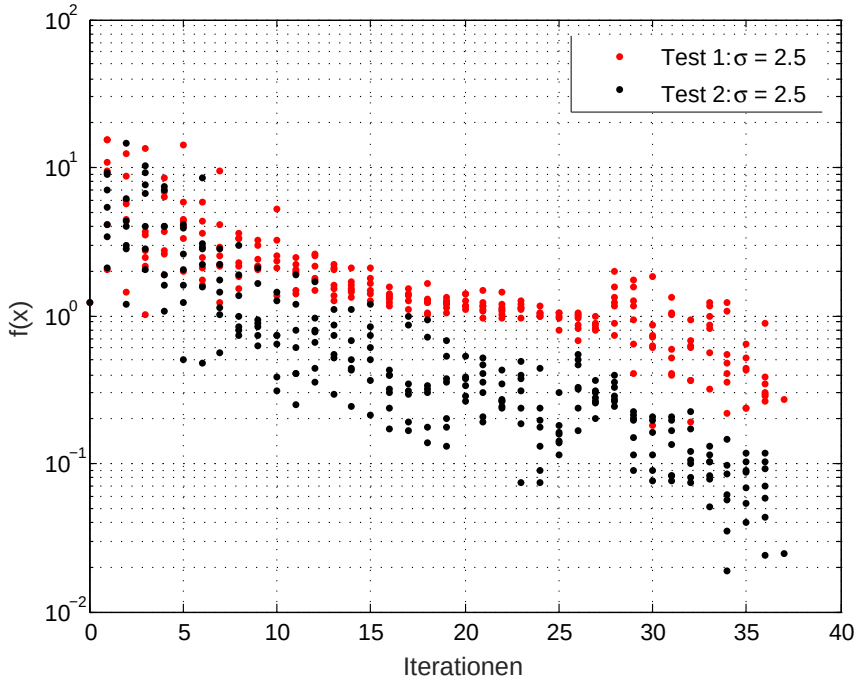


Abbildung 7.7: Konvergenz von CMA-ES mit $\sigma = 2.5$

2 erreicht einen steileren Abstieg. Test 1 zeigt im Bereich zwischen 15 und 25 Iterationen ein stagnierendes Verhalten, während die Funktionswerte beim zweiten Test breit gestreut sind. Von Hansen wird eine Startschrittweite σ zwischen 0.2 und 0.5 mal der Länge des Intervalls empfohlen, um eine vernünftige globale Suchperformance zu erreichen. Damit passt $\sigma = 2.5$ in den empfohlenen Bereich für eine gute Leistung des Algorithmus.

7.1.1.4 Testproblem 1: Vergleich von BOBYQA und CMA-ES

Nun folgt eine erste Gegenüberstellung der beiden Algorithmen. Für den Vergleich wird ein Spiegelmodell mit 50×50 Elementen und einer Startschrittweite von 1 gewählt. In Tabelle 7.2 sind die wichtigsten Daten gelistet. Auffällig ist die Laufzeit beim Vergleich der beiden Algorithmen. Während der Algorithmus von Powell nur 48 min benötigt, um eine Genauigkeit von $4.703 \cdot 10^{-5}$ zu erreichen, stoppt der CMA-ES mit der maximalen Anzahl an Funktionsauswertungen, einer Zeit von 72 min und einer Exaktheit von $\triangle \mathrm{SDx} = 1.090 \cdot 10^{-2}$ mrad. Abbildung 7.8 veranschaulicht das unterschiedliche Abstiegsverhalten der beiden Algorithmen. Um die beiden unterschiedlichen Verfahren vergleichen zu können, wird auf der x-Achse die Anzahl an Funktionsauswertungen aufgetragen. Außerdem werden die Werte punktförmig geplottet. Deshalb geht in dieser Grafik zwar das populationsbasierte Verhalten des CMA-ES Verfahrens verloren, aber es wird verdeutlicht wie viele Funktionsauswertung dieser benötigt. Sofort erkennbar ist, dass BOBYQA im Vergleich mit CMA-ES schneller konvergiert. Bereits nach ca. 150 Funktionsauswertungen wird das gegebene Abbruchkriterium von CMA-ES mit 10^{-4} von BOBYQA erreicht. Nach einigen weiteren Iterationen bricht BOBYQA mit dem Erreichen des minimalen Trust-Region Radius

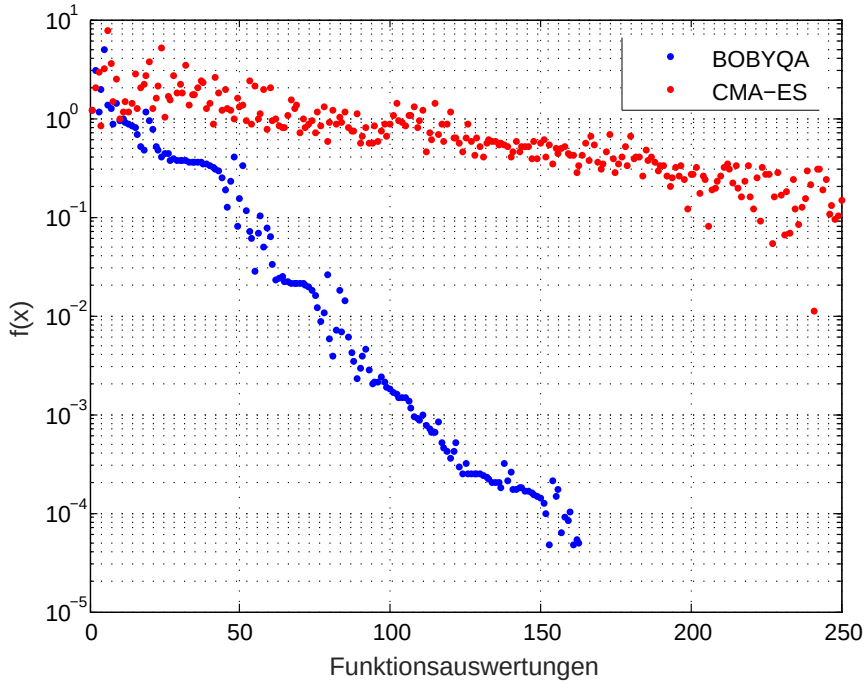


Abbildung 7.8: Konvergenz von BOBYQA und CMA-ES

Tabelle 7.2: Resultate von Testproblem 1 für BOBYQA und CMA-ES

	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
BOBYQA	0.500	1.500	1.000	$4.703 \cdot 10^{-5}$	156	163	48
CMA-ES	0.496	1.494	0.996	$1.090 \cdot 10^{-2}$	36	254	72

ab. Der CMA-ES Algorithmus schafft in den letzten Iterationen einen Sprung zum, für den evolutionären Algorithmus, minimal berechneten Zielfunktionswert.

7.1.2 Vergleich zwischen Klebstoff- und Padmodell

Erste Tests haben gezeigt, dass eine Diskretisierung von 50×50 Elementen zu einer relativ schnellen Funktionsauswertung führt. Aufgrund dessen wird für weitere Durchläufe zunächst diese Diskretisierung gewählt. Um noch schnellere Funktionsauswertungen zu erreichen werden die vier Montierungspads weggelassen und die Fixierungen beziehungsweise Verschiebungen an den Klebstoffunterseiten definiert. Ein Vergleich der unterschiedlichen Modelle findet in diesem Abschnitt statt.

7.1.2.1 Slope Deviation Plots für Klebstoff- und Padmodell

Abbildung 7.9 zeigt links den Slope Deviation Plot in x des Klebstoffmodells und auf der rechten Seite den Plot für das Padmodell mit 50×50 Elementen. Bei einer Verschiebung des ersten Aufhängepunktes um $[0.5, 1.5, 1.0]^T$ bei beiden Modellen, zeigt sich, dass zwar die gleichen Bereiche in den verschiedenen Modellen verbogen sind, jedoch ist die Verformung

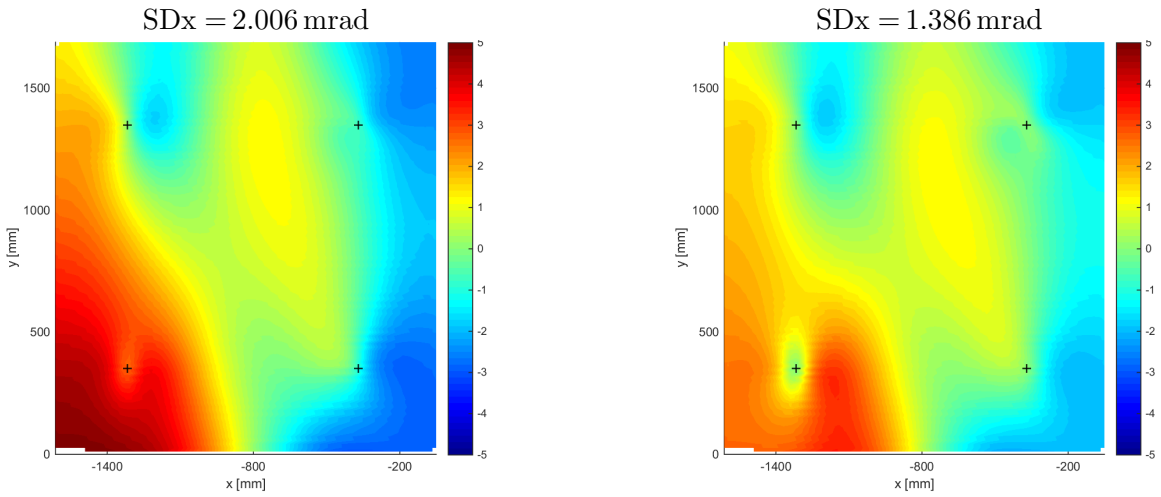


Abbildung 7.9: Slope Deviation Plots des Klebstoffmodells (links) und des Padmodells (rechts)

beim Klebstoffmodell sehr viel deutlicher ausgeprägt. Dies liegt daran, dass sich der Kleber leichter verformen lässt als die vier relativ steifen Montierungspads. Während das Padmodell einen Slope Deviation-Wert von SDx = 1.386 mrad besitzt, liegt dieser beim Klebstoffmodell bei SDx = 2.006 mrad. Die Slope Deviation Plots der Spiegelfacetten stellen die Referenz für das jeweilige Modell dar. Sie werden durch das Optimierungsmodell von den beiden Algorithmen nachbildet.

7.1.2.2 BOBYQA

In Tabelle A.8 im Anhang und Abbildung 7.10 sind die Ergebnisse der beiden Testdurchläufe mit dem BOBYQA Algorithmus dargestellt. BOBYQA findet das Optimum bei beiden Modellen mit genügender Genauigkeit von $f(x_*) = 4.703 \cdot 10^{-5}$ für das Padmodell bzw. mit $2.130 \cdot 10^{-4}$ mrad für das Klebstoffmodell. Es wird in Abbildung 7.10 deutlich, dass beide Modelle ein ähnliches Konvergenzverhalten zeigen. Das Klebstoffmodell bricht jedoch früher ab, während das Padmodell eine höhere Genauigkeit erreicht. Auffällig ist die Zeit, die der Algorithmus benötigt. Während das Verfahren bei der Anwendung auf das Padmodell 48 min braucht bis der Algorithmus stoppt, sind es beim Klebstoffmodell nur 14 min. Damit wird eine Zeitersparnis bei der Verwendung des Klebstoffmodells von mehr als 70 % im Vergleich zum Padmodell erreicht. Die Zeit kann für eine Iteration bestimmt werden. Das Padmodell benötigt 17 sec und das Klebstoffmodell 7 sec für eine Auswertung. Da die Optimierung nicht nur bei einem Pad, also im Dreidimensionalen, bleiben soll, sondern auf weitere Dimensionen und damit Verschiebungen und Verkippungen von mehreren Pads betrachtet werden, ist eine schnellere Funktionsauswertung wünschenswert. Aus diesem Grund wird für die nächsten Untersuchungen auf das Klebstoffmodell zurückgegriffen, um zunächst die Algorithmen zu optimieren. Erst später wird wieder das Padmodell betrachtet.

7.1.2.3 CMA-ES

Tabelle A.9 im Anhang und Abbildung 7.11 zeigen die Resultate der Durchläufe mit dem CMA-ES Algorithmus angewendet auf das Klebstoffmodell und das Padmodell. Da das CMA-ES Verfahren stochastisch arbeitet, werden zwei Durchläufe für jedes Modell gemacht. Alle Durchläufe stoppen mit der maximalen Anzahl an Funktionsauswertungen und nähern sich dem Optimum mit einer Genauigkeit zwischen $5.352 \cdot 10^{-2}$ und $3.032 \cdot 10^{-3}$ mrad. Für die gleiche Anzahl an Iterationen ist der erste Test im Vergleich zu Test 2 um einen Wert von $4.068 \cdot 10^{-2}$ schlechter. Auch hier ist wieder erkennbar, dass der Test mit dem Klebstoffmodell ein viel schnelleres Laufzeitverhalten im Vergleich zum Padmodell zeigt. Während das Padmodell mehr als 70 min braucht, benötigt das Klebstoffmodell nur 30 min. Die Zeit für eine Iteration beträgt wie bereits beim BOBYQA Algorithmus ungefähr 17 sec für das Padmodell und 7 sec beim Klebstoffmodell. Aufgrund der Ähnlichkeiten der Zielfunktionen von Klebstoff- und Padmodell und der deutlichen Laufzeitoptimierung der beiden Algorithmen wird für erste Tests auf das Klebstoffmodell zurückgegriffen.

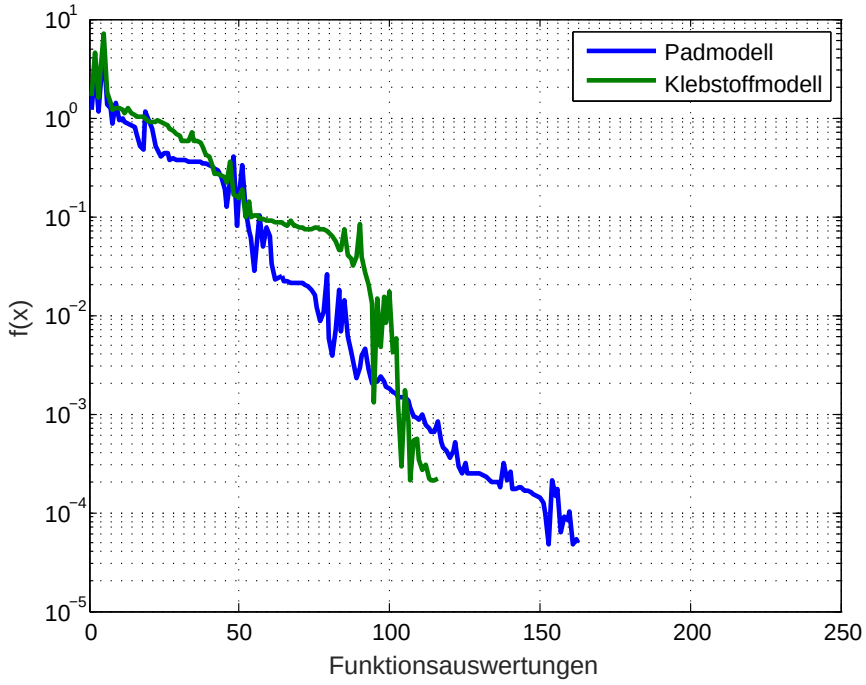


Abbildung 7.10: Konvergenz von BOBYQA für das Klebstoff- und Padmodell

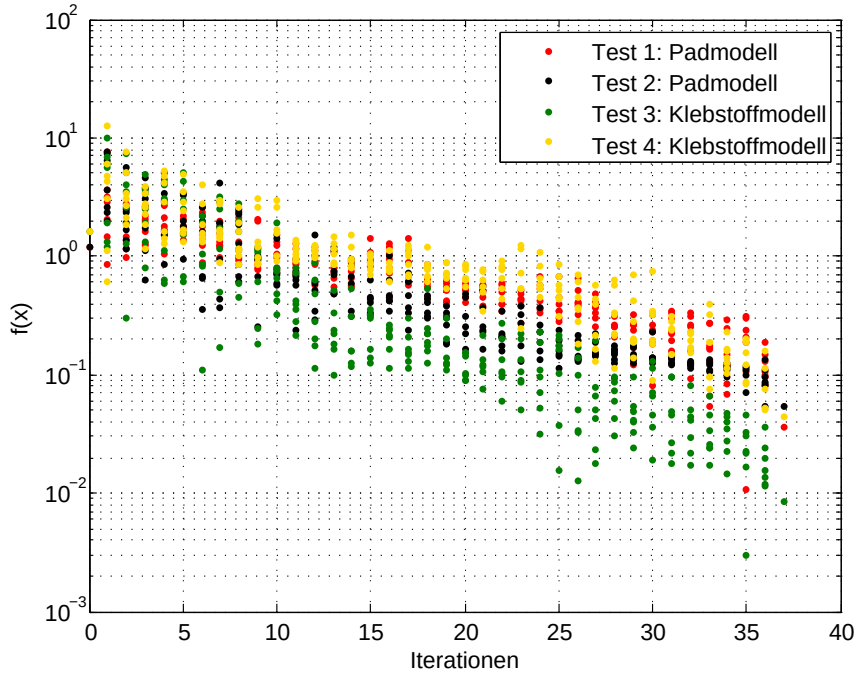


Abbildung 7.11: Konvergenz von CMA-ES für das Klebstoff- und Padmodell

7.1.3 Testproblem 2: Vergleich von BOBYQA und CMA-ES

Das Testproblem 2 aus Abschnitt 6.1.1.2 mit der konstruierten Lösung $[1,1,1]^T$ wird auf die beiden Algorithmen angewendet. Für beide Verfahren wird der Startwert $x_0 = [0,0,0]^T$ sowie die Grenzen $a_i = -5$ und $b_i = 5$ für $i = 1,2,3$ gewählt. Es werden maximal 200 Evaluationen pro Durchlauf erlaubt. Es wird für den BOBYQA Algorithmus der Start-Trust-Region auf $\rho_{beg} = 1$ und der finale Trust-Region Radius auf $\rho_{end} = 10^{-4}$ gesetzt. Als Abbruchkriterium für CMA-ES wird zusätzlich eine geforderte Genauigkeit $\epsilon_{end} = 10^{-3}$ definiert. Des Weiteren macht der CMA-ES Algorithmus 7 Funktionsauswertungen pro Iteration. Für die Gravitation gilt $g = [0,0,9810]^T$.

Tabelle 7.3: Resultate von Testproblem 2 für BOBYQA und CMA-ES

	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
BOBYQA	1.000	1.000	1.000	$5.809 \cdot 10^{-4}$	154	161	20
CMA-ES	1.010	1.325	0.982	$1.760 \cdot 10^{-1}$	29	205	23

In Abbildung 7.12 ist die Konvergenz des BOBYQA Algorithmus sowie die des CMA-ES Verfahrens geplottet. Auf der x-Achse sind die Funktionsauswertungen aufgetragen. Es zeigt sich deutlich, dass der CMA-ES Algorithmus mehr Funktionsauswertungen benötigt als das BOBYQA Verfahren. Während BOBYQA nach 161 Funktionsauswertungen stoppt, da der Trust-Region mit dem finalen Trust-Region Radius übereinstimmt, iteriert der CMA-ES bis die maximale

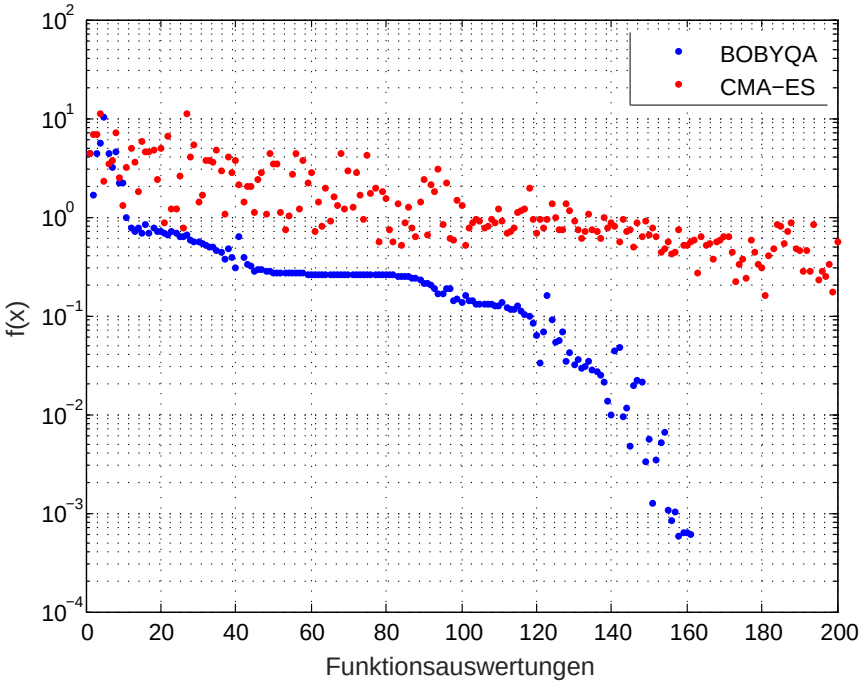


Abbildung 7.12: Konvergenz von BOBYQA und CMA-ES

Anzahl an Funktionsauswertungen erreicht ist. Trotzdem kann er sich der Genauigkeit, welche der BOBYQA Algorithmus erreicht, nicht nähern. Für die ersten 200 Funktionsauswertungen zeigen beide Algorithmen ein relativ ähnliches Verhalten, wobei BOBYQA einen etwas kleineren Zielfunktionswert aufweist. Dann fällt die Kurve beim BOBYQA Algorithmus jedoch stark ab, wobei die des CMA-ES weiter leicht abnimmt. Tabelle 7.3 zeigt unter anderem die beste Lösung, welche die Algorithmen erreichen und den besten Zielfunktionswert. BOBYQA erzielt eine Genauigkeit von $5.809 \cdot 10^{-4}$ mrad, während der kleinste Zielfunktionswert bei CMA-ES bei $\triangle\text{SDx} = 1.760 \cdot 10^{-1}$ liegt.

7.2 Sechsdimensionales Problem

Für die nächsten Optimierungsversuche wird das sechsdimensionale Testproblem aus Abschnitt 6.1.2 betrachtet. Damit sind sechs Optimierungsparameter gegeben und die Zielfunktion ist durch $f : \mathbb{R}^6 \to \mathbb{R}, (x_1, y_1, z_1, x_3, y_3, z_3)^T \mapsto f(x_1, y_1, z_1, x_3, y_3, z_3)$ definiert. Des Weitern gelten für die Grenzen $a_i = -5$ und $b_i = 5$ für $i = 1, ..., 5$. Die Optimierer werden auf das Testproblem angewendet und die Ergebnisse dieser Tests sind nachfolgend dargestellt. BOBYQA und CMA-ES werden hinsichtlich ihrer Genauigkeit und Laufzeit untersucht und eventuell verifiziert und optimiert. Als Optimierungsmodell dient das FE-Klebstoffmodell.

7.2.1 Test 1

Das Testproblem wird von den beiden Algorithmen gelöst. Das Optimum befindet sich hier im Punkt $[1, 1, 1, 0, 0, 0]^T$. Als Startwert wird $x_0 = [0, 0, 0, -1, 1, 1]^T$ gewählt. Die maximale Anzahl an Funktionsauswertungen beträgt 900. Die Startschrittweite beim BOBYQA Algorithmus wird auf $\rho_{beg} = 1$ gesetzt. Als Abbruchkriterium gilt für BOBYQA $\rho_{end} = 10^{-4}$ und der CMA-ES Algorithmus bricht mit einer Genauigkeit von 10^{-3} ab. Für CMA-ES wird die für sechsdimensionale Probleme empfohlene Populationsgröße von 9 Individuen und Startschrittweite $\sigma = 2.5$ übernommen. Damit ist 100 die maximale Anzahl an Iteration.

Tabelle A.10 im Anhang listet die wichtigsten Werte der Testläufe auf. Bei der Betrachtung der Spalte mit den Funktionsauswertungen, fällt auf, dass der BOBYQA Algorithmus bereits nach 661 Funktionsauswertungen abbricht, während CMA-ES die maximale Anzahl an Iterationen benötigt. BOBYQA erreicht mit $f(x_*) = \triangle\text{SDx} = 1.384 \cdot 10^{-3}$ mrad im Vergleich zu CMA-ES mit $7.628 \cdot 10^{-2}$ mrad eine höhere Genauigkeit. Abbildung 7.13 präsentiert die Ergebnisse nochmals grafisch. Wenige Auswertungen nach dem Optimierungsstart erreicht BOBYQA bereits Zielfunktionswerte kleiner als 1. Dies gelingt CMA-ES erst nach fast 200 Funktionsauswertungen. Des Weiteren sieht die Abstiegskurve von BOBYQA treppenförmig mit wenigen Ausreißern aus, während CMA-ES eher ein sprung- und schwarmhaftes Verhalten zeigt.

7.2.2 Test 2

Der Startwert für diesen Optimierungslauf wird auf $x_0 = [-1, -1, -1, -1, 1, 1]^T$ gesetzt. Für den CMA-ES Algorithmus werden maximal 1494 Funktionsauswertungen zugelassen. Bei einer Populationsgröße von 9 Individuen, sind das maximal 166 zugelassene Iterationen. Des Weiteren gilt $\sigma = 2.5$ und $\epsilon_{end} = 10^{-3}$.

CMA-ES bricht nach 160 Iterationen, das heißt 1442 Funktionsauswertungen mit der geforderten Genauigkeit, ab. Die Optimierungsergebnisse sind in Tabelle A.11 im Anhang zu entnehmen. Wie weit kommt der BOBYQA Algorithmus mit 1442 Funktionsauswertungen? Dazu wird BOBYQA mit einer Startschrittweite $\rho_{beg} = 1$ und einem finalen Trust-Region von $\rho_{end} = 1 \cdot 10^{-4}$ auf das Testproblem angewendet. Ein zweiter Test mit $\rho_{end} = 10^{-6}$ wird durchgeführt.

Die Resultate des BOBYQA Algorithmus, sowie die Ergebnisse des CMA-ES Durchlaufs sind in Tabelle A.11, sowie Abbildung 7.14 festgehalten. Es ist zu erkennen, dass immer das geforderte Optimum gefunden wird. Der CMA-ES und der BOBYQA mit $\rho_{end} = 1 \cdot 10^{-4}$ erreichen sehr ähnliche Zielfunktionswerte mit $2.218 \cdot 10^{-4}$ mrad für den CMA-ES Algorithmus und $\triangle$SDx $= 1.514 \cdot 10^{-3}$ mrad für den Fall mit $\rho_{end} = 10^{-4}$. Jedoch schafft es BOBYQA diese Genauigkeit mit unter 800 Funktionsauswertungen zu erreichen, während der CMA-ES 1442 Auswertungen benötigt. Zum Vergleich wird auch der BOBYQA mit einem finalen Trust-Region von $\rho_{end} = 1 \cdot 10^{-6}$ geplottet. Es ist eine langsamere Konvergenzgeschwindigkeit im Vergleich zum ersten BOBYQA Durchlauf erkennbar, jedoch erreicht der zweite Testlauf einen exakteren Lösungsvektor x_* und damit den kleinsten Zielfunktionswert mit $f(x_*) = \triangle SDx = 9.053 \cdot 10^{-4}$ mrad.

7.2.3 Test 3

Als Startvektor wird $x_0 = [0, 0, 0, -0.5, 0, 0]^T$ gewählt und es werden maximal 1000 Funktionsauswertungen zugelassen. Alle anderen nicht beschriebenen Größen bleiben wie bei den vorherigen Tests. Zunächst wird der BOBYQA mit einer Startschrittweite $\rho_{end} = 10^{-4}$, sowie $\rho_{end} = 1 \cdot 10^{-6}$ getestet. Abbildung 7.15 zeigt grafisch die Ergebnisse der Durchläufe. Weitere Details zur gefundenen Lösung oder beispielsweise zu der benötigten Anzahl an Iterationen können in Tabelle A.12 im Anhang erfahren werden. Das Optimum wird bei allen Durchläufen gefunden, jedoch unterscheidet sich die Genauigkeit. Auch hier zeigt sich wieder, dass BOBYQA mit $\rho_{end} = 10^{-6}$ exakter ist, jedoch auch langsamer konvergiert. Während für den Fall mit $\rho_{end} = 10^{-4}$ eine Genauigkeit von $1.445 \cdot 10^{-3}$ mrad bei 994 Funktionsauswertungen und einer Zeit von 3 h:06 min erreicht wird, gelingt es im Fall $\rho_{end} = 10^{-6}$ eine Exaktheit von $f(x_*) = 2.400 \cdot 10^{-4}$ mrad bei 1001 Funktionsauswertungen und einer Zeit von 3 h:13 min zu erzielen.

Nun wird der CMA-ES Algorithmus, mit dem Ziel eine schnellere Konvergenzgeschwindigkeit und Genauigkeit zu erreichen, optimiert. Dazu wird zunächst die Populationsgröße von 9 Individuen auf 4 verringert. Dadurch wird eine stärkere Fokussierung des Algorithmus und damit eine schnellere

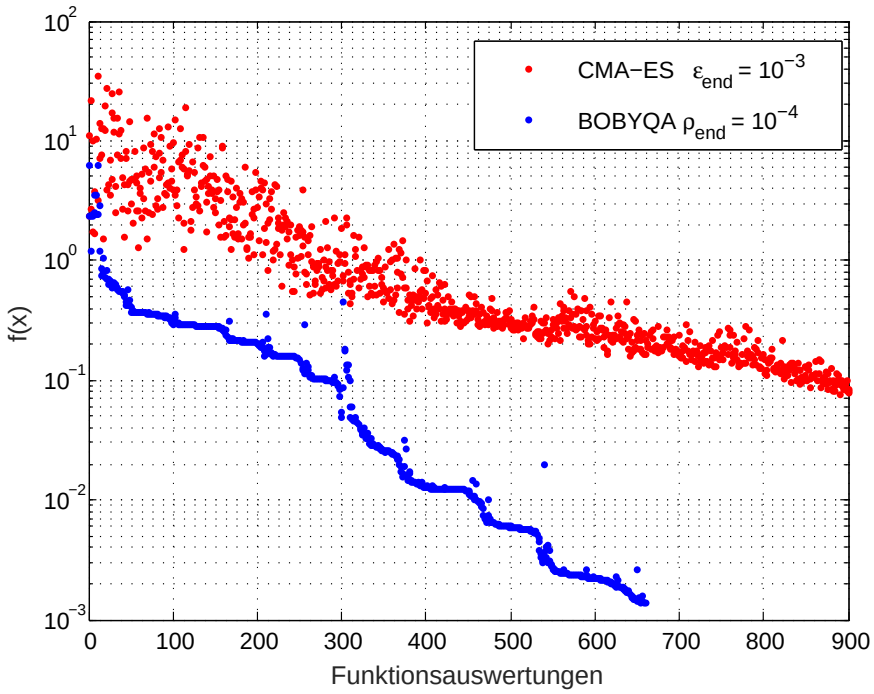


Abbildung 7.13: Konvergenz von BOBYQA und CMA-ES

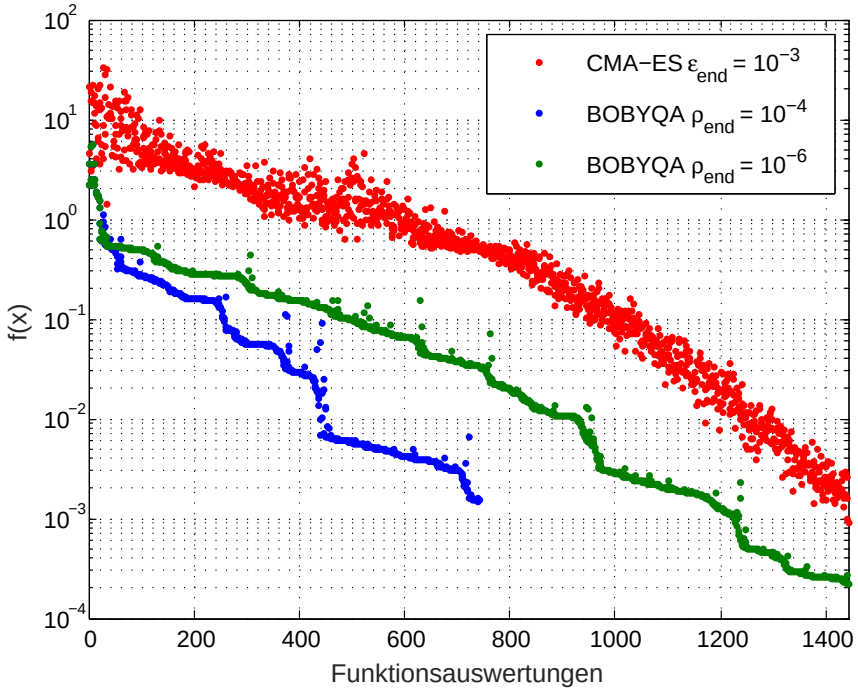


Abbildung 7.14: Konvergenz von BOBYQA und CMA-ES

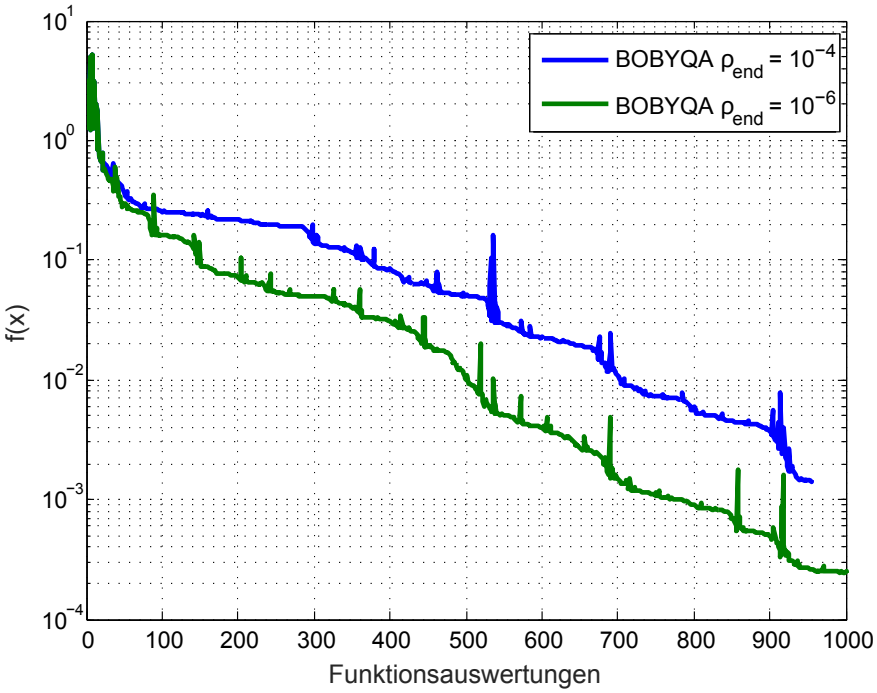


Abbildung 7.15: Konvergenz von BOBYQA für verschiedene Startschrittweiten

Konvergenz erhofft. Abbildung 7.16 zeigt zwei Testläufe des CMA-ES mit kleinerer Population. Zum Vergleich wird auch das Ergebnis für eine Population mit $\lambda = 9$ geplottet. Alle Testläufe brechen mit der maximalen Anzahl an Funktionsauswertungen ab. Während Testdurchlauf 2 anfangs den stärksten Abstieg aufweist, stagniert er nach 400 Funktionsauswertungen fast und zeigt erst wieder nach 600 Auswertungen ein abfallendes Verhalten. Test 1 hingegen weist erst nach 600 Iterationen einen starken Abfall der Zielfunktionswerte auf. Der Vergleichstest mit 9 Individuen schneidet anfangs gegenüber den Durchläufen mit kleiner Population schlechter ab, schafft es am Ende mit einer Exaktheit von $f(x_*) = 2.537 \cdot 10^{-2}$ mrad genauer als Test 2 mit $4.728 \cdot 10^{-2}$ mrad zu sein. Den besten Wert erreicht Test 1 mit $2.699 \cdot 10^{-3}$ mrad. Weitere Details zu den Optimierungsergebnissen finden sich in Tabelle A.12 im Anhang. Folglich kann die Reduktion der Populationsgröße einen Vorteil für die Konvergenzgeschwindigkeit bringen, jedoch sinkt durch die Verringerung der Individuenzahl auch die Wahrscheinlichkeit für das Finden eines globalen Optimums.

Als nächstes werden für den CMA-ES Algorithmus verschiedene Startschrittweiten getestet und miteinander verglichen. Für $\sigma = 0.5$, $\sigma = 1.0$ und $\sigma = 2.5$ ist das Ergebnis der Optimierung in Abbildung 7.17 veranschaulicht. Dieses Ergebnis zeigt erstaunlicherweise für $\sigma = 0.5$ das beste Verhalten. Es wird für $\sigma = 0.5$ eine Genauigkeit von $f(x_*) = \triangle\text{SDx} = 9.419 \cdot 10^{-4}$ mrad erreicht. Für $\sigma = 2.5$ gelingt es einen Wert von $2.537 \cdot 10^{-2}$ mrad zu erzielen und für $\sigma = 1.0$ einen minimalen Zielfunktionswert $f(x_*) = 9.905 \cdot 10^{-3}$ mrad . In Tabelle A.12 sind die Ergebnisse dieser Optimierung präsentiert. Da alle Durchläufe mit der maximalen Anzahl an Funktionsauswertungen stoppen, beträgt die Laufzeit für jeden Test ca. 3 h:14 min.

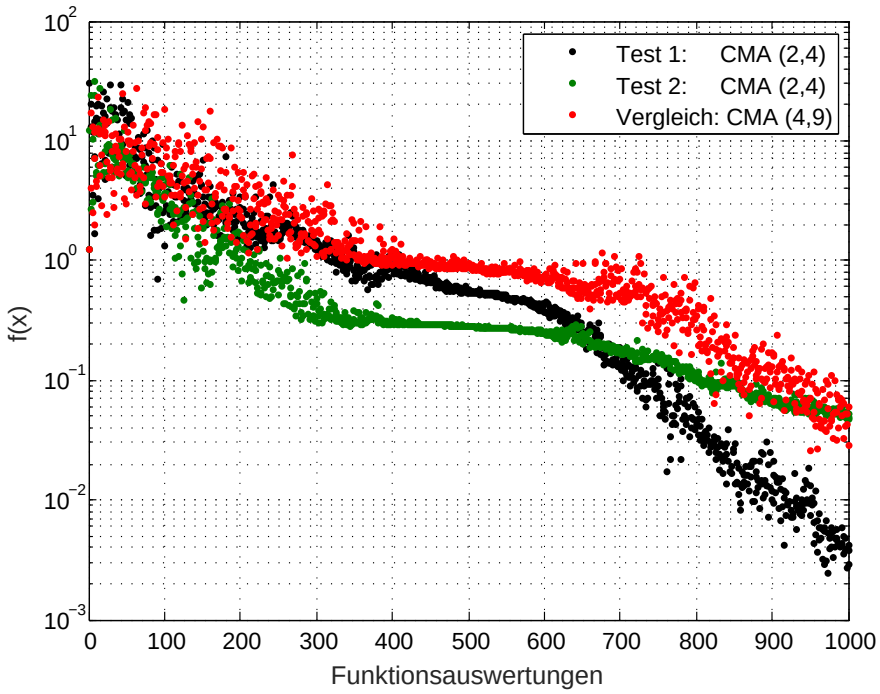


Abbildung 7.16: Konvergenz von CMA-ES für verschiedene Populationsgrößen

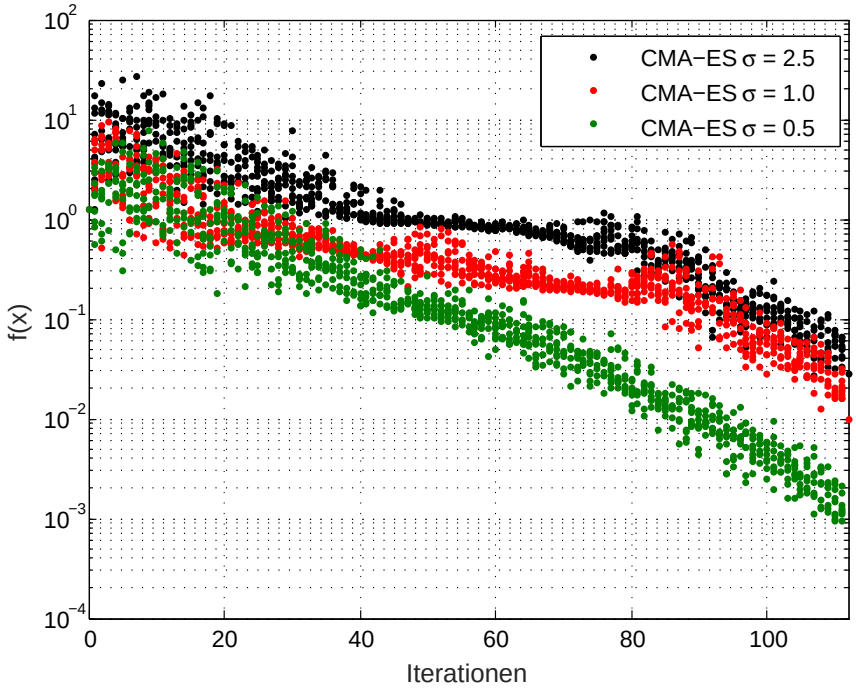


Abbildung 7.17: Konvergenz von CMA-ES für verschiedene Startschrittweiten σ

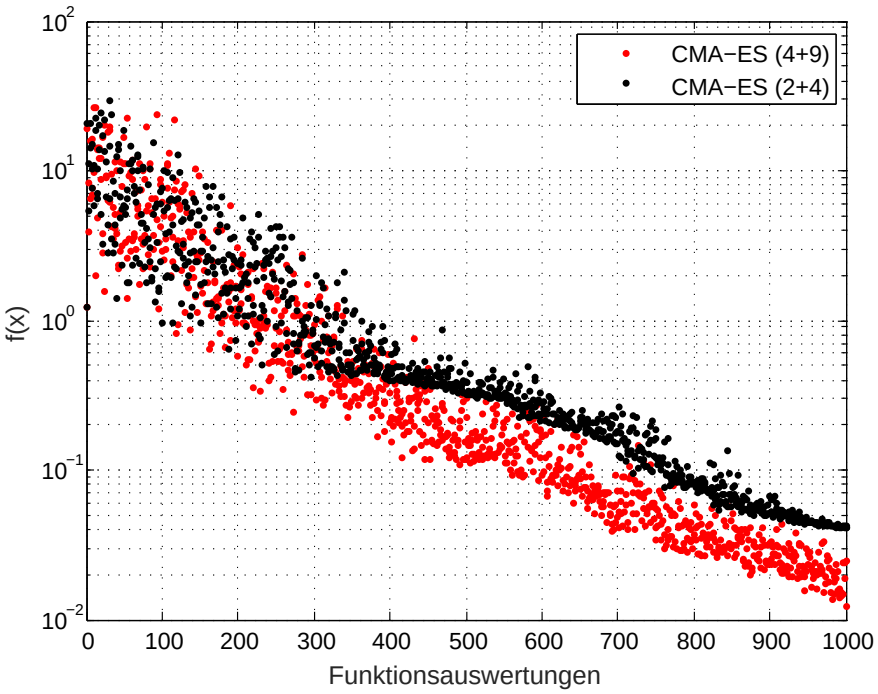


Abbildung 7.18: Konvergenz von CMA-ES für die +-Strategie

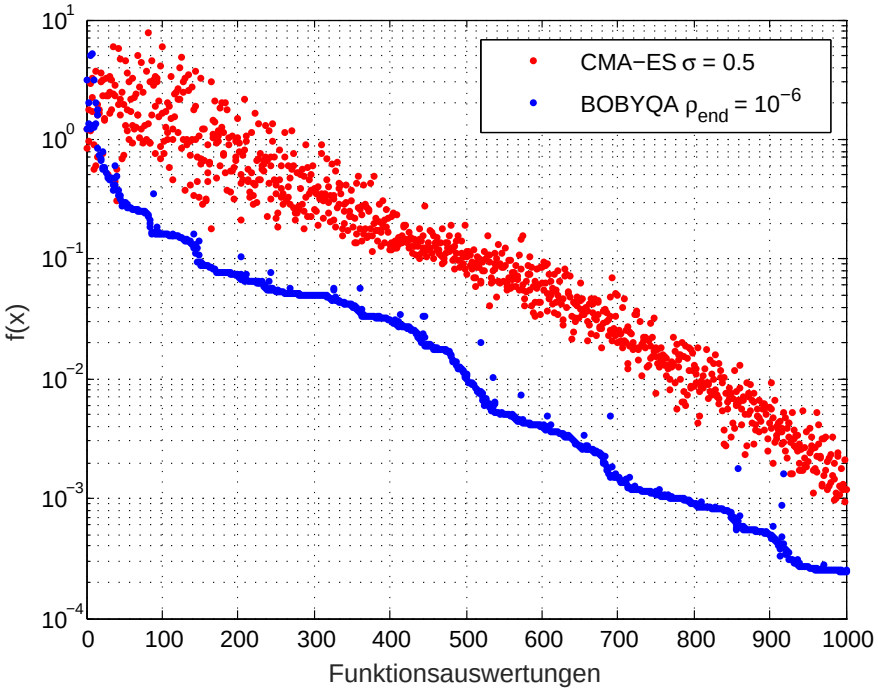


Abbildung 7.19: Konvergenz von BOBYQA und CMA-ES

Die Plus-Strategie aus Abschnitt 5.4 kommt nun zum Einsatz. Abbildung 7.18 zeigt die Resultate der Plus-Strategie für eine Populationsgröße von 4 und 9 Individuen. Es fällt auf, dass die größere Population mit 9 Individuen eine bessere Verteilung der Zielfunktionswerte zeigt. Tabelle A.12 im Anhang sind weitere Details der Optimierungsergebnisse zu entnehmen. Es wird eine Genauigkeit von $1.224 \cdot 10^{-3}$ für den Test mit einer Populationsgröße von 9 erzielt. Im Vergleich dazu wird für eine Population mit 4 Individuen eine Exaktheit von $\triangle \text{SDx} = 4.130 \cdot 10^{-2}$ erreicht. Trotz der stärkeren Fokussierung durch die Plus-Strategie gelingt es dem Optimierer nicht bessere Ergebnisse zu produzieren.

Zuletzt werden die besten Ergebnisse der beiden Algorithmen BOBYQA und CMA-ES verglichen. Abbildung 7.19 präsentiert die Resultate in einem Plot. Beide zeigen eine ähnliche Verlaufskurve. BOBYQA schafft es bereits nach 800 Funktionsauswertungen eine Genauigkeit von 10^{-3} zu erreichen, welche CMA-ES nach 1000 Iterationen erzielt. Es gelingt dem BOBYQA Algorithmus gleich zu Beginn eine gute Abstiegsrichtung zu finden, sodass der Zielfunktionswert bereits am Anfang des Optimierungslaufs stark verringert wird. Nach 150 Evaluationen werden Zielfunktionswerte kleiner als 0.1 erreicht. CMA-ES schafft es erst nach 450 Funktionsauswertungen in diesen Wertebereich zu gelangen. Während der ersten Iterationen erreicht der CMA-ES Algorithmus bessere Zielfunktionswerte als BOBYQA. Hier ist BOBYQA noch mit dem Aufstellen des ersten quadratischen Modells zugange. Dafür braucht dieser im sechsdimensionalen 13 Auswertungen. Und auch zwischen 14 und 50 Funktionsauswertungen erreicht der CMA-ES Algorithmus Zielfunktionswerte, die mit denen des BOBYQAs übereinstimmen. Jedoch schafft es CMA-ES nicht diese Abstiegsrichtung schnell genug zu verfolgen. BOBYQA erreicht einen Zielfunktionswert von $2.400 \cdot 10^{-4}$ mrad und CMA-ES eine Genauigkeit von $f(x_*) = 9.419 \cdot 10^{-4}$ mrad.

Diese Tests zum CMA-ES haben gezeigt, dass der Algorithmus durch die stochastische Vorgehensweise ein unvorhersehbares Konvergenzverhalten an den Tag legt. Aufgrund dessen ist der Algorithmus schwer einschätzbar. Auch die Optimierungsversuche haben keine eindeutige Verbesserung des Algorithmus gezeigt. Deshalb wird sich für weitere Tests auf den BOBYQA Algorithmus konzentriert, der durchweg sehr gute Lösungen liefert.

7.3 12-dimensionales Problem

Die Optimierung wird auf den 12-dimensionalen Fall erweitert. Das 12-dimensionale Testproblem wird in Abschnitt 6.1.3 vorgestellt. Hierzu werden Translationen in x-, y- und z-Richtung an allen vier Aufhängepunkten erlaubt, also $f : \mathbb{R}^{12} \to \mathbb{R}$, $(x_1, y_1, z_1, x_2, y_2, z_2, x_3, y_3, z_3, x_4, y_4, z_4)^T \mapsto f(x_1, \cdots, z_4)$.

7.3.1 Hybridalgorithmus

Der Hybridalgorithmus aus Abschnitt 5.5 kommt nun zum Einsatz. Die Optimierungsparameter sind durch $a_i = -5$ und $b_i = 5$ für $i = 1, .., 12$ begrenzt. Abbildung 7.20 zeigt die Konvergenz der BOBYQA Durchläufe mit den von CMA-ES berechneten Startwerten. Die Startwerte, welche von CMA-ES generiert werden und die von BOBYQA berechneten Lösungen sind in Tabelle A.14 im Anhang dargestellt. Bei der Betrachtung der von BOBYQA ermittelten Lösungsvektoren in Tabelle A.14 wird festgestellt, dass keiner der konstruierten Lösung $[1, 0, 1, 0, 0, 0, 0, 0, \frac{1}{2}, \frac{1}{2}, 0, 0]^T$ nahe kommt. Die erste Vermutung ist, dass der Suchraum viele Optima enthält und so nicht die konstruierte Lösung gefunden wird. Jedoch zeigt der Vergleich der Slope Deviation Plots in Tabelle A.15 kaum Unterschiede. Erst durch die Gegenüberstellung der Plots der Slope Deviation Differenzen von Optimierungs- und Referenzmodell in Tabelle A.16 werden kleine Unterschiede zwischen Optimierungs- und Referenzmodell erkennbar. In einigen Fällen sind die Bereiche um die Pads stärker verformt als für andere Testdurchläufe. Da der Spiegel an keiner Stelle fixiert

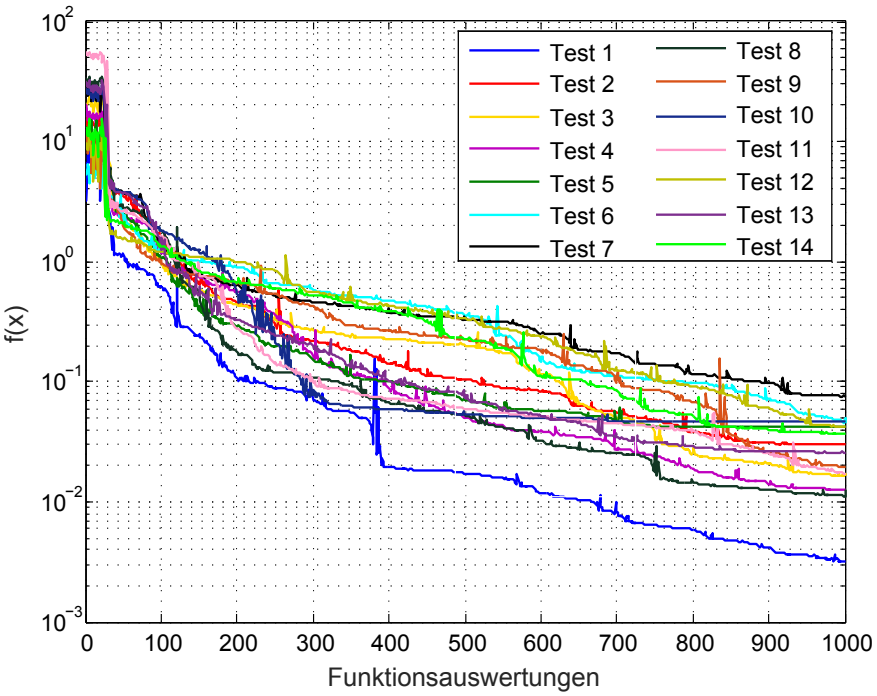


Abbildung 7.20: Konvergenz der Zwischenergebnisse des Hybridalgorithmus

wird, kann der Algorithmus den Referenzspiegel auf viele unterschiedliche Weisen nachbilden. Dies hängt, wie bereits in Abschnitt 4.1 beschrieben, damit zusammen, dass die Spiegelkrümmung bei einer Deflektometriemessung nur durch die Oberflächennormalenvektoren bestimmt werden. Translationen des Spiegels können zur gleichen Lösung führen. Aufgrund dessen ist zunächst kein Hybridalgorithmus nötig und die weiteren Optimierungen werden nur mit dem BOBYQA Verfahren durchgeführt.

Des Weiteren wird durch die Anwendung des Hybridalgorithmus indirekt eine Sensitivitätsanalyse bezüglich des Startvektors gemacht. Tabelle A.14 zeigt, dass der erste Startwert durch die BOBYQA Optimierung den besten Zielfunktionswert liefert. Dieser Startwert ist dem Nullvektor am nächsten. Auffällig ist auch Testlauf 7, welcher am schlechtesten abgeschneidet. Hier wird als Startwert für x_3 der Grenzwert $a_7 = -5$ generiert sowie für x_2 und x_4 Werte nah an den Optimierungsgrenzen. Während des Optimierungsprozesses wandern die Werte von den Intervallgrenzen weg. Auch bei Lauf 6, welcher den kleinsten Zielfunktionswert des CMA-ES-Startwertes aufweist, gelingt es BOBYQA nicht einen kleineren Zielfunktionswert für den Lösungsvektor im Vergleich zum ersten Durchlauf zu finden. Wahrscheinlich wird durch den nahe der Grenze gewählten Wert von $x_4 = 4$ eine nahe liegende Lösung ausgeschlossen, weshalb der Optimierer in einer anderen Richtung suchen muss. Aufgrund dessen wird als Startwert der Nullvektor und als Grenzwert für die maximalen Translationen und Rotationen $\pm 5\,\text{mm}$ bzw. $\pm 5\,\text{mrad}$.

7.3.2 Identische Zielfunktionswerte

Betrachtet werden die Slope Deviation Plots in x und y für die beiden Verschiebungsvektoren $\hat{x} = [0, 0, 1,\ 0, 0, 0,\ 0, 0, 0,\ 0, 0, 0]^T$ und $\tilde{x} = [0, 0, 0,\ 0, 0, -1,\ 0, 0, -1,\ 0, 0, -1]^T$ mit den Gravitationskomponenten $g = [0,\ 0,\ 9810]^T$. Es lässt sich mittels Abbildung 7.21 feststellen, dass, obwohl die Eingabewerte unterschiedlich sind, die gleichen Slope Deviation Plots in x sowie in y erzeugt werden. Die Abbildung 7.22 der dreidimensionalen Oberflächenbilder sind komplett

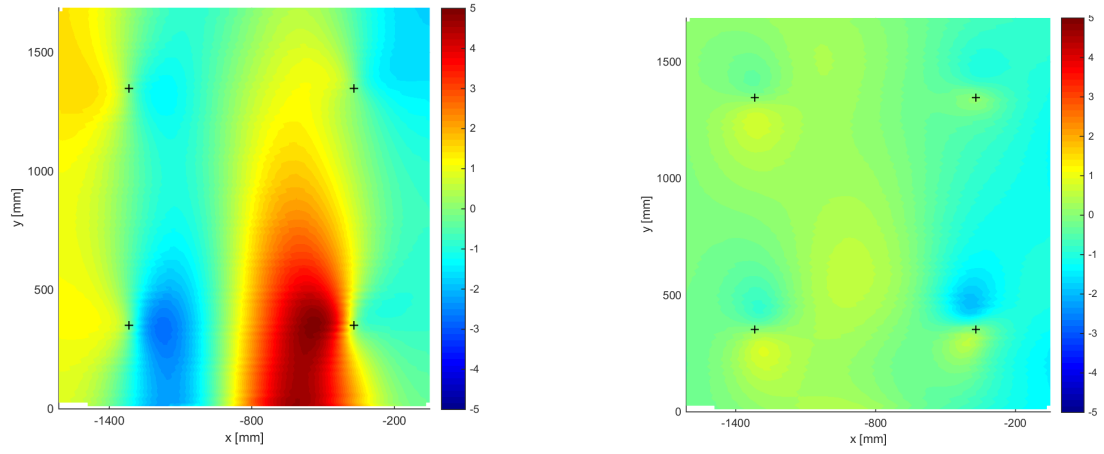


Abbildung 7.21: Plots der Slope Deviation in x (links) und y (rechts)

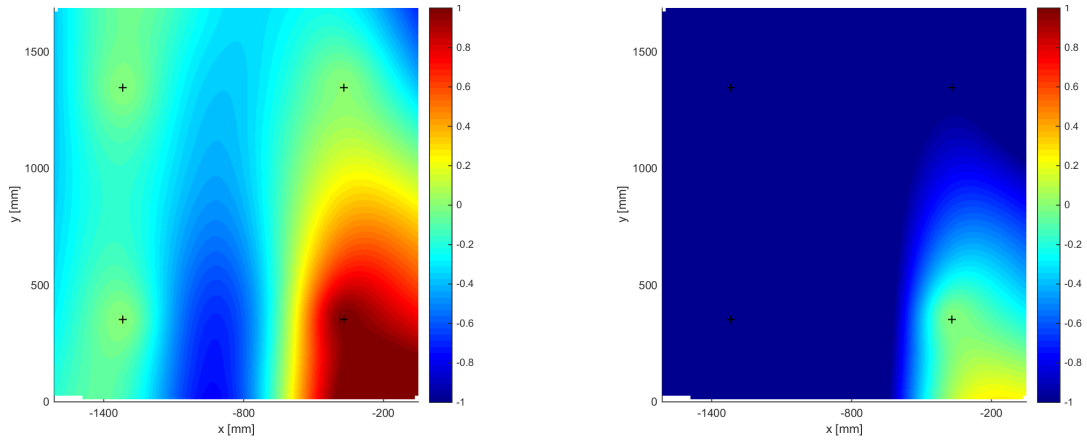


Abbildung 7.22: Plots der z-Deviation in $\hat{x}$ (links) und $\tilde{x}$ (rechts)

verschieden. Deutlich erkennbar ist, dass links Pad 1 hochgezogen wird, während beim Test auf der rechten Seite nur dieses Pad auf seiner Sollposition liegt. Folglich kann eine Wahl von unterschiedlichen Werten zu gleichen Slope Deviation Plots führen. Damit ist die Funktion $f : \mathbb{R}^n \to \mathbb{R}, (x_1, y_1, z_1, ...)^T \mapsto \triangle$SDx nicht injektiv. Da mittels Algorithmen vom RMS der Steigungsabweichungsdifferenzen $\triangle$SDx auf die ursprünglichen Verschiebungen und Verkippungen geschlossen wird, sind dadurch verschiedene Lösungsvektoren möglich und es handelt sich damit nur um translatierende Vektoren, die in einen Vektor mit gleichem RMS Wert umgerechnet werden können.

Damit liegt die Idee nahe, die z-Deviation in die Zielfunktion aufzunehmen. Durch diese zusätzliche Information könnte der Optimierungsprozess beschleunigt werden. Die Ergebnisse werden im nächsten Abschnitt 7.3.3 vorgestellt.

7.3.3 Optimierung der z-Deviation

Nun wird die Optimierung der z-Deviation mit dem 12-dimensionale Testproblem betrachtet. Dabei dienen die z-Werte der vier Aufhängepunkte als Optimierungsparameter. Damit gilt $f : \mathbb{R}^4 \to \mathbb{R}, (z_1, z_2, z_3, z_4)^T \longmapsto f(z_1, z_2, z_3, z_4)$, wie in Abschnitt 5.6.1 beschrieben. Der gesuchte Lösungsvektor ist durch $[1, 0, \frac{1}{2}, 0]^T$ gegeben. Für den BOBYQA Algorithmus gilt $\rho_{beg} = 1.5$, $\rho_{end} = 1 \cdot 10^{-10}$, als Startwert wird der Nullvektor gewählt und die maximale Anzahl an Funktionsauswertungen wird auf 125 gesetzt. Tabelle A.13 im Anhang listet die Ergebnisse des Optimierungslaufes auf. Es zeigt sich, dass die Optimierung bereits nach 105 Evaluationen mit einer Laufzeit von 22 min abbricht. Der gefundene Vektor ist $z_* = [0.890, -0.027, 1.888, -0.695]^T$ und der Optimierer bricht mit einer Genauigkeit von $8.125 \cdot 10^{-1}$ ab. Während die ersten beiden Komponenten der berechneten Lösung das konstruierte Ergebnis gut annähern, sind die z-Werte an Pad 3 und 4 ungenügend genau berechnet. Dies zeigt sich auch in den z-Deviation Plots in Abbildung 7.23. Auf der linken Seite ist der Plot der z-Deviation vom Optimierungsprozess und auf der rechten Seite der Referenzplot der z-Deviation abgebildet. Es ist deutlich zu erkennen,

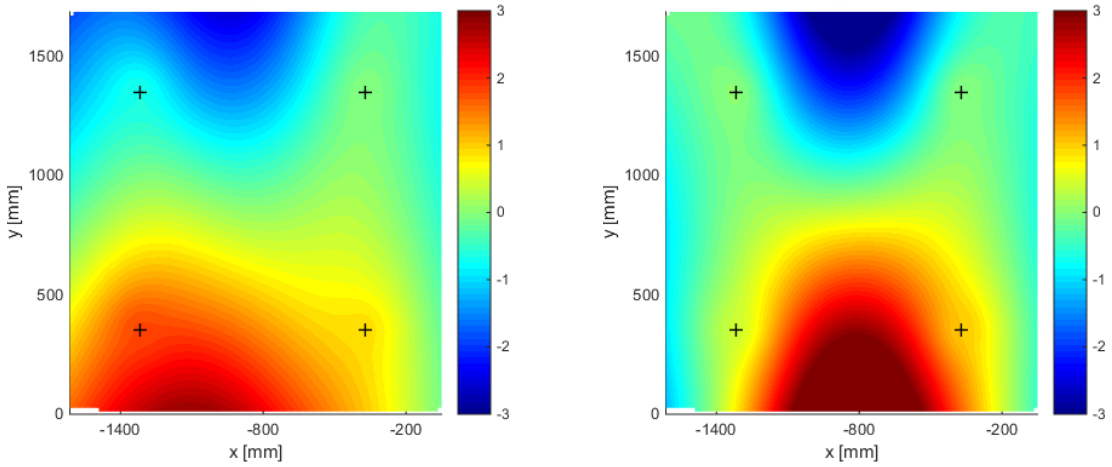


Abbildung 7.23: Plots der z-Deviation von Optimierungsmodell (links) und Referenz (rechts)

dass Pad 3 zu weit in $+z$-Richtung verschoben ist. Ferner findet sich oberhalb von Pad 4 eine Verformung der Spiegelfacette, welche nicht dem Referenzmodell gleicht. Die beiden Plots zeigen auf den ersten Blick eine ähnliche Verformung, jedoch auch deutliche Unterschiede. Die unterschiedlichen Lösungsvektoren sowie die verschiedenen z-Deviation Plots zeigen, dass der Algorithmus bei dieser Problemstellung in ein lokales Optimum läuft. Auch Abbildung 7.24, welche auf der linken Seite die Konvergenz mit der z-Deviation als Zielfunktion und rechts die zugehörige Funktionswerte der RMS-Werte der lokalen Steigungsabweichungsdifferenzen darstellt, verdeutlicht, dass bereits nach 25 Funktionsauswertungen keine Optimierung des Funktionswertes stattfindet. Auf der rechten Seite ist zu sehen, dass der kleinste Funktionswert in Bezug auf die Slope Deviation nach 10 Funktionsauswertungen gefunden wird und anschließend nur noch größere Werte berechnet werden. Dies ist ein weiteres Indiz, dass die Optimierung mit der z-Deviation als Zielfunktion nicht zum gewünschten Erfolg führt. Es ist davon auszugehen, dass der Lösungsraum viele lokale Optima aufweist und es wird von einer Einbeziehung der z-Deviation in den Optimierungsprozess abgeraten.

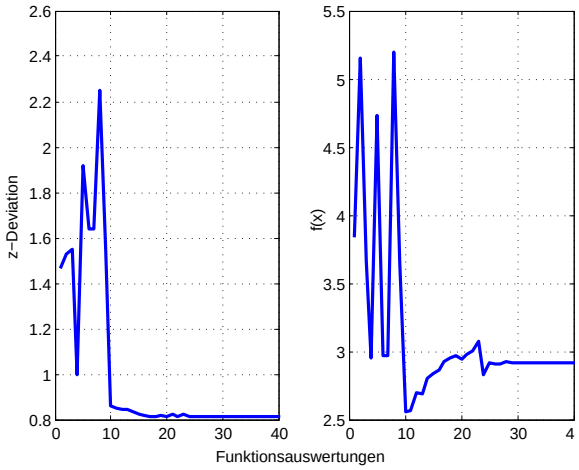


Abbildung 7.24: Konvergenz für die z-Deviation (links) und zugehöriger Slope Deviation Plot in x (rechts) Optimierung des 12-dimensionalen Testproblem

7.3.4 Parameterreduktion

Für die Optimierung wird die Klebstoffschicht Nummer 1 festgehalten und die anderen Aufhängungen optimiert sowie in einem zweiten Durchlauf die Klebstoffschicht am zweite Aufhängepunkt fixiert, während die Aufhängungen 1, 3 und 4 optimiert werden, wie in Abschnitt 5.7 beschrieben. Die maximale Anzahl an Funktionsauswertungen beträgt 1000 und es gilt für die Trust-Region Radien $\rho_{beg} = 1$ und $\rho_{end} = 10^{-6}$.

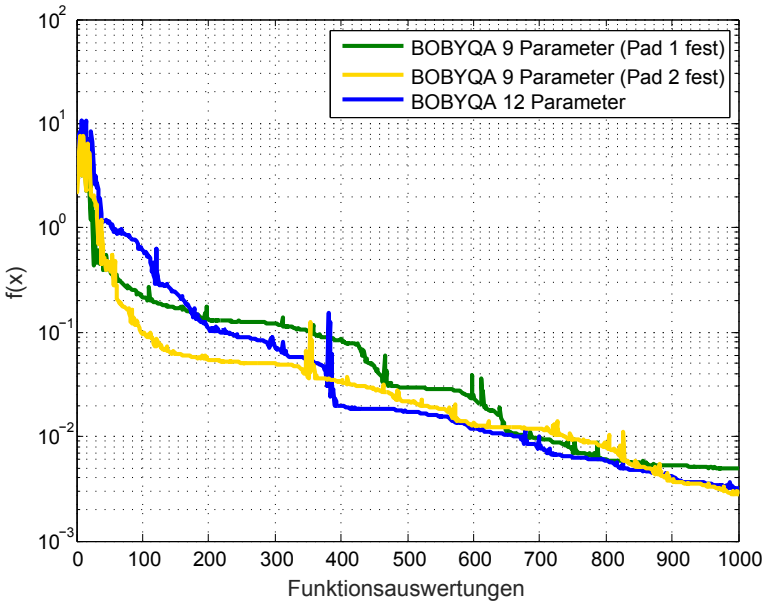


Abbildung 7.25: Konvergenz für das 12-dimensionale Testproblem mit unterschiedlicher Parameteranzahl

Die Ergebnisse der Optimierungsdurchläufe sind in Tabelle A.17 im Anhang aufgelistet sowie in Abbildung 7.25 dargestellt. Die Abbildung zeigt ein ähnliches Verhalten der drei Optimierungsläufe. Jeder Testlauf schafft es an einem bestimmten Punkt während der Optimierung den kleinsten Zielfunktionswert zu erreichen. Die Optimierung mit festem Aufhängepunkt 1 zeigt ein leicht schlechteres, jedoch vergleichbares Verhalten. Bei der Betrachtung von Tabelle A.17 fällt auf, dass alle Durchläufe mit 13 h:17 min nahe zu die gleiche Zeit benötigen. Naheliegend wäre, dass die Reduktion der Optimierungsparameter eine schnellere Funktionsauswertung zur Folge hätte. Obwohl ein Pad festgehalten wird und damit die Anzahl an Parametern um 3 reduziert wird, ist die Laufzeit bei allen Testläufen nahezu identisch. Die Parameter der festgehaltenen Aufhängung gehen zwar nicht in den Optimierungsprozess ein, jedoch findet keine Änderung des FE-Modells statt. Da dieses ausschlaggebend für die Laufzeit ist findet hier keine Beschleunigung dieser statt.

7.4 20-dimensionales Problem

Es werden nun Rotationen mit in das Klebstoff-Optimierungsmodell aufgenommen. Zunächst werden nur Verkippungen um die x-, sowie die y-Achse in den Optimierungsprozess einbezogen, sodass für die Zielfunktion $f : \mathbb{R}^{20} \to \mathbb{R}$ gilt. Das heißt die Variablen sind $x_i, y_i, z_i, \alpha_i, \beta_i$ für $i = 1, 2, 3, 4$. Als Referenz dient das 20-dimensionale Testproblem aus Abschnitt 6.1.4. Es findet die Anwendung der Coordinate-Descent Methode statt. Ferner wird eine weitere Reduktion der Optimierungsparameter durchgeführt.

7.4.1 Coordinate-Descent Methode

Die Coordinate-Descent Methode aus Abschnitt 5.9 wird auf das Testproblem angewendet. Es werden die folgenden Eingabedaten für den Algorithmus gewählt $\rho_{beg} = 1$, $\rho_{end} = 10^{-5}$, $a_i = -4$, $b_i = 4$, x_0 ist durch den Nullvektor gegeben und die maximale Anzahl an Funktionsauswertungen ist 200. Abbildung A.1 zeigt den ersten Versuch der Coordinate-Descent Optimierung. Es fällt das sprunghafte Verhalten der Zielfunktionswerte auf. Es zeigt sich, dass die Rotationswerte deutlich größere Funktionswerte als die Translationsparameter erreichen. Dies liegt daran, dass die Berechnung der Rotationsparameter in der Einheit rad erfolgt. Ein Korrekturfaktor, wie in 5.8 beschrieben, wird eingeführt, sodass die Rotationsparameter in mrad berechnet werden. Es erfolgte ein erneuter Test mit der Coordinate-Descent Methode. Für jede Coordinate-Descent Teiloptimierung werden 25 Funktionsauswertungen erlaubt. Des Weiteren findet zum Vergleich ein Optimierungslauf mit allen 20 Parametern statt. Die Ergebnisse der einzelnen Teiloptimierungen sind in Tabelle A.19 im Anhang sowie für die Optimierung ohne die Coordinate-Descent Methode in Tabelle A.18 aufgelistet. Der Konvergenzplot der beiden Optimierungen zeigt Abbildung 7.26. Es fällt auf, dass die Zielfunktionswerte nach jeweils 25 Auswertungen zunächst wieder größer werden und die Zielfunktionswerte insgesamt sehr sprunghaft sind. Dies liegt daran, dass beim Coordiante Descent Verfahren für jedes Parameterset an einem Pad zunächst das erste quadratische Modell aufgestellt wird. Erst anschließend wird mit der Optimierung begonnen. Innerhalb der ersten Iterationen zeigt das Coordinate-Descent Verfahren deutlich kleinere Zielfunktionswerte als bei der Optimierung des einfachen BOBYQA Algorithmus. Dies liegt daran, dass im zweiten Fall des Optimierungsprozesses mit 20 freien Parametern zunächst das erste quadratische Modell aufgestellt wird, indem 40 Funktionsauswertungen um den Startwert erfolgen. Im Bereich zwischen 120 und 150 Funktionsauswertungen sind beide Methoden gleichauf. Dann schafft es BOBYQA leicht bessere Zielfunktionswerte zu finden. Während der letzten Iterationen sind die beiden Durchläufe mit einer Genauigkeit von $8.013 \cdot 10^{-2}$ mrad und $8.593 \cdot 10^{-2}$ für den Vergleichstest fast gleich auf.

Da BOBYQA beim Coordinate-Descent Verfahren beim Wechsel der Pads jeweils das erste quadratische Modell neu aufstellt, werden im Laufe einer Optimierung viele Funktionsauswertungen dafür genutzt. Je mehr Iterationen gemacht werden, desto häufiger müsssen Funktionsauswertungen für das erneute Aufstellen des ersten quadratischen Modells genutzt werden. Aufgrund

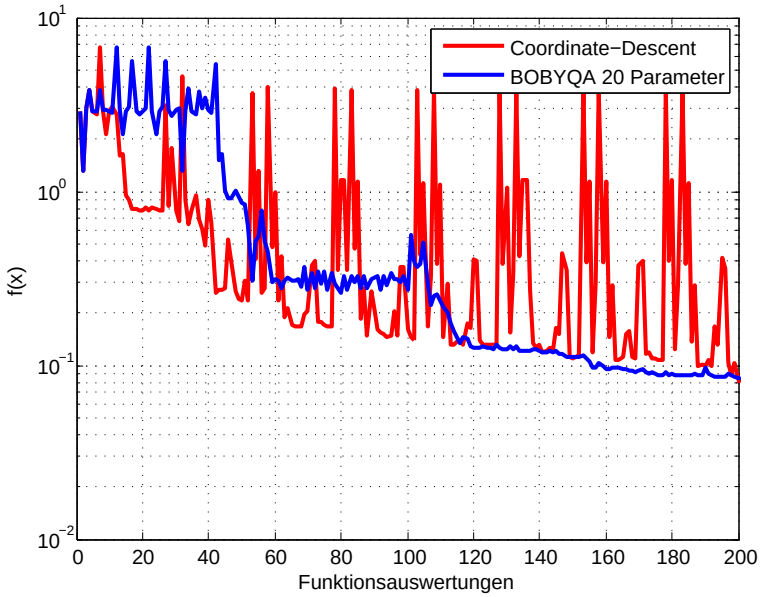


Abbildung 7.26: Konvergenz für das Coordinate-Descent Verfahren und Optimierungslauf mit 20 Parametern

der teuren Auswerteprozedur ist die Kombination von Coordinate-Descent Verfahren und BOBYQA Algorithmus nicht die beste Kombination. Dennoch kann das Verfahren durch weitere Optimierungsversuche eine gute Alternative bieten. Statt einen zyklischen Durchlauf des Verfahrens zu wählen, können die Bereiche der Spiegelfacette um die Pads betrachtet werden. Der Optimierer kann durch das Auswerten dieser Bereiche zu dem Montagepad mit dem größten Optimierungsbedarf springen. Das heißt, das Pad, in dessen Umgebung die größten RMS-Werte der Steigungsabweichungsdifferenzen sind, wird als nächstes optimiert. Auch eine Anpassung des Start Trust-Region bei jedem Subdurchlauf dürfte den Optimierungsprozess beschleunigen.

7.4.2 Parameterreduktion

Bei diesem Parameterreduktionsversuch ist zu beachten, dass nicht ein komplettes Pad festgehalten werden kann. Werden alle Parameter eines Montagepads als fest definiert, so ist es nicht mehr möglich die Slope Deviation von möglichen Verkippungen an diesem Pad nachzubilden. Aufgrund dessen werden die Rotationsvariabeln in die Optimierung integriert. Damit ergibt sich ein 17-dimensionales Problem mit α_1, β_1 und $x_i, y_i, z_i, \alpha_i, \beta_i$ für $i = 2, 3, 4$. Die Eingabedaten stimmen mit denen aus dem vorherigen Abschnitt 7.4.1 überein.

Abbildung 7.27 zeigt die Konvergenz mit 20 und 17 Optimierungsparametern und in Tabelle A.18 im Anhang sind die Lösungswerte, Anzahl an Funktionsauswertungen und die Laufzeit aufgelistet. Deutlich zu erkennen ist, dass der Lauf mit 20 Parametern besser abschneidet als die Optimierung mit 17 Variablen, obwohl die Anzahl der Optimierungsparameter verringert ist. Jedoch weist die Optimierung mit 17 Parametern ein insgesamt glatteres Abstiegsverhalten

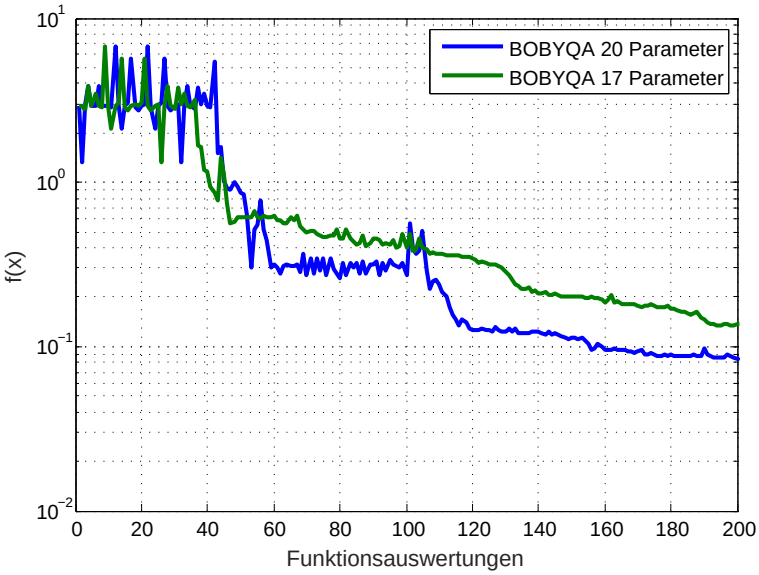


Abbildung 7.27: Konvergenz für das 20-dimensionale Testproblem mit unterschiedlicher Parameteranzahl

auf. Nach 200 Funktionsauswertungen brechen die Algorithmen mit einer Genauigkeit von $f(x_*) = \triangle\text{SDx} = 8.593 \cdot 10^{-2}$ mrad für 20 Parameter bzw. $1.386 \cdot 10^{-1}$ mrad für 17 Unbekannte. In beiden Fällen beträgt die Laufzeit ca. 1 h. Der Grund für das bessere Abschneiden der Optimierung mit 20 Parametern könnte daran liegen, dass durch die Fixierung des Pads 1, eine naheliegende Lösung ausgeschlossen wird, weshalb die Optimierung hier langsamer verläuft als für den Fall von 20 Optimierungsparametern.

7.5 Laborunterstruktur

Das FE-Modell mit Laborunterstruktur wird in Abbschnitt 4.2.2.2 und das zugehörige Testproblem in 6.1.5 vorgestellt. Als Optimierungsmodell wird auf das Padmodell zurückgegriffen. Nun werden die Optimierungsresultate präsentiert.

7.5.1 Rotationsuntersuchung

Es werden nur die Rotationen entlang der x-, y- und z-Achse an allen vier Montagepads betrachtet. Translationen fließen nicht in den Optimierungsprozess ein, sondern werden auf null gesetzt. Dadurch entsteht ein 12-dimensionales Optimierungsproblem mit den Variablen $\alpha_i, \beta_i, \gamma_i$, für $i = 1, 2, 3, 4$. Als Startwert wird $x_0 = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]^T$ gewählt. Es werden 150 Evaluationen erlaubt und für die Trust Region Radien gilt $\rho_{beg} = 10$ und $\rho_{end} = 10^{-10}$. Es ist zu beachten, dass die Berechnung in der Einheit mrad erfolgt. Des Weiteren stellt sich die Frage, ob

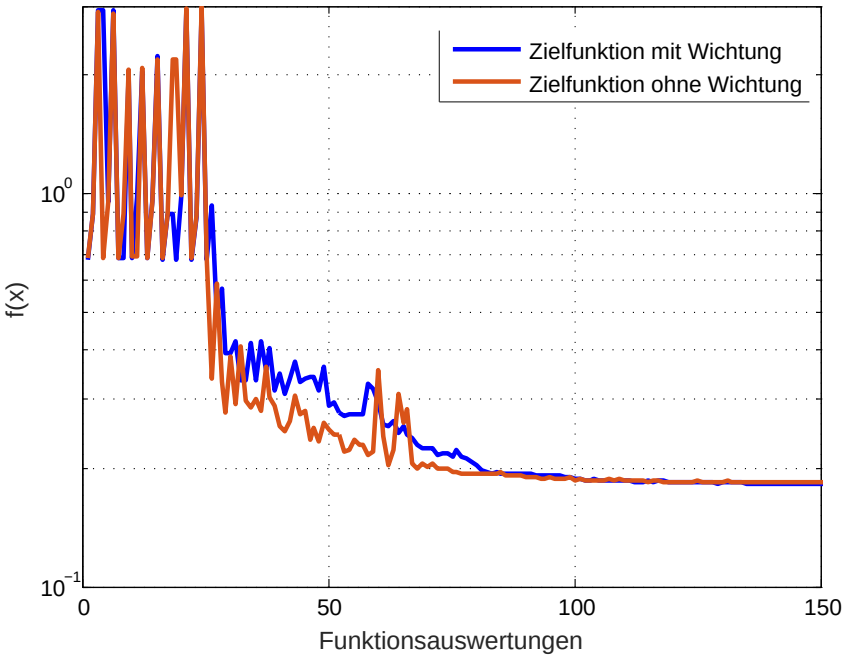


Abbildung 7.28: Konvergenz mit und ohne Wichtung der Zielfunktion

der Wichtungsterm $\frac{ak}{A_{tot}}$ in der Zielfunktion (5.3) einen Einfluss auf den Optimierungsprozess hat. Aufgrund dessen werden zwei Versuchsdurchläufe mit und ohne Wichtung der Zielfunktion getestet. Abbildung 7.28 zeigt die Konvergenz der beiden Versionen. Das sprunghafte Verhalten zu Beginn, zeigt die Auswertung der $2n + 1 = 25$ Interpolationspunkte. Zwischen 25 und 60 Funktionsauswertungen weist der Test ohne Wichtungsfaktor ein besseres Verhalten auf, danach sind die Kurven nahezu identisch. Abbildung 7.29 zeigt die Plots der lokalen Slope Deviation Differenzen in x, wobei auf der linken Seite die Resultate für die gewichtete und rechts für die ungewichtete Zielfunktion aufgezeigt sind. Für den gewichteten Fall wird eine Genauigkeit von 0.184 mrad erreicht, während die ungewichtete Zielfunktion eine Exaktheit von $\triangle$SDx = 0.184 mrad erlangt. Es ist kein Unterschied zwischen beiden Plots erkennbar. In beiden Fällen ist die Position der vier Aufhängepunkte nicht ideal. Dies kann daran liegen, dass die Translationen unberücksichtigt geblieben sind und auf null gesetzt wurden. Statt die Verschiebungen auf null zu setzten, führt die Wahl der Werte aus Tabelle 6.1 für die Translationskomponenten, welche mittels Simulationssoftware Ansys bestimmt wurden, zu besseren Zielfunktionswerten und folglich zu einem optimaleren Ergebnis.

Da die neue Zielfunktion weder eine Verbesserung in der Genauigkeit der Lösung noch eine Optimierung in der Geschwindigkeit bringt, wird die alte gewichtete Zielfunktion beibehalten. Des Weiteren wird festgestellt, dass sich die Wichtungsterme nur minimal unterscheiden. Da sie das Flächendreieck repräsentieren, auf welchem die Oberflächennormalenvektoren stehen, erscheint es sinnvoll diese Information in der Zielfunktion zu bewahren.

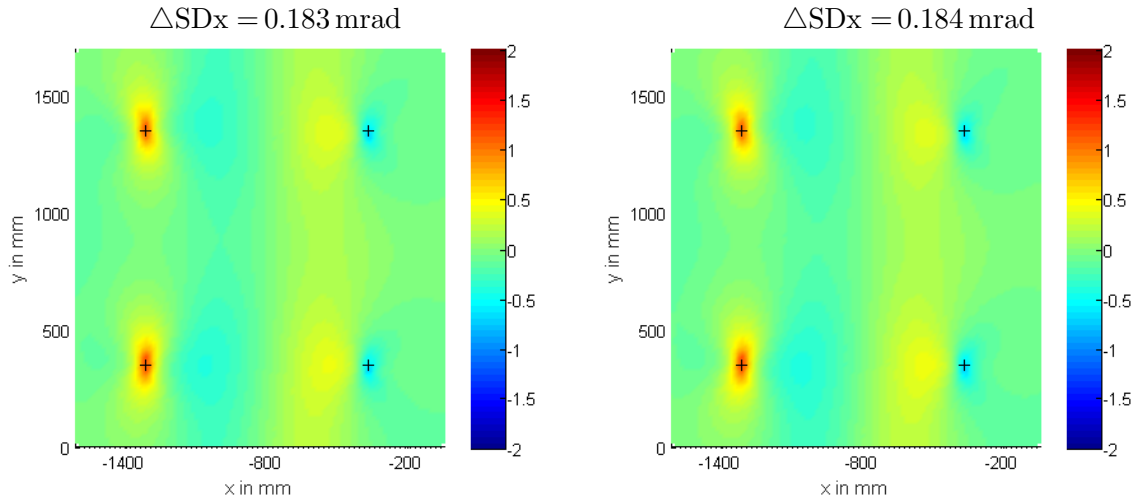


Abbildung 7.29: Plot der lokalen Differenzen der Slope Deviation in x mit gewichteter (links) und ungewichteter (rechts) Zielfunktion (in mrad)

7.5.2 Translation und Rotation

Zunächst wird Pad 1 für Translationen fixiert. Alle anderen Parameter werden für die Optimierung zugelassen. Damit gilt für die Zielfunktion $f : \mathbb{R}^{21} \to \mathbb{R}$.

7.5.2.1 Skalierung der Rotationskomponenten

Wie in 5.8 aufgeführt, verwendet der ANSYS-Solver für die Rotationsparameter immer die Einheit rad. Es finden Optimierungstests mit den gleichen Eingabedaten statt, jedoch erfolgt die Übergabe der Rotationsparameter beim ersten Durchlauf in rad und beim zweiten Test in mrad. Es gilt $x_0 = 0_{\mathbb{R}^{21}}$, $a_i = -5$, $b_i = 5$, $\rho_{beg} = 1$, $\rho_{end} = 10^{-10}$ und die maximale Anazahl an Funktionsauswertungen wird auf 1000 gesetzt. Das Ergebnis der Gegenüberstellung der Maßeinheiten ist in Abbildung 7.30 dargestellt. Auffällig ist die deutlich schnellere Konvergenz für den mrad-Fall. Für den rad-Fall wird eine Genauigkeit von $\triangle$SDx = 0.110 mrad erzielt und die mrad-Optimierung schafft eine Genauigkeit von 0.017 mrad. Des Weiteren zeigt sich beim rad-Modell, dass die Zielfunktionswerte beim Aufstellen des ersten quadratischen Modells Werte größer als 100 erreicht. Dies weist auf eine prägnante und unrealistische Verformung des Spiegels hin. Abbildung 7.32 zeigt die zugehörigen Plots der lokalen Differenzen der Slope Deviation in x. Hier wird nochmals der drastische Unterschied deutlich, welchen der Einheitenwechsel bewirkt. Während der Plot des mrad-Optimierungsdurchlaufs eine fast gleichmäßig grüne Fläche zeigt, sind im rad-Fall stark rote und blaue Bereiche sichtbar, welche auf Verformungen der Spiegelfacette weisen. Aufgrund dessen wird die Einheit mrad für die Translationen in die Optimierung integriert.

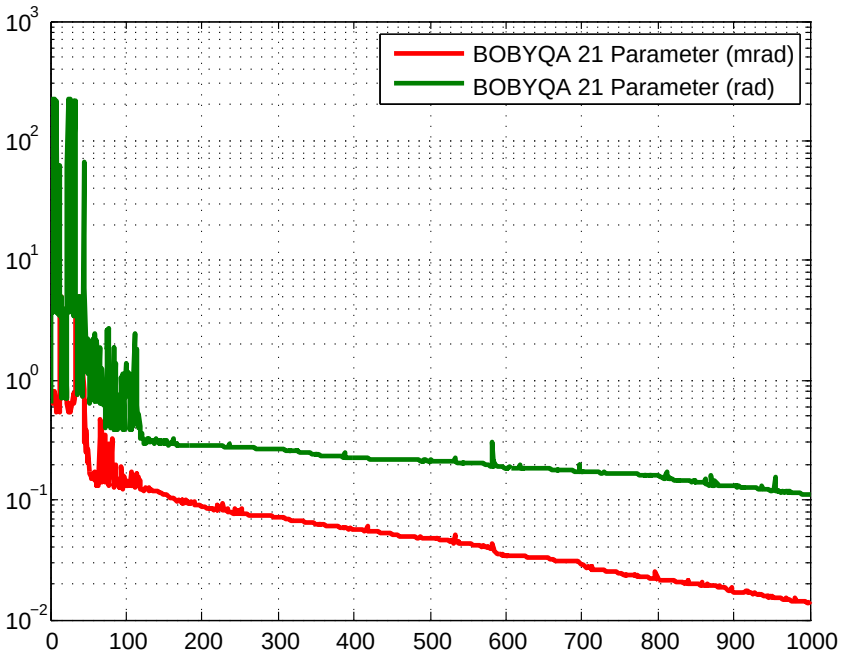


Abbildung 7.30: Konvergenz für Rotationsparameter rad und mrad

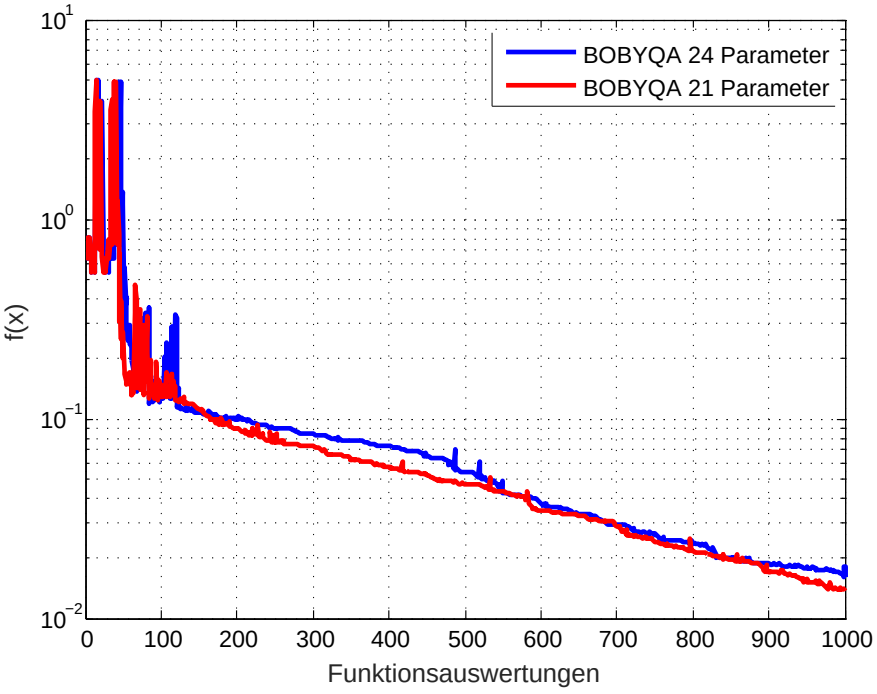


Abbildung 7.31: Konvergenz für verschiedene Wahl von Optimierungsparameter

7.5.2.2 Anzahl an Optimierungsparameter

Es erfolgt ein Vergleich zwischen der Optimierung mit 21 und 24 Parametern. Für die 21 Parameteroptimierung wird das erste Montagepad für Translationen fixiert und damit werden diese drei Verschiebungsparameter nicht als Optimierungsvariablen betrachtet. Im Fall mit 24 Parametern sind alle 24 Variablen für den Optimierungsprozess gegeben. Die Eingabedaten entsprechen denen aus dem vorherigen Abschnitt 7.5.2.1. Für die Optimierung mit 24 Variablen wird $x_0 = 0_{\mathbb{R}^{24}}$ gewählt.

Die Plots der lokalen Differenzen der Slope Deviation in x für die Durchlauf mit 21 und 24 Parametern sind in Abbildung 7.32 dargestellt. In beiden Fällen bildet das Optimierungsmodell die Referenz gut nach. Während bei der 21-Parameteroptimierung die stärksten Verformungen um Pad 3 herrschen, ist bei dem Testlauf mit 24 Variablen der Bereich um Montagepad 2 am stärksten deformiert. Die Konvergenzplots in Abbildung 7.31 zeigen ein gleiches Abstiegverhalten. Die erreichte Genauigkeit liegt im Fall mit 21 Variablen bei $f(x_*) = \triangle\text{SDx} = 0.017\,\text{mrad}$ und für die 24 Parameteroptimierung bei einem Wert von 0.016 mrad. Damit zeigt der Test mit 21 Variablen ein leicht besseres Verhalten als für 24 Parameter. Es ist jedoch zu bemerken, dass sich an der Laufzeit kein großer Unterschied feststellen lässt. Da die FE-Auswertung der Spiegelfacette den größten Zeitfaktor für das vorliegende Problem ausmacht. Auf der einen Seite kann die

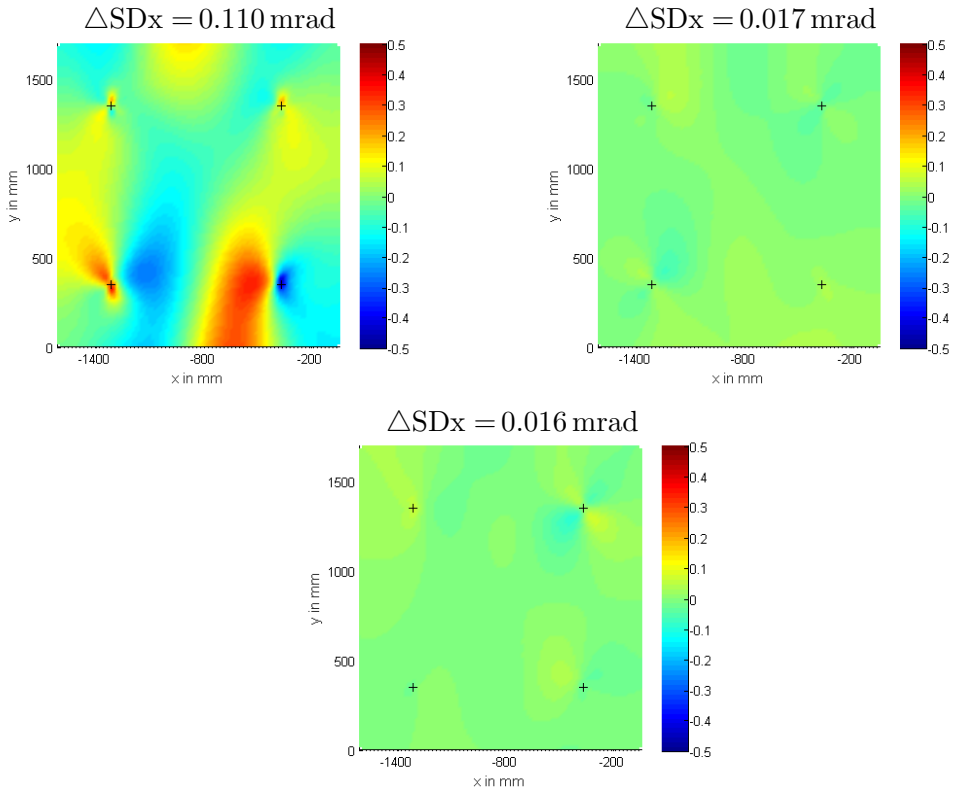


Abbildung 7.32: Plot der lokalen Differenzen der Slope Deviation in x von der Optimierung in rad (links oben), Optimierung mit 21 Parametern (rechts oben) und mit 24 Parametern (unten)

Verringerung der Parameteranzahl den Optimierungsprozess beschleunigen, da der Optimierer mit weniger Variablen arbeiten muss. Auf der anderen Seite könnte ein naheliegendes Optimum durch das Festhalten eines Pads ausgeschlossen werden. Deshalb bleibt es dem Anwender überlassen die Anzahl an Optimierungsparametern zu wählen.

7.5.2.3 Sensitivitätsanalyse bezüglich des finalen Trust-Region Radius

Die Wahl der Genauigkeit mit der gerechnet wird beziehungsweise mit der das Modell abbricht, einen entscheidenden Einfluss auf den Optimierungsprozess. Aus diesem Grund werden die finalen Trust-Region Radien $\rho_{end} = 1 \cdot 10^{-6}$ und $\rho_{end} = 1 \cdot 10^{-10}$ für das Testproblem der Laborunterstruktur untersucht. Es gilt $x_0 = 0_{\mathbb{R}^{24}}$, $a_i = -5$, $b_i = 5$, $\rho_{beg} = 1$ und die maximale Anazahl an Funktionsauswertungen wird auf 800 gesetzt.

Der zugehörige Plot der Konvergenz beider Durchläufe ist in Abbildung 7.33 aufgezeigt. Anfangs schafft es der Test mit $\rho_{end} = 10^{-10}$ kleinere Zielfunktionswerte zu erreichen, aber nach 300 Evaluationen zeigen beide Algorithmen denselben Verlauf. Die Durchläufe zeigen kaum Unterschiede bei der Wahl der beiden Radien. Während der Fall mit $\rho_{end} = 10^{-10}$ eine Exaktheit von $\triangle$SDx $= 0.024$ mrad erreicht, schafft es die $\rho_{end} = 10^{-6}$ Variante eine Genauigkeit von 0.023 mrad zu erzielen. Da ein kleinerer finaler Trust-Region hier jedoch keine Nachteile bringt und eine höhere Genauigkeit erreichbar ist, wird für weitere Tests der Trust-Region $\rho_{end} = 1 \cdot 10^{-10}$ empfohlen.

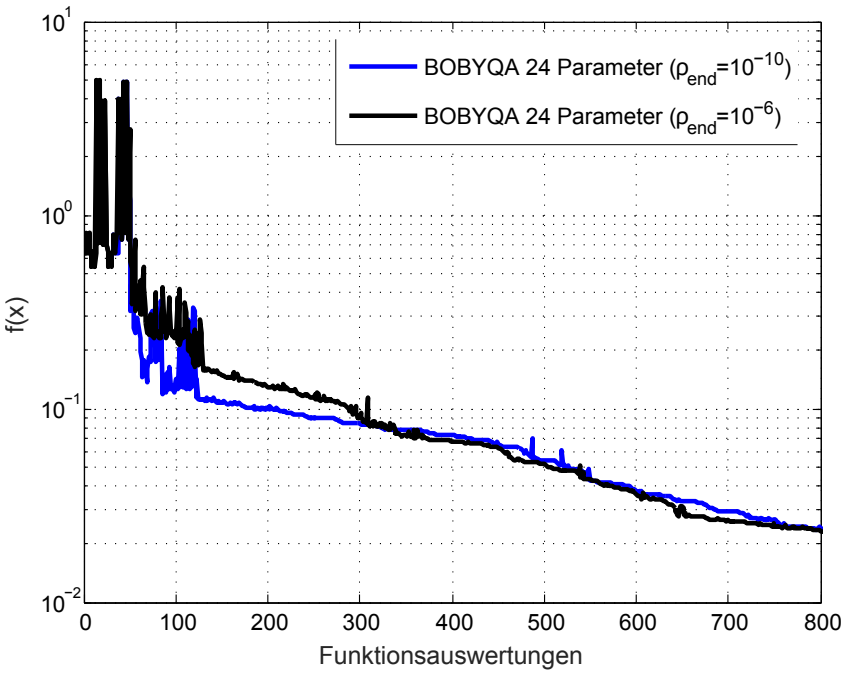


Abbildung 7.33: Konvergenz für unterschiedliche finale Trust-Region-Radien

7.6 Reale Messung

Als letztes wird der Algorithmus BOBYQA auf ein Ergebnis einer Deflektometriemessung aus Abschnitt 6.2 angewendet. Der Test findet mit 21 Optimierungsparametern statt. Da der Spiegel in der 0 °-Zenitstellung vermessen wurde, gilt $g = [0, 0, 9810]^T$. Als Startwert wird der Nullvektor gewählt, $\rho_{beg} = 1$, $\rho_{end} = 10^{-10}$, $a_i = -5$, $b_i = 5$ für $i = 1, ..., 21$ und die Anzahl an Funktionsauswertungen ist durch 1000 begrenzt. Tabelle 7.4 zeigt die Lösung der Optimierung. Es sind die Translationen und Rotationen der Pads aufgelistet. Durch das Festhalten des ersten Montagepads für Translationen sind diese Parameter unverändert geblieben. Die zugehörigen

Tabelle 7.4: Translationen und Rotationen der Pads

	Pad 1	Pad 2	Pad 3	Pad 4
x [mm]	0	0.174	−0.022	0.218
y [mm]	0	−0.108	0.201	−0.190
z [mm]	0	1.787	1.530	−0.821
α_x [mrad]	0.367	0.620	−2.275	−1.734
α_y [mrad]	1.432	1.711	1.498	−3.070
α_z [mrad]	0.091	−1.334	0.499	−0.023

Slope Deviation Plots sind in Abbildung 7.34 dargestellt. Hier ist auf der linken Seite das reine Optimierungsmodell abgebildet und rechts der Plot, welche die herstellungsbedingten Fehler der Spiegelfacette aufweist. Des Weiteren zeigt Tabelle 7.5 die Slope Deviation Plots in x (oben) und in y (unten) für die Lösung des Optimierers auf der linken Seite und die Referenz, also die Plots des vermessenen Spiegels auf der rechten Seite. Damit stellt der Plot oben links die Addition von reinem Optimierungsmodell und herstellungsbedingten Fehler dar. Es ist zu erkennen, dass der Optimierer die gemessene Spiegelfacette gut nachahmt. Auch die Slope Deviation in y wird

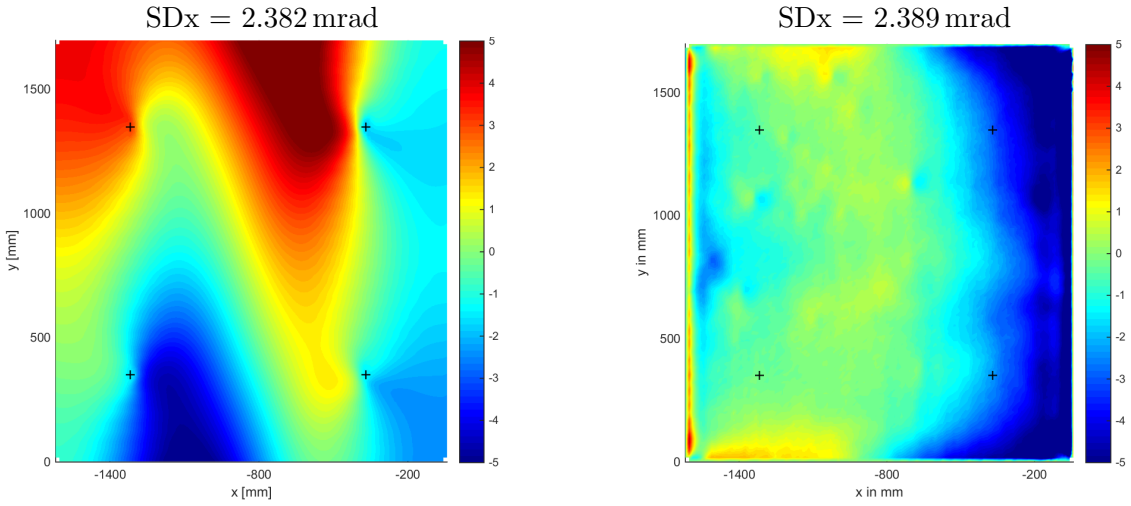
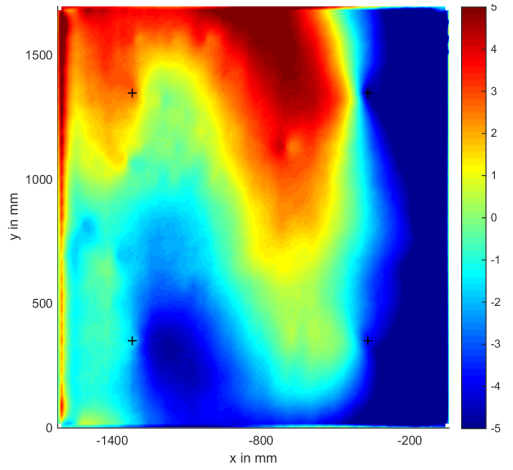
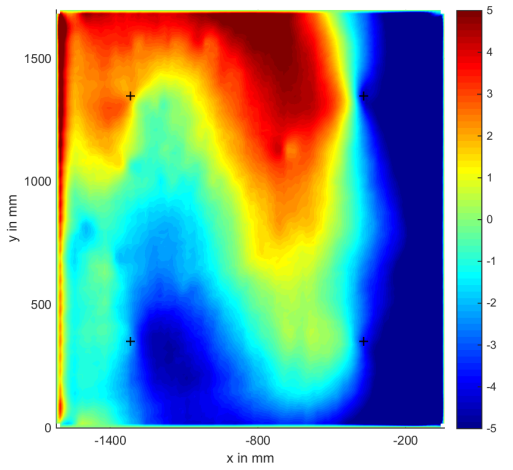
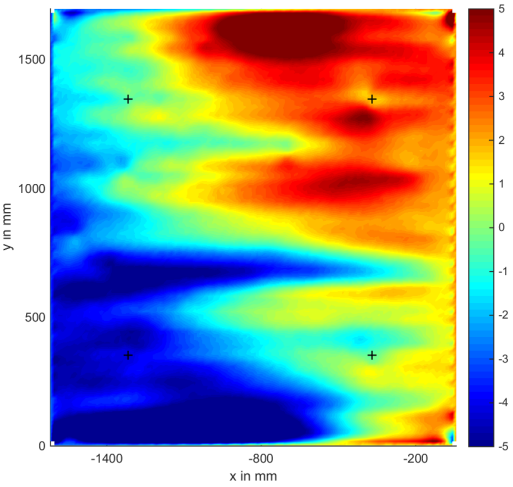
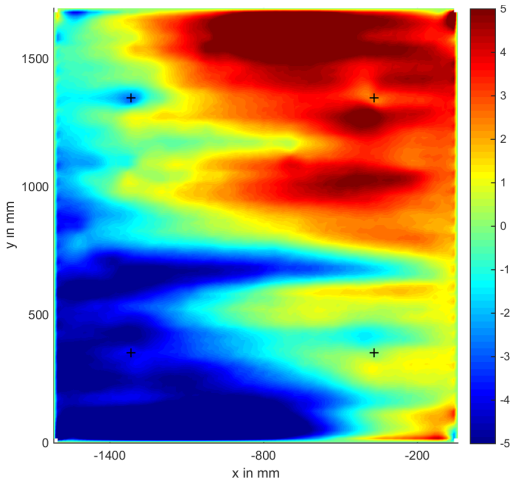


Abbildung 7.34: Slope Deviation Plots des Optimierungsmodells (links) und herstellungsbedingte Formabweichungen (rechts)

Tabelle 7.5: Slope Deviation Plots des Optimierungsmodells (links) und der Messung (rechts)

	Optimierungsergebnis	Referenz
	SDx = 3.546 mrad	SDx = 3.564 mrad
Slope Deviation Plot in x		
Slope Deviation Plot in y		

gut nachgebildet, obwohl diese nicht Teil der Optimierung ist. Hier sind deutlicher Unterschiede zwischen Optimierungsmodell und Referenz zu erkennen. Bei der Deflektometriemessung weist der rechte, obere Spiegelteil höhere Steigungsabweichungen auf als vom Optimierer berechnet. Auch um Pad 4 sind Abweichungen zwischen der Lösung von BOBYQA und den gemessenen Werten sichtbar. Um die jeweiligen Slope Deviations besser miteinander vergleichen zu können und um Unterschiede zwischen der Lösung des Optimierers und der Messung herauszustellen, wird in Abbildung 7.35 ein Plot der Differenzen gezeigt. Bei der Betrachtung des Plots der lokalen Steigungsabweichungsdifferenzen in x in Abbildung 7.35 stellt sich heraus, dass das Optimierungsmodell die Messung sehr gut nachbildet. Am Rand des Plots unterscheiden sich die Modelle am stärksten. Die Randeffekte können jedoch vernachlässigt werden. Auffällige Unterschiede sind auch um Pad 4 erkennbar. Hier stimmen die beiden Modelle noch nicht gut genug überein. Der Slope Deviation Plot in y zeigt größere Differenzen zwischen simulierter und gemessener Spiegelfacette. Vor allem der obere Bereich unterscheidet sich. Wiederholt zeigt sich, dass der Optimierer Pad 4 nicht an die richtige Position gesetzt hat.

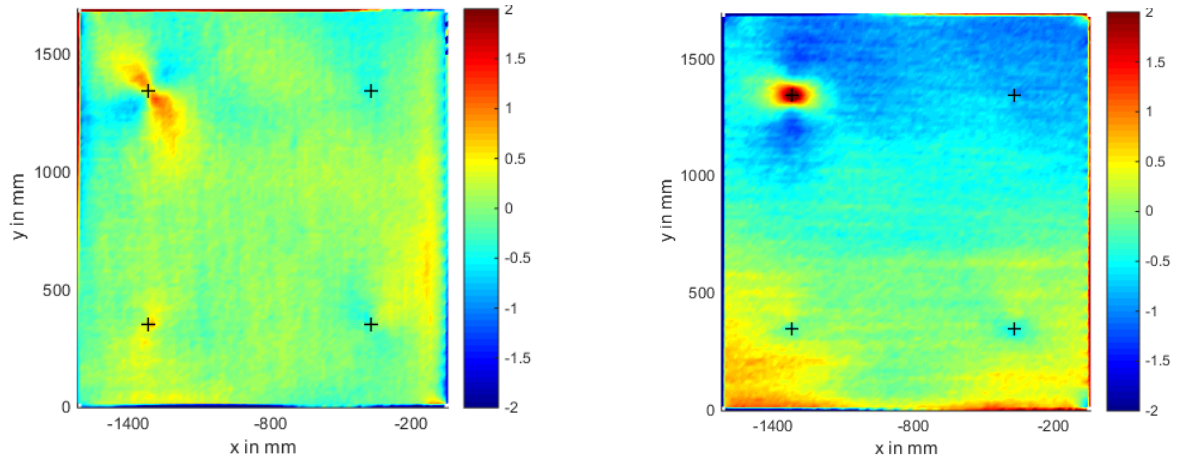


Abbildung 7.35: Plot der Slope Deviation Differenzen der Lösung des Optimierungsmodells in x und y (in mrad)

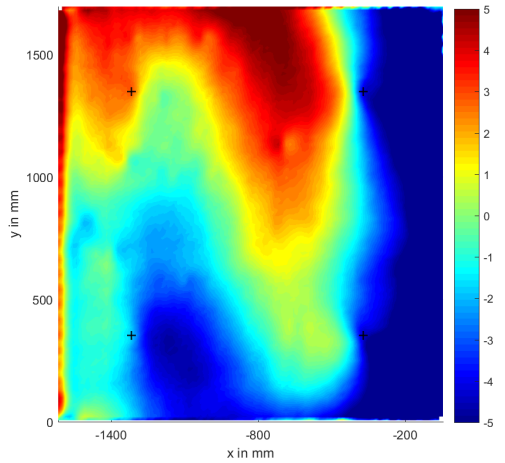
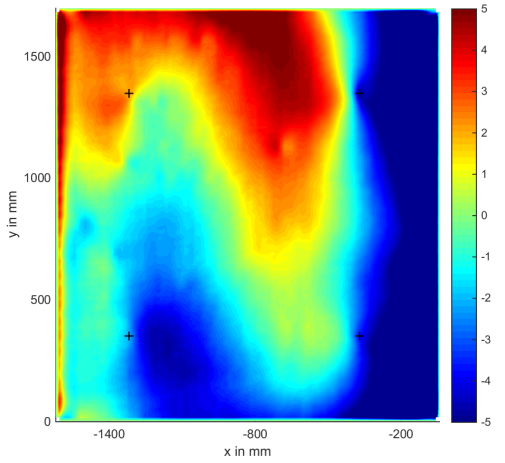
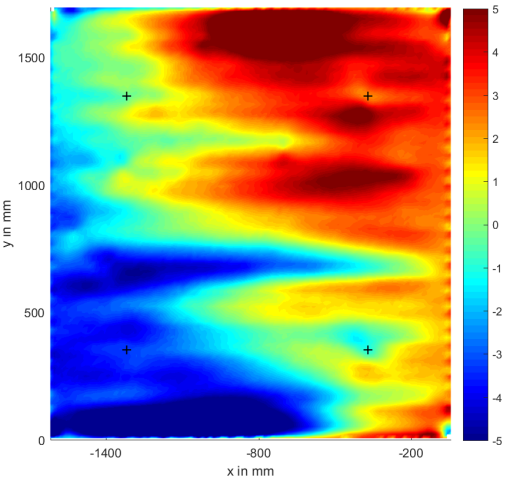
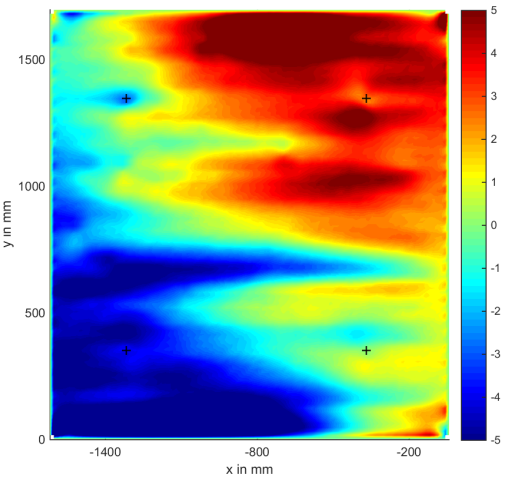
Eine Möglichkeit um die Optimierung zu verbessern ist die Slope Deviation in y mit in den Optimierungsprozess einzubauen, um die Optimierung mit Hilfe dieser zusätzlichen Informationen noch effizienter zu gestalten. Die Konvergenz des BOBYQA Algorithmus wird in Abbildung 7.37 dargestellt. Das Modell fällt stetig und erreicht nach 1001 Auswertungen eine Exaktheit von $f(x_*) = \triangle\text{SDx} = 4.131 \cdot 10^{-1}$ mrad. Für diesen Optimierungsvorgang werden 15 h:40 min benötigt.

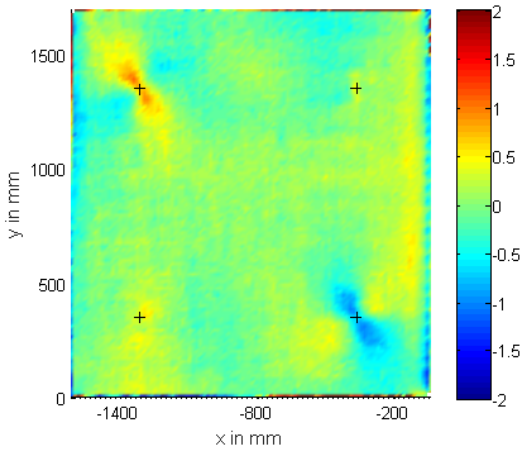
7.6.1 Multilevelvariante

Da die Auswertung der realen Messung mit 15 h:40 min eine lange Zeit in Ansprung nimmt, wird der Optimierungsprozess beschleunigt, indem die in Abschnitt 5.11.1 vorgestellte Multilevelvariante zum Einsatz kommt.

Tabelle A.20 im Anhang listest die vom Optimierer berechneten Translationen und Rotationen. Abbildung A.3 im Anhang zeigt die Slope Deviation Plots von reinem Optimierungsmodell und herstellungsbedingten Fehlern. Die Optimierungsmodelle der beiden Testläufe zeigen ein ähnliches Verformungsverhalten. Bei der Multileveloptimierung ist der RMS der Slope Deviation mit SDx = 2.4079 mrad größer als beim ersten Optimierungsversuch mit SDx = 2.382 mrad. In Tabelle 7.6 werden die Slope Deviation Plots in x und y für die Lösung des Optimierers sowie für die Referenz gegenübergestellt. Abbildung 7.37 zeigt die Plots der Differenzen der lokalen Slope Deviation von Optimierungs- und Referenzmodell in x und y. Diese verdeutlichen die Unterschiede zwischen Optimierungsergebnis und realer Messung. Auch hier werden bei der Betrachtung des Differenzenplot in y Unterschiede zwischen Optimierungmodell und Referenzlösung deutlich. Vor allem im unteren Bereich des Spiegels besteht noch die Möglichkeit der Optimierung durch die Hinzunahme der Slope Deviation in y in die Zielfunktion. Der Plot der Steigungsabweichungsdifferenzen in x zeigt kleine Differenzen um Pad 1 sowie Pad 4. Ansonsten stimmen die lokalen Steigungsabweichungsdifferenzen überein. Es wird eine Genauigkeit von $f(x_*) = \triangle\text{SDx} = 0.332$ mrad erreicht.

Tabelle 7.6: Slope Deviation Plots des Optimierungsmodells (links) und der Messung (rechts) der Multilevelvariante

	Optimierungsergebnis	Referenz
	SDx = 3.546 mrad	SDx = 3.564 mrad
Slope Deviation Plot in x		
Slope Deviation Plot in y		



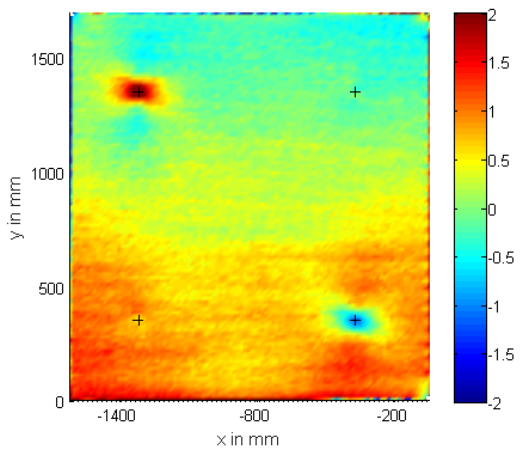


Abbildung 7.36: Plot der Slope Deviation Differenzen der Lösung des Optimierungsmodells mit Multilevelansatz in x und y (in mrad)

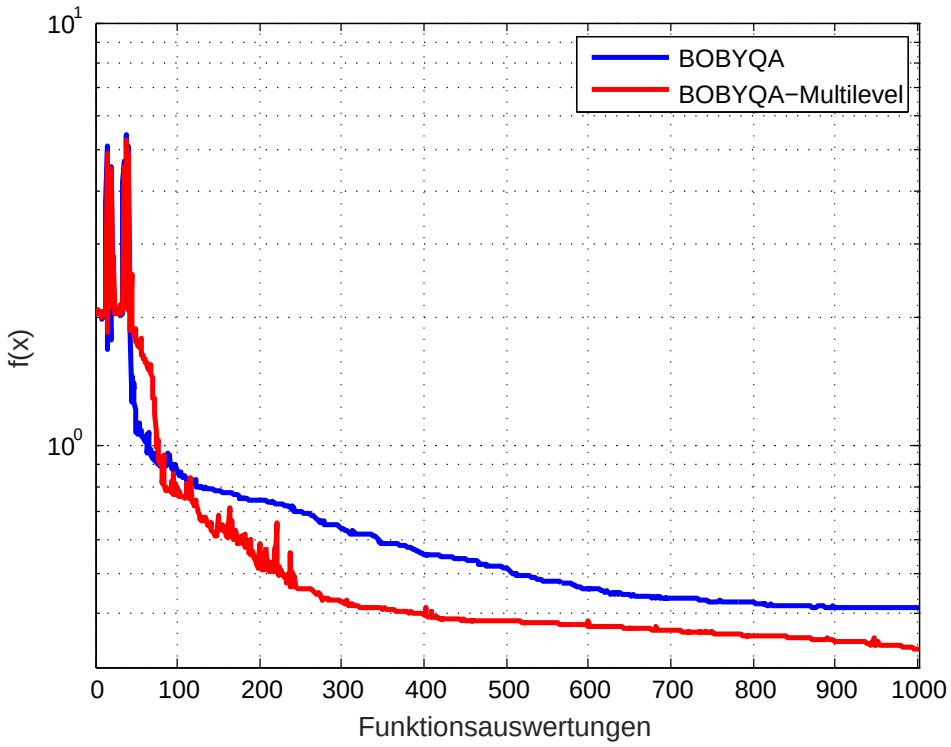


Abbildung 7.37: Konvergenz von BOBYQA für die reale Messung und der Multilevelvariante

Beim ersten Optimierungslauf erzielt BOBYQA eine Exaktheit von 0.413 mrad. Damit erreicht der Test mit der Multilevelvariante eine höhere Genauigkeit als der erste Durchlauf. Dies wird auch in Abbildung 7.37 verdeutlicht. Die Multilevelvatiante schafft es nach 100 Auswertungen kleinere Zielfunktionswerte als der reine BOBYQA Algorithmus zu finden. Des Weiteren findet beim Multilevelansatz nach 350 Funktionsauswertungen ein Wechsel von einer Netzteilung mit 100×100 zu 200×200 Elementen statt. Da kein Sprung bei den Multilevelfunktionswerten zu erkennen ist, zeigt dies, dass der Zusammenhang zwischen den Zielfunktionswerten der beiden Diskretisierungen gut approximiert wird. Die Optimierung stoppt nach 13 h. Damit gelingt es mit der Multilevelmethode nicht nur die Laufzeit der Optimierung zu verbessern, sondern auch die Genauigkeit der Lösung zu erhöhen.

8 Zusammenfassung und Ausblick

Die Untersuchung der Fragestellung, wie auftretende Lasten und Randbedingungen die Spiegelform im Kollektor beeinflussen, stand für diese Masterarbeit im Fokus. Ziel der vorliegenden Arbeit war die Entwicklung einer Methode, welche von gemessenen Spiegelformen Rückschlüsse auf die ursächlichen Verschiebungen und Rotationen an den Spiegelaufhängungen gibt. Dazu wurden die zwei ableitungsfreien Optimierungsverfahren BOBYQA (siehe Abschnitt 3.2.3) und CMA-ES (in Abschnitt 3.2.2) ausgewählt. Die Algorithmen wurden erfolgreich in die bereits vorhandene Auswerteprozedur implementiert, sodass in jeder Iteration der Einfluss von Verschiebungen und Rotationen mit Ansys simuliert und die Spiegelform mittels Matlab analysiert wird. BOBYQA und CMA-ES wurden auf ihre Performance hin untersucht und gegenübergestellt. Es haben Verbesserungsversuche für die Optimierungen stattgefunden.

Zunächst wurde das Spiegelmodell vereinfacht, indem die Montagepads aus der Ansys Simulation entfernt wurden. Des Weiteren kam es zu der Wahl einer gröberen Diskretisierung. Dies führte bereits zu einem besseren Laufzeitverhalten. Der CMA-ES Algorithmus wurde stärker fokussiert, um eine schnellere Optimierung zu erreichen. Die Nachteile des evolutionären Algorithmus CMA-ES wurden schnell deutlich. Es ist keine Garantie für das Finden eines Optimums in angemessener Zeit gegeben. Des Weiteren wird eine große Anzahl an teuren Funktionsauswertungen benötigt, um eine genügend genaue Lösung zu generieren. Da der BOBYQA Algorithmus trotz der Verbesserungen des CMA-ES eine höhere Genauigkeit und schnellere Konvergenz erreichte, folgten weitere Optimierungsläufe mittels des BOBYQA Algorithmus.

Der Einsatz eines Hybridalgorithmus hat gezeigt, dass die Problemstellung nicht eindeutig lösbar ist. Die gleichen Verformungen des Spiegels können durch unterschiedliche Verschiebungsvektoren erzeugt werden. Die Transformation einer Lösung in eine andere ist jedoch möglich. Um dieses Problem von vorne herein auszuschließen, wurde ein Pad fixiert. Der Vorteil für die Optimierung ist, dass die Dimension des Optimierungsproblems reduziert wurde. Es haben sich jedoch keine Verbesserungen in der Laufzeit gezeigt, da die teure Auswerteprozedur von Ansys gleich geblieben ist. Auch die Optimierungsergebnisse haben keine eindeutigen Vorteile gegenüber der Optimierung mit allen Parametern gezeigt. Ferner wurde versucht die Zielfunktion zu verbessern. Zum einen wurde das Hinzufügen der z-Deviation in den Optimierungsprozess getestet, zum anderen wurden die flächengewichteten Terme aus der Zielfunktion entfernt. Da der Optimierer beim Testlauf mit der z-Deviation in ein lokales Optimum gelaufen ist, wird davon abgeraten diese in die Zielfunktion einzubeziehen. Das Coordinate-Descent Verfahren lieferte - nach der Korrektur der Einheit der Rotationsparameter - einen vielversprechenden Ansatz für den Optimierungsprozess.

Es bietet sich an, diese Methode beispielsweise als Voroptimierer zu nutzen, um eine erste akzeptable Lösung zu generieren, mit der die Optimierung gestartet werden kann.

Des Weiteren wurde das validierte Modell mit Laborunterstruktur betrachtet. Auch hier gelingt es BOBYQA das Problem zu lösen. Zuletzt wurde der Algorithmus auf eine reale Messung eines RP3-Innenspiegels angewandt. Der Optimierer schafft es die gegebene Spiegelfacette gut nachzubilden. Auch ein erster Test mit dem Multilevelansatz fand in diesem Zusammenhang statt und lieferte eine Verbesserung in der Laufzeit und Genauigkeit. Insgesamt hat sich gezeigt, dass der BOBYQA Algorithmus ein robuster und schneller Optimierer ist, für den auch reale Messungen kein Problem darstellen.

In Zukunft wäre die Untersuchung der Slope Deviation in y-Richtung und die Berücksichtigung dieser in der Zielfunktion interessant. Dadurch werden weitere Informationen in den Optimierungsprozess aufgenommen. Dies bietet sich vor allem an, da Slope Deviation in x und y in der gleichen Einheit gemessen werden. Abbildung 7.35 zeigt die Slope Deviation Plots der Differenzen von Referenz und Optimierungsmodell der realen Messung. Es sind leichte Verformungen der Slope Deviation in y erkennbar. Bei der Hinzunahme der Steigungsabweichung in y in die Zielfunktion könnten diese Bereiche noch optimiert und damit das Ergebnis weiter verbessert werden. Ferner besteht die Möglichkeit den z-Deviation Plot zu nutzen und im Optimierungsprozess zu berücksichtigen. Die Abweichungen in z können an den Pads abgelesen werden und als Startwerte der z-Komponenten für die Optimierung dienen. Da die z-Deviation bei Deflektometriemessungen mit einem Fehler behaftet ist, welcher sich aufgrund der Integration aufsummiert, können die zugehörigen Fehlertoleranzen als Grenzen für die Optimierung dienen. Durch die zusätzlichen Daten kann der Algorithmus bereits in die richtige Richtung gelenkt werden. Außerdem ist eine Suchraumanalyse denkbar, um die Struktur des Suchraums zu untersuchen und ein besseres Verständnis von diesem zu erhalten. Darüber hinaus bietet es sich an das Coordinate-Descent Verfahren weiter zu optimieren. Beispielsweise ist es möglich die Spiegelfacette in vier Teile aufzuteilen und nur von den jeweiligen Teilstücken die Slope Deviation zu bestimmen. Es könnte dann das Pad in dem Teil mit dem größten Zielfunktionswert optimiert werden. So würden gut oder optimal liegende Aufhängepunkte übersprungen und dadurch teure Funktionsauswertungen vermieden werden. Falls es trotz allem zu lokalen Optima kommt, bietet sich ein Hybridalgorithmus aus dem CMA-ES und dem BOBYQA Algorithmus, ähnlich wie zu Anfang dieser Arbeit verwendet, an. Zunächst sollte der CMA-ES Algorithmus optimieren und einige gute Lösungskandidaten generieren und anschließend folgt der lokale und schnelle Optimierer BOBYQA, welcher die von CMA-ES berechneten Kandidaten als Startwerte für seine Optimierung nutzt. Ein nächstes Ziel wäre es statt externen Verschiebungen und Rotationen anliegende Kräfte an den Spiegelaufhängungen zu bestimmen. Da das Optimierungsmodell in Ansys an mindestens einer Stelle fixiert werden muss, ist es nicht einfach machbar direkt Kräfte an die Montagepads anzulegen. Mit Hilfe der hier verwendeten externen Verschiebungen ist jedoch eine Fixierung des Modells gewährleistet. Die Umrechnung der externen Verschiebungen in anliegende Kräfte kann in einem gesonderten Schritt folgen. Gelingt es dennoch statt der externen Verschiebungen direkt Kräfte anzulegen, so kann die Dimension des Optimierungsproblems auf 12 reduziert und der Optimierungsprozess beschleunigt werden.

Literaturverzeichnis

[Alt 11] *W. Alt: Nichtlineare Optimierung: Eine Einführung in Theorie, Verfahren und Anwendungen.* Vieweg+Teubner Verlag, Braunschweig/Wiesbaden, 2011

[BF 07] *P. Bayer, M. Fink: Optimization of concentration control by evolution strategies: Formulation, application, and assessment of remedial solutions.* Water resources research, 2007

[BG 10] *R. Buck, S. Giulian: Solare Kraftwerkprozesse für Wüstengebiete.* 13. Kölner Sonnenkolloquium, Köln-Porz, 2010

[BZ 12] *R. Burkard, U. Zimmermann: Einführung in die Mathematische Optimierung.* Springer-Verlag Berlin Heidelberg, 2012

[CGT 87] *A. Conn, N. Gould, P. Toint: Trust-Region Methods.* Society for Industrial and Applied Mathematics, Philadelphia, 1987

[CSV 08] *A. Conn, K. Scheinberg, L. Vicente: Introduction to derivative-free optimization.* Society for Industrial and Applied Mathematics, Philadelphia, 2008

[DS 96] *J. Dennis, R. Schnabel: Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Englewood Cliffs, USA, 1996

[EU 09] *Europäische Parlament und Rat der europäischen Union: Richtlinie* 2009/28/*EG des Europäischen Parlaments und des Rates vom 23. April 2009 zur Förderung der Nutzung von Energie aus erneuerbaren Quellen und zur Änderung und anschließenden Aufhebung der Richtlinien 2001/77/EG und* 2003/30/*EG.* 2009

[FG 09] *A. Farr, R. Gee: The SkyTrough Parabolic Trough Solar Collector.* ASME International Conference of Energy Sustainability, 2009

[Gey et al. 02] *M. Geyer, H. Lerchenmüller, V. Wittwer, A. Häberle, E. Lüpfert, K. Hennecke, W. Schiel, G. Brakmann: Solarthermische Kraftwerke – Technologie und Perspektiven.* FVS Themen 2002

[GMW 81] *P. Gill, W. Murray, M. Wright: Practical Optimization.* Academic Press, London, 1981

[GQT 66] *S. Goldfeldt, R. Quandt, H. Trotter: Maximization by quadratic hill-climbing.* Econometrica, 34: 541 – 551, 1966

[GW 08] *A. Griewank, A. Walther: Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation.* 2. Auflage, SIAM, 2008

[GHW 14] *D. Gross, W. Hauger, P. Wriggers: Technische Mechanik 4: Hydromechanik, Elemente der Höheren Mechanik, Numerische Methoden.* Springer-Verlag, Berlin Heidelberg, 2014

[GT 93] *C. Großmann, J. Terno: Numerik der Optimierung.* Teubner, Stuttgart, 1993

[Gün 15] *M. Günther: Energieeffizienz durch Erneuerbare Energien: Möglichkeiten, Potenziale, Systeme.* Springer Vieweg Wiesbaden, 2015

[HO 96] *N. Hansen, A. Ostermeier: Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation.* Proceedings of the 1996 IEEE International Conference on Evolutionary Computation, 1996

[HMK 03] *N. Hansen, S. D. Müller, P. Koumoutsakos: Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES).* Evolutionary computation, 2003

[HK 04] *N. Hansen, S. Kern: Evaluating the CMA Evolution Strategy on Multimodal Test Functions.* Springer, 2004

[Han 06] *N. Hansen: The CMA Evolution Strategy: A Comparing Review.* Springer-Verlag Berlin Heidelberg, 2006

[Han 11] *N. Hansen: The CMA Evolution Strategy: A Tutorial.* 2011

[Han 12] *N. Hansen: CMA-ES Source Code.* `https://www.lri.fr/~hansen/cmaes_inmatlab.html`, MATLAB-Code Version 3.61.beta, Stand: April, 2012, abgerufen am 23. Februar 2015

[Hof 14] *B. Hoffschmidt: Solare Komponenten.* RWTH Aachen, 2014

[Kle 11] *C. Kleine-Büning: Entwicklung eines FEM-Modells zur Verformungsanalyse von Spiegeln solarthermischer Parabolrinnenkraftwerke.* Diplomarbeit, Rheinisch-Westfälischen Technischen Hochschule Aachen, 2011.

[KS 09] *R. Kienzler, R. Schröder: Einführung in die Höhere Festigkeitslehre.* Springer-Verlag, Berlin Heidelberg, 200

[KR 14] *D. King, U. Römer: M.J.D. Powell's BOBYQA algorithm using a mex-*

function created with dlib's C++ implementation. `http://www.mathworks.com/matlabcentral/fileexchange/48689-bobyqa-algorithm`, Stand: Dezember 2014, abgerufen am 17. März 2015

[Kis03] *R. Kistner: Simulation of Solar Thermal Power Plants and Their Evaluation Emphasising on Energy Economic Issues and Financing Issues.* VDI Verlag, Düsseldorf, 2003

[Lev 44] *K. Levenberg: A method for the solution of certain problems in last squares.* Quarterly Journal on Applied Mathematics, 2: 164 – 168, 1944

[Lü et al. 07a] *E. Lüpfert, K. Pottler, S. Ulmer, K. Riffelmann, A. Neumann, B. Schiricke: Parabolic Trough Optical Performance Analysis Techniques.* Journal of Solar Energy Engineering, 2007

[Lü et al. 07b] *E. Lüpfert, K. Pottler, S. Ulmer, R. Pitz-Paal, W. Schiel, W. Platzer, A. Heimsath: Qualitätssicherungsmaßnahmen bei der Herstellung solarthermischer Kraftwerkskomponenten.* FVS Jahrestagung, 2007

[Lüp 12] *E. Lüpfert: Prüfmethoden für die geometrische Qualität von Solar-Konzentratoren.* 15. Kölner Sonnenkolloquium, Köln-Porz, 2012

[Mar 63] *D. Marquardt: An algorithm for least-square estimation of nonlinear parameters.* SIAM Journal on Applied Mathematics, 11: 431 – 441, 1963

[Mei 13] *S. Meiser: Analysis of Parabolic Trough Concentrator Mirror Shape Accuracy in Laboratory and Collector.* Dissertation, Rheinisch-Westfälischen Technischen Hochschule Aachen, 2013.

[MLSP 14] *S. Meiser, E. Lüpfert, B. Schiricke, R. Pitz-Paal: Conversion of parabolic trough mirror shape results measured in different laboratory setups.* Solar Energy Journal, 2014

[Nav 09] *P. Nava: Bedeutung von Spezifikationen für die Solarkraftwerksindustrie.* 12. Kölner Sonnenkolloquium, Köln-Porz, 2009

[NM 65] *J. Nelder, R. Mead: A simplex method for function minimization.* Computer Journal, 7, 1965

[NW 99] *J. Nocedal, S. Wright: Numerical Optimization.* Springer-Verlag, New York, 1999

[Pow 70] *M. Powell: A new algorithm for unconstrained optimization.* J. B. Rosen, O. L. Mangasarian, K. Ritter, Nonlinear Programming, Academic Press, London, 1970

[Pow 73] *M. Powell: On search directions for minimization algorithms.* Mathematical Programming 4, 1973

[Pow 75] *M. Powell: Convergence properties of a class of minimization algorithms.* O.L. Mangasarian, R.R. Meyer, and S.M. Robinson, editors, Nonlinear programming 2, Academic Press, New York, 1975

[Pow 00] *M. Powell: On the Lagrange functions of quadratic models defined by interpolation.* Technical report, Department of Applied Mathematics and Theoretical Physics, Camridge University, Cambridge, 2000

[Pow 02] *M. Powell: UOBYQA: unconstrained optimization by quadratic interpolation.* Mathematical Programming, Series A, 92, 2002

[Pow 03] *M. Powell: On the use of quadratic models in unconstrained minimization without derivatives.* Technical Report, Departement of Applied Mathematics and Theoretical Physics, Camridge University, Cambridge, 2003

[Pow 04a] *M. Powell: Least frobenius norm updating of quadratic models that satisfy interpolation conditions.* Mathematical Programming Series B, 2004

[Pow 04b] *M. Powell: The NEWUOA software for unconstrained optimization without derivatives.* Technical report, Department of Applied Mathematics and Theoretical Physics, Cambridge University, Cambridge, 2004

[Pow 09] *M. Powell: The BOBYQA algorithm for bound constrained optimization without derivatives.* Technical report, Department of Applied Mathematics and Theoretical Physics, Cambridge University, Cambridge, 2009

[PHHB 13] *R. Pitz-Paal, K. Hennecke, P. Heller, R. Buck: Solarthermische Kraftwerke: Konzentriertes Sonnenlicht zur Energieerzeugung nutzen.* FIZ Karlsruhe GmbH - Leibniz-Institut für Informationsinfrastruktur, 2013

[Pit 12] *R. Pitz-Paal: Vorlesungsskript: Grundlagen der Solartechnik.* RWTH Aachen, 2012

[Qua 09] *V. Quaschning: Regenerative Energiesysteme Technologie - Berechnung - Simulation.* Hanser Verlag München, 2009

[Ra et al. 04] *R. Kistner, K. Grethe, M. Geyer, H. Gladen, J. Nebrera: The Progress of the AndaSol projects in Spain.* EuroSun2004, 2004

[Rec 73] *I. Rechenberg: Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Frommann-Holzboog, Stuttgart, 1973.

[RR 13] *G. Reich, M. Reppich: Regenerative Energietechnik: Überblick über ausgewählte Technologien zur nachhaltigen Energieversorgung.* Springer Vieweg, Wiesbaden 2013

[Schi 11] *W. Schiel: Mit dem DLR zum „Next Generation Parabolic Trough".* Kölner Sonnenkolloquium, Jülich, 2011

[Scho 13] *G. Scholz: Heisswasser- und Hochdruckdampfanlagen- Planungshandbuch für Industrie- und Fernwärmeversorgung.* Springer-Verlag Berlin Heidelberg, 2013

[Tse 01] *P. Tseng: Convergence of a Block Coordinate Descent Method for Nondifferentiable Minimization.* Journal of Optimization Theory and Applications, Volume 109, 2001

[Wat 09] *H. Watter: Nachhaltige Energiesysteme: Grundlagen, Systemtechnik und Anwendungsbeispiele aus der Praxis.* Vieweg + Teubner, Wiesbaden, 2009

[Wil 09] *S. Wilbert: Weiterentwicklung eines optischen Messsystems zur Bestimmung der Formabweichungen von Konzentratoren solarthermischer Kraftwerke unter dynamischem Windeinfluss* Rheinischen Friedrich-Wilhelms-Universität Bonn, Diplomarbeit, 2009

[WSLF 13] *V. Wesselak, T. Schabbach, T. Link, J. Fischer: Regenerative Energietechnik.* Springer Vieweg Berlin Heidelberg, 2013

[Wri 15] *S. Wright: Coordinate descent algorithms.* Mathematical Programming, 2015

A Anhang

A.1 Basis Trust-Region Algorithmus

Algorithmus A.1 : Basis Trust-Region Algorithmus [Alt 11]

Schritt 1: Initialisierung:
Gegeben seien die Konstanten $0 < \delta_1 < \delta_2 < \tilde{\delta}$, $\sigma_1 \in (0,1)$, $\sigma_2 > 1$ und ein Radius $0 < \rho_0 \leq \tilde{\delta}$. Wähle $x_0 \in \mathbb{R}^n$, berechne $q_0 = \nabla f(x_0)$, $Q_0 = \nabla^2 f(x_0)$ und $k := 0$.
Schritt 2: Modelldefinition:
Definiere ein Modell Q_k in $\mathcal{B}_{\rho_k}(x_k)$.
Schritt 3: Berechnung des Schrittes:
Berechne eine Richtung d^k, die (3.32) erfüllt.
Schritt 4: Abbruchkriterium:
Falls $f(x_k) = Q_k(d_k)$, STOP.
Schritt 5: Quotientenberechnung:
Berechne den Quotienten r_k.
Schritt 6: Iterationspunkt und Trust-Region-Radius Update:
Falls $r_k \geq \delta_1$ (erfolgreicher Iterationsschritt):
Setze $x^{k+1} = x^k + d^k$ und berechne $q_{k+1} = \nabla f(x_{k+1})$, $H_{k+1} = \nabla^2 f(x_{k+1})$. Wähle

$$\rho_{k+1} \in \begin{cases} [\sigma_1 \rho_k, \rho_k] & \text{, falls } r_k \in [\delta_1, \delta_2) \\ [\rho_k, \min\{\sigma_2 \rho_k, \tilde{\delta}\}] & \text{, falls } r_k \geq \delta_2. \end{cases}$$

Setze $k := k + 1$ und gehe zu 2.
Falls $r_k < \delta_1$ (nicht erfolgreicher Iterationsschritt):
Wähle $\rho_{k+1} \in (0, \sigma_1 \rho_k]$.
Setze $x_{k+1} = x_k$, $q_{k+1} = q_k$, $H_{k+1} = H_k$, $k := k + 1$ und gehe zu 2.

A.2 CMA-ES Algorithmus

Algorithmus A.2 : CMA-ES Algorithmus

Schritt 1: Initialisierung:
Setzte die Parameter λ, μ, w_i für $i = 1, ..., \mu$, c_σ, d_σ, c_c, c_{cov} und μ_{eff}.
Setzte die evolutionären Pfade $p_{\sigma,0} = p_{c,0} = 0$, die Kovarianzmatrix $C_0 = I$ und $t = 0$.
Wähle den Mittelwert $m_t \in \mathbb{R}^n$ und die Schrittweitenkontrolle $\sigma_t \in \mathbb{R}_+$ problemspezifisch.
Schritt 2: Abbruchkriterium:
Maximale Anzahl an Funktionsauswertungen oder Iterationen erreicht?
Kondition der Kovarianzmatrix bestimmten Wert unterschritten?
Schritt 3: Wahl einer neuen Population von Suchpunkten:

$$\begin{aligned} z_{k,t} &\sim \mathcal{N}(0, I) \\ y_{k,t} &= B_t D_t z_{k,t} && \sim \mathcal{N}(0, C_t) \\ x_{k,t} &= m_t + \sigma_t y_{k,t} && \sim \mathcal{N}(m_t, \sigma_t^2 C_t) \end{aligned}$$

Schritt 3: Selektion und Rekombination:

$$m_{t+1} = \sum_{i=1}^{\mu} w_i x_{i,t} \quad \text{mit} \quad \sum_{i=1}^{\mu} w_i = 1,\ w_i > 0$$

Schritt 4: Schrittweitenkontrolle:

$$\begin{aligned} p_{\sigma,t+1} &= (1 - c_\sigma) p_{\sigma,t} + \sqrt{c_\sigma (2 - c_\sigma) \mu_{eff}}\, C_t^{-\frac{1}{2}} \frac{m_{t+1} - m_t}{\sigma_t} \\ \sigma_{t+1} &= \sigma_t \exp\left(\frac{c_\sigma}{d_\sigma} \left(\frac{\|p_{\sigma,t}\|}{E\|\mathcal{N}(0, I)\|} - 1 \right) \right) \end{aligned}$$

Schritt 5: Anpassung der Kovarianzmatrix:

$$\begin{aligned} p_{c,t+1} =& (1 - c_c) p_{c,t} + \sqrt{c_c (2 - c_c) \mu_{eff}} \frac{m_{t+1} - m_t}{\sigma_t} \\ C_{t+1} =& (1 - c_{cov}) C_t + \frac{c_{cov}}{\mu_{eff}} p_{c,t+1} p_{c,t+1}^T \\ & + c_{cov} (1 - \frac{1}{\mu_{eff}}) \sum_{i=1}^{\mu} w_i \left(\frac{x_{i,t+1} - m_t}{\sigma_t} \right) \left(\frac{x_{i,t+1} - m_t}{\sigma_t} \right)^T \end{aligned}$$

A.3 Ein- und Ausgabewerte für die Optimierung

A.3.1 CMA-ES

xstart	x_0	Für den Startvektor gilt $x_0 = 0_{\mathbb{R}^n}$, wobei n die Dimension des Problems darstellt.
LBounds *UBounds*	a b	Da maximale Verschiebungen und Rotationen von ± 5 mm und ± 5 mrad an den Pads stattfinden, werden diese für die Box-Beschränkungen empfohlen.
sigma	σ	Für eine realistische globale Suche sollte die Startschrittweite σ auf eine Wert zwischen 0.2 und 0.5 mal der Länge des Intervalls gesetzt werden.
PopSize	λ	Für λ gilt standardmäßig $\lambda = 4 + \lfloor 3 \ln n \rfloor$. Größere Populationen können besser mit verrauschten Funktionen und lokalen Minima umgehen, während kleinere Populationen ein schnelleres Konvergenzverhalten aufweisen.
MaxFunEval		Für die maximale Anzahl an Funktionsauswertungen wird ein Wert zwischen $30 \cdot n$ und $300 \cdot n$ empfohlen.
StopFitness	ϵ_{end}	Genauigkeit mit der der Algorithmus stoppt. Für die Optimierung von 3 bis 6 Translationsvariablen wurde ein Wert zwischen 10^{-3} und 10^{-4} gewählt.
fmin	$f(x_*)$	Bester Zielfunktionswerte der Optimierung.
xmin	x_*	Zugehöriger Optimierungsvektor des besten Zielfunktionswerts.

A.3.2 BOBYQA

xstart	x_0	Für den Startvektor gilt $x_0 = 0_{\mathbb{R}^n}$, wobei n die Dimension des Problems darstellt.
LBounds *UBounds*	a b	Da maximale Verschiebungen und Rotationen von ± 5 mm und ± 5 mrad an den Pads stattfinden, werden diese für die Box-Beschränkungen empfohlen.
npt		Die Anzahl der Punkte, welche für die Approximation der Zielfunktion durch das Aufstellen des quadratischen Modells benutzt werden. Es muss gelten $npt \in [n+2, \frac{1}{2}(n+1)(n+2)]$. Es wird eine Wahl von $npt = 2n + 1$ empfohlen.
rho_beg	ρ_{beg}	Für den Start-Trust-Region müssen die folgenden beiden Bedingungen gelten: $\rho_{beg} > 0$ $\min_{i=1,..,n} \lvert b_i - a_i \rvert > 2\rho_{beg}$. Für die in dieser Arbeit vorliegende Problemstellung wird ein Wert von $\rho_{beg} = 1$ empfohlen.
rho_end	ρ_{end}	Für den finale Trust-Region Radius muss gelten $0 < \rho_{end} < \rho_{beg}$. Falls kein Wert gewählt wird, so $\rho_{end} = 10^{-6}$.
maxFunEval		Der Algorithmus bricht nach dem Erreichen der maximalen Anzahl an Funktionsauswertungen ab. Je größer die Dimension des Optimierungsproblems desto größer sollte *maxFunEval* gewählt werden. Für 21 bis 24 Optimierungsparameter sind in dieser Arbeit 1000 Funktionsauswertungen gegeben. Mit *maxFunEval* = 1000 wird ein gutes Optimierungsergebnis erreicht.
nF_opt	$f(x_*)$	Bester Zielfunktionswerte der Optimierung.
vX_opt	x_*	Zugehöriger Optimierungsvektor des besten Zielfunktionswerts.

A.4 Lösungen des dreidimensionalen Testproblems

A.4.1 Testproblem 1

Tabelle A.1: Resultate für BOBYQA mit $\rho_{beg} = 0.01$

Elemente	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
10×10	0.5000	1.4985	1.0000	$1.0245 \cdot 10^{-4}$	225	232	58
25×25	0.5000	1.4999	0.9999	$1.5682 \cdot 10^{-5}$	186	193	48
50×50	0.5000	1.5000	0.9999	$3.5362 \cdot 10^{-5}$	127	134	38
75×75	0.5000	1.4999	1.0002	$1.6792 \cdot 10^{-4}$	243	250	85
100×100	0.5000	1.5000	1.0001	$5.3160 \cdot 10^{-5}$	180	187	67
200×200	0.5000	1.5001	1.0000	$2.6909 \cdot 10^{-5}$	214	221	84

Tabelle A.2: Resultate für BOBYQA mit $\rho_{beg} = 1.0$

Elemente	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
10×10	0.5000	1.5024	0.9999	$1.7091 \cdot 10^{-4}$	240	247	65
25×25	0.5000	1.5000	1.0000	$1.5682 \cdot 10^{-5}$	204	211	55
50×50	0.5000	1.5001	1.0000	$4.7027 \cdot 10^{-5}$	156	163	48
75×75	0.5000	1.5002	0.9999	$1.1510 \cdot 10^{-4}$	168	175	54
100×100	0.5000	1.4999	0.9998	$1.1292 \cdot 10^{-4}$	151	158	56
200×200	0.5000	1.5004	0.9999	$1.9127 \cdot 10^{-4}$	181	188	122

Tabelle A.3: Resultate für BOBYQA mit $\rho_{beg} = 2.0$

Elemente	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
10×10	0.5000	1.4972	1.0000	$1.9380 \cdot 10^{-4}$	208	215	53
25×25	0.5000	1.5002	1.0000	$4.8479 \cdot 10^{-5}$	213	220	59
50×50	0.5000	1.5006	1.0000	$2.1827 \cdot 10^{-4}$	134	141	40
75×75	0.5000	1.5000	1.0000	$1.3638 \cdot 10^{-5}$	196	203	61
100×100	0.5000	1.5000	1.0001	$4.7737 \cdot 10^{-5}$	178	185	66
200×200	0.5000	1.4999	0.9999	$8.5092 \cdot 10^{-5}$	170	177	115

Tabelle A.4: Resultate des CMA-ES Algorithmus für verschiedene Diskretisierungen

Elemente	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
50×50 (Test 1)	0.4963	1.4943	0.9961	$1.0896 \cdot 10^{-2}$	36	254	72
50×50 (Test 2)	0.4880	1.3874	0.9402	$5.3524 \cdot 10^{-2}$	36	254	71
200×50 (Test 3)	0.5068	1.5414	1.0322	$2.5856 \cdot 10^{-2}$	36	254	162
200×50 (Test 4)	0.5014	1.5239	1.0090	$1.2171 \cdot 10^{-2}$	36	254	164

Tabelle A.5: Resultate für CMA-ES mit $\sigma = 0.01$

	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
Test 1	0.4807	0.6032	0.9317	$3.3911 \cdot 10^{-1}$	36	254	72
Test 2	0.4575	1.5209	0.8625	$9.3895 \cdot 10^{-2}$	36	254	73

Tabelle A.6: Resultate für CMA-ES mit $\sigma = 1.0$

	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
Test 1	0.4963	1.4943	0.9961	$1.0896 \cdot 10^{-2}$	36	254	72
Test 2	0.4880	1.3874	0.9402	$5.3524 \cdot 10^{-2}$	36	254	71

Tabelle A.7: Resultate für CMA-ES mit $\sigma = 2.5$

	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
Test 1	0.5363	1.1887	1.0259	$1.8124 \cdot 10^{-1}$	36	254	72
Test 2	0.5009	1.5361	1.0140	$1.8918 \cdot 10^{-2}$	36	254	71

Tabelle A.8: Resultate des BOBYQA Algorithmus von Klebstoff- und Padmodell

	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
Padmodell	0.5000	1.5001	1.0000	$4.7027 \cdot 10^{-5}$	156	163	48
Klebstoffmodell	0.5000	1.4996	1.0000	$2.1295 \cdot 10^{-4}$	109	116	14

Tabelle A.9: Resultate des CMA-ES Algorithmus von Klebstoff- und Padmodell

	Beste Lösung x_*			Bester Funktionswert $f(x_*)$	Iteration	Funktionsauswertung	Laufzeit [min]
	x_1	y_1	z_1				
Padmodell (Test 1)	0.4963	1.4943	0.9961	$1.0896 \cdot 10^{-2}$	36	254	72
Padmodell (Test 2)	0.4880	1.3874	0.9402	$5.3524 \cdot 10^{-2}$	36	254	71
Klebstoffmodell (Test 1)	0.4995	1.4954	1.0000	$3.0321 \cdot 10^{-3}$	36	254	29
Klebstoffmodell (Test 2)	0.4891	1.5063	0.9397	$4.3716 \cdot 10^{-2}$	36	254	30

A.5 Lösungen des sechsdimensionalen Testproblems

Tabelle A.10: Resultate von Test 1, sechsdimensionales Testproblem für BOBYQA und CMA-ES

	x_*						$f(x_*)$	Funktionsauswertung	Laufzeit [h:min]
	x_1	y_1	z_1	x_3	y_3	z_3			
BOBYQA	0.998	1.001	0.992	−0.003	0.003	−0.008	$1.384 \cdot 10^{-3}$	661	2 : 14
CMA-ES	0.924	1.097	0.449	−0.080	−0.067	−0.566	$7.628 \cdot 10^{-2}$	902	2 : 36

Tabelle A.11: Resultate von Test 2, sechsdimensionales Testproblem für BOBYQA und CMA-ES

	x_*						$f(x_*)$	Funktionsauswertung	Laufzeit [h:min]
	x_1	y_1	z_1	x_3	y_3	z_3			
CMA-ES $\epsilon_{end} = 1 \cdot 10^{-3}$	1.001	1.001	1.002	0.001	0.001	0.002	$9.053 \cdot 10^{-4}$	1442	4 : 39
BOBYQA $\rho_{end} = 1 \cdot 10^{-4}$	0.998	1.001	0.989	−0.002	0.001	−0.011	$1.514 \cdot 10^{-3}$	743	2 : 25
BOBYQA $\rho_{end} = 1 \cdot 10^{-6}$	1.000	1.000	0.998	0.000	0.000	−0.002	$2.218 \cdot 10^{-4}$	1443	4 : 38

Tabelle A.12: Resultate von Test 3, sechsdimensionales Testproblem für BOBYQA und CMA-ES

	x_*						$f(x_*)$	Iteration	Funktions-auswertung	Laufzeit [h:min]
	x_1	y_1	z_1	x_3	y_3	z_3				
BOBYQA: $\rho_{end} = 1 \cdot 10^{-4}$	1.005	1.005	0.996	0.005	−0.005	−0.005	$1.445 \cdot 10^{-3}$	949	956	3 : 06
BOBYQA: $\rho_{end} = 1 \cdot 10^{-6}$	0.999	0.999	0.999	−0.001	−0.001	−0.001	$2.400 \cdot 10^{-4}$	994	1001	3 : 13
CMA-ES: $\sigma = 2.5$, $(4, 9)$	1.032	1.046	1.129	0.034	0.009	0.137	$2.537 \cdot 10^{-2}$	111	1001	3 : 10
CMA-ES: $\sigma = 2.5$, $(2, 4)$ (1)	0.998	1.001	0.981	−0.002	−0.006	−0.019	$2.699 \cdot 10^{-3}$	250	1002	3 : 21
CMA-ES: $\sigma = 2.5$, $(2, 4)$ (2)	1.125	1.169	0.780	0.129	−0.161	−0.197	$4.728 \cdot 10^{-2}$	250	1002	3 : 15
CMA-ES: $\sigma = 1.0$, $(4, 9)$	0.996	0.986	0.996	−0.003	−0.023	0.000	$9.905 \cdot 10^{-3}$	111	1001	3 : 16
CMA-ES: $\sigma = 0.5$, $(4, 9)$	0.999	0.998	1.005	−0.001	0.004	0.005	$9.419 \cdot 10^{-4}$	111	1001	3 : 15
CMA-ES: $\sigma = 2.5$, $(4 + 9)$	1.006	1.005	0.935	0.004	−0.024	−0.072	$1.224 \cdot 10^{-3}$	111	1001	3 : 10
CMA-ES: $\sigma = 2.5$, $(2 + 4)$	0.979	1.049	0.679	−0.023	−0.043	−0.317	$4.130 \cdot 10^{-2}$	250	1002	3 : 11

A.6 Lösungen des 12-dimensionalen Testproblems

Tabelle A.13: Resultate der z-Deviation Optimierung

Beste Lösung z_*				Bester Funktions-wert $f(z_*)$	Iteration	Funktions-auswertung	Laufzeit [min]
z_1	z_2	z_3	z_4				
0.8895	−0.0266	1.8881	−0.6951	$8.12515 \cdot 10^{-1}$	96	105	22

Tabelle A.14: Resultate vom Hybridalgorithmus, 12-dimensionale Testproblem

	x_*												$f(x_*)$
	x_1	y_1	z_1	x_2	y_2	z_2	x_3	y_3	z_3	x_4	y_4	z_4	
CMA-ES	-1	-1	-1	0	0	0	0	0	0	0	0	0	6.6846
BOBYQA	0.5368	-0.2538	0.3708	-0.3071	-0.2546	-0.6519	-0.4634	-0.0997	-0.1298	0.1929	-0.1007	-0.6523	0.0032
CMA-ES	0.4468	-4	0.4749	-3.4229	-2.3406	-0.3197	-3.8381	-0.8029	-0.5314	-1.8167	2.6630	2.8349	13.5900
BOBYQA	-3.9884	-1.8941	1.4482	-3.2926	-1.8972	-0.0119	-4.9914	-0.1781	0.9345	-2.7923	-0.1743	-0.0098	0.0301
CMA-ES	-0.4952	2.801	-1.9289	3.9343	-1.0622	-0.0706	0.9085	-1.9124	-1.4499	-0.5496	-2.5425	-0.6746	20.5752
BOBYQA	1.5432	0.1342	-0.9328	0.0797	0.1133	-1.7183	0.5422	-0.3842	-1.4332	0.5808	-0.3521	-1.7170	0.0166
CMA-ES	1.5180	0.9587	0.4148	0.5047	-1.7257	-0.7504	-3.8634	-2.4351	-2.3323	0.3236	1.3606	-1.0030	16.0949
BOBYQA	0.1215	-2.1008	0.4153	-0.2452	-2.0987	-0.6879	-0.8798	-1.4720	-0.0884	0.2555	-1.4770	-0.6867	0.0124
CMA-ES	0.1974	-0.3022	1.5030	0.5150	-3.7217	1.4983	1.8038	-0.0718	0.7946	-0.2759	3.6954	2.1107	11.6579
BOBYQA	-1.784	-1.2325	1.5185	-0.3756	-1.2339	-0.0539	-2.7876	1.1926	1.0030	0.1244	1.1925	-0.0543	0.0415
CMA-ES	0.3489	2.3645	1.5913	2.7595	-0.6485	-0.7136	-0.7952	-3.7644	-1.3712	4	2.5239	-1.5414	5.7567
BOBYQA	1.6286	1.9592	0.7368	-0.3775	1.9854	0.2742	0.6323	0.8806	0.2537	0.1314	0.8798	0.2987	0.0450
CMA-ES	1.3517	0.4804	0.6458	4	-2.9514	-0.1003	-5	-1.0929	2.0890	2.1066	0.6321	-4	25.3180
BOBYQA	-1.5838	-2.1688	1.3164	1.1877	-2.1971	-0.8947	-2.5864	1.6798	0.7911	1.6859	1.6924	-0.8917	0.0755
CMA-ES	1.2926	2.1460	2.3705	0.4344	-5	-2.9642	-4	-1.1564	1.7839	-3.6612	-1.3667	2.0281	31.1158
BOBYQA	0.7267	-0.6153	1.2300	-0.8882	-0.6097	0.3511	-0.2731	-1.2225	0.73416	-0.3891	-1.2275	0.3511	0.0124
CMA-ES	-2.5446	2.37791	0.3832	-0.1812	-1.2183	-2.5759	-5	-1.5077	1.3926	0.4061	1.6110	0.3931	8.8137
BOBYQA	-0.0435	-0.4535	0.8734	-0.6034	-0.4639	-0.3820	-1.0424	0.0161	0.3729	-0.1036	0.0240	-0.3786	0.01969
CMA-ES	0.6556	2.2155	4	0.3334	-3.0190	-2.9267	-4	-4	-1.8407	-3.4115	-1.4040	1.6490	25.1888
BOBYQA	-0.6846	0.3766	1.2208	-4.3465	0.3780	0.8630	-1.6807	-2.3073	0.7360	-3.8466	-2.3107	0.8628	0.0459
CMA-ES	2.6730	0.7672	-0.1400	4	-2.7191	-2.0604	-4	-1.2506	0.5944	-5	-4	3.2622	52.8202
BOBYQA	0.1986	-1.4803	1.3232	-0.8465	-1.4545	0.4570	-0.8035	-1.5293	0.8165	-0.3456	-1.5480	0.4570	0.0168
CMA-ES	-1.3475	1.0661	3.5454	-2.3524	-3.2630	-3.3140	-2.4080	-3.5977	2.3084	-3.3756	-0.2152	-0.9710	8.4656
BOBYQA	-0.5997	-1.8567	2.0540	-2.3602	-1.8308	0.9193	-1.5966	-2.5055	1.5577	-1.8645	-2.5624	0.9108	0.0419
CMA-ES	-0.2139	5	4	1.1689	-5	-5	-5	0.1189	1.2541	-2.5204	-3.5544	1.7768	28.7646
BOBYQA	-1.4256	-2.8857	2.6893	-0.9441	-2.8819	1.314	-2.4276	-1.3839	2.1809	-0.4443	-1.3849	1.3129	0.0257
CMA-ES	-0.9552	1.2912	3.8056	-1.8264	-3.2152	-3.2382	-2.7322	-3.6932	1.4959	-3.3826	-0.4480	-0.4579	11.4390
BOBYQA	-1.1941	-0.4108	0.8428	-4.2754	-0.3993	0.2756	-2.1893	-2.4873	0.3582	-3.7765	-2.4871	0.2732	0.0369

Tabelle A.15: Slope Deviation Plots der Zwischenlösungen des Hybridalgorithmus

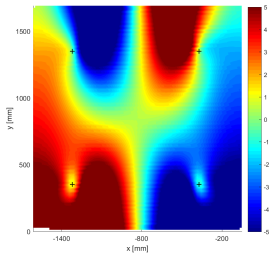
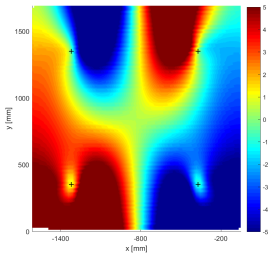
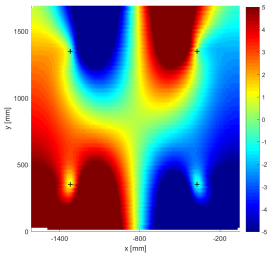
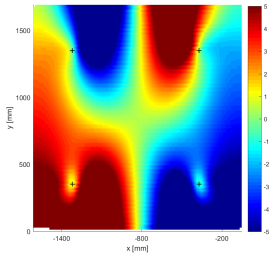
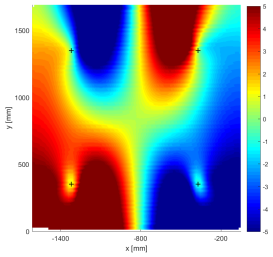
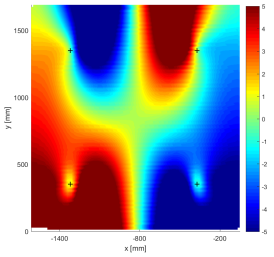
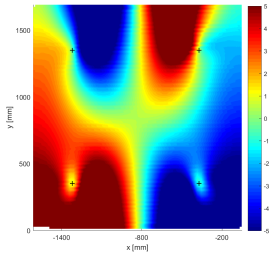
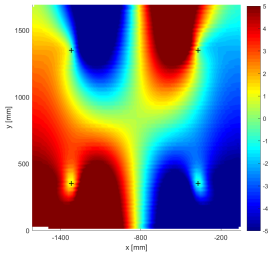
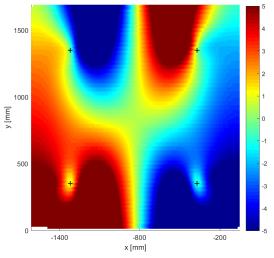
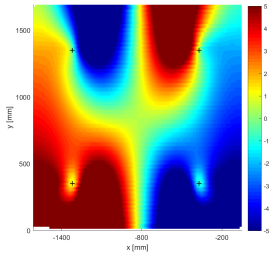
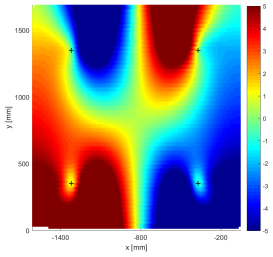
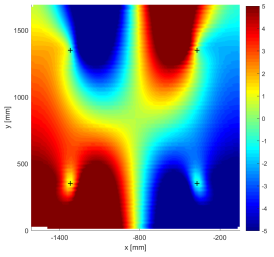
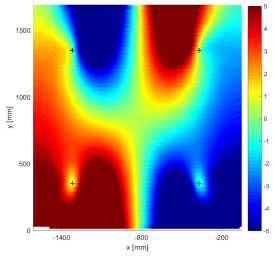
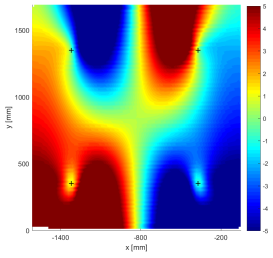
Test 1:	Test 2:	Test 3:
SDx =4.1795 mrad	SDx = 4.18 mrad	SDx =4.1803 mrad
		
Test 4:	**Test 4:**	**Test 6:**
SDx = 4.1793 mrad	SDx = 4.1796 mrad	SDx =4.1747 mrad
		
Test 7:	**Test 8:**	**Test 9:**
SDx = 4.1759 mrad	SDx = 4.1818 mrad	SDx =4.1798 mrad
		
Test 10:	**Test 11:**	**Test 12:**
SDx = 4.1799 mrad	SDx = 4.1786 mrad	SDx =4.1808 mrad
		
Test 13:	**Test 14:**	
SDx = 4.1802 mrad	SDx = 4.1792 mrad	
		

Tabelle A.16: Slope Deviation Differenzen Plots der Zwischenlösungen des Hybridalgorithmus

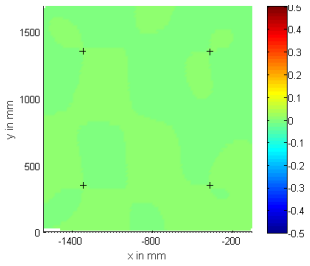
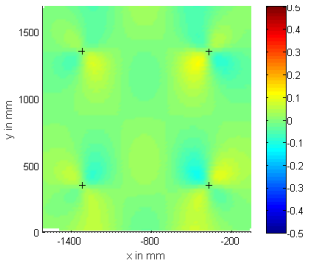
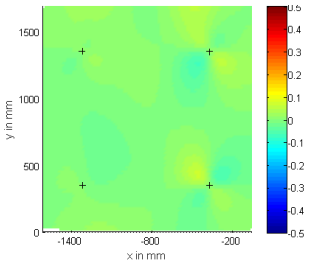
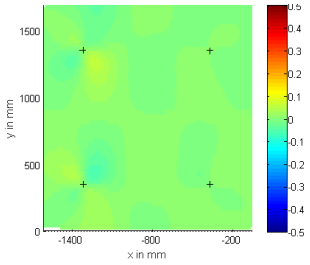
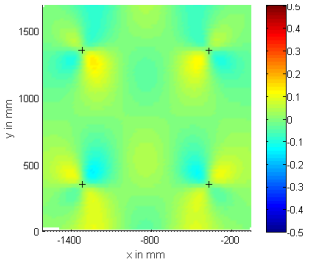
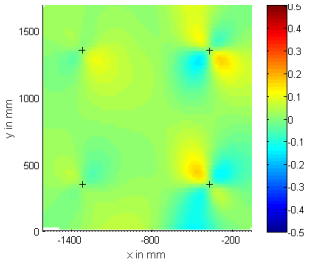
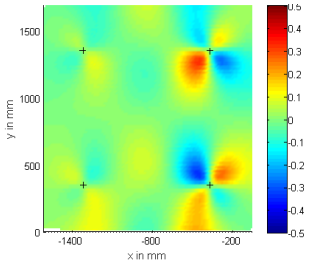
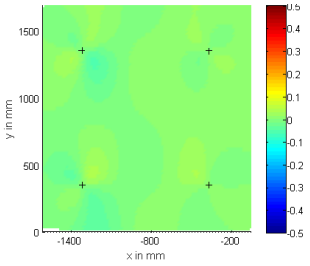
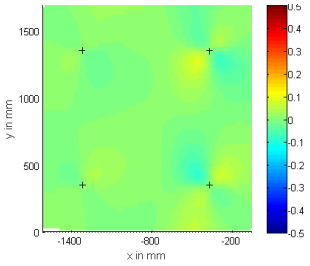
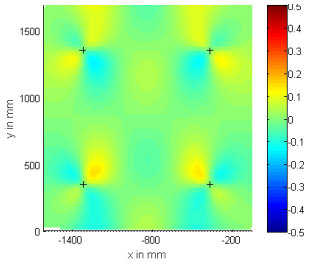
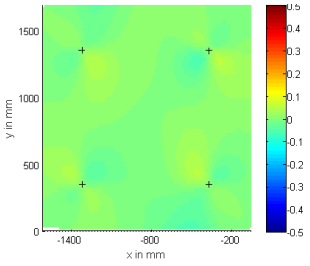
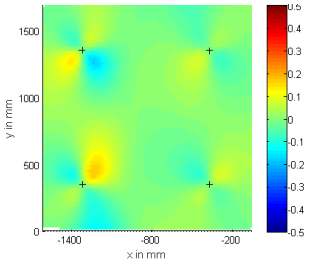
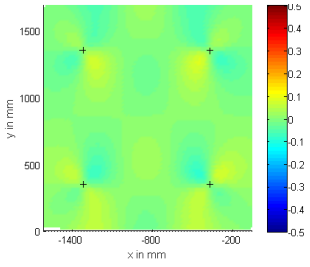
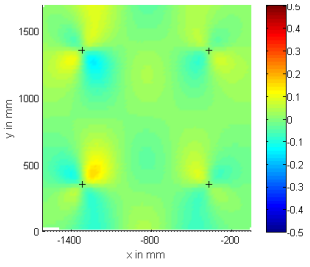
Test 1:	Test 2:	Test 3:
△SDx =0.0032 mrad	△SDx =0.0301 mrad	△SDx =0.0166 mrad
		
Test 4:	**Test 4:**	**Test 6:**
△SDx =0.0124 mrad	△SDx =0.0415 mrad	△SDx =0.0449 mrad
		
Test 7:	**Test 8:**	**Test 9:**
△SDx =0.0755 mrad	△SDx =0.0124 mrad	△SDx =0.0197 mrad
		
Test 10:	**Test 11:**	**Test 12:**
△SDx =0.0460 mrad	△SDx = 0.0168 mrad	△SDx =0.0419 mrad
		
Test 13:	**Test 14:**	
△SDx =0.0257 mrad	△SDx =0.0369 mrad	
		

Tabelle A.17: Resultate der Parameterreduktion, 12-dimensionale Testproblem

	x_*												$f(x_*)$	Funktions-auswertung	Laufzeit [h:min]
	x_1	y_1	z_1	x_2	y_2	z_2	x_3	y_3	z_3	x_4	y_4	z_4			
4 Pads variabel	0.5368	-0.2538	0.3708	-0.3071	-0.2546	-0.6519	-0.4634	-0.0997	-0.1298	0.1929	-0.1007	-0.6523	$3.1924 \cdot 10^{-3}$	1001	3:16
Pad 1 fest	0	0	0	-0.7323	-0.0021	-1.0494	-1.0004	0.2647	-0.5009	-0.2322	0.2656	-1.0496	$4.9410 \cdot 10^{-3}$	1001	3:18
Pad 2 fest	1.0685	-0.0010	0.9641	0	0	0	0.0683	-0.0746	0.4641	0.5003	-0.0753	0.0006	$2.7711 \cdot 10^{-3}$	1001	3:16

A.7 Lösungen des 20-dimensionalen Testproblems

Tabelle A.18: Resultate der Parameterreduktion, 20-dimensionale Testproblem

Para-	x_*																			
meter	x_1	y_1	z_1	α_1	β_1	x_2	y_2	z_2	α_2	β_2	x_3	y_3	z_3	α_3	β_3	x_4	y_4	z_4	α_4	β_4
20	0.851	0.387	0.042	-0.166	-0.024	0.132	0.011	-0.196	0.042	0.020	0.151	-0.083	0.037	0.071	-0.026	-0.062	0.154	-0.169	-0.108	-0.090
17	0	0	0	-0.064	-0.185	-0.013	-0.438	-0.251	-0.238	0.128	-0.731	0.151	-0.055	-0.150	0.208	-0.190	0.491	-0.201	0.029	-0.158

Para-meter	Bester Funktions-wert $f(x_*)$	Funktions-auswertung	Laufzeit [min]
20	$8.5931 \cdot 10^{-2}$	201	57
17	$1.3860 \cdot 10^{-1}$	201	57

Tabelle A.19: Resultate der Coordinate-Descent Methode, 20-dimensionale Testproblem

	x_*																			
	x_1	y_1	z_1	α_1	β_1	x_2	y_2	z_2	α_2	β_2	x_3	y_3	z_3	α_3	β_3	x_4	y_4	z_4	α_4	β_4
Startwert	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Pad 1	0.724	0.359	-0.050	0.000	0.106	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Pad 2	0.724	0.359	-0.050	0.000	0.106	0.127	0.032	-0.212	0.271	0.254	0	0	0	0	0	0	0	0	0	0
Pad 3	0.724	0.359	-0.050	0.000	0.106	0.127	0.032	-0.212	0.271	0.254	0.023	0.034	-0.072	-0.409	-0.001	0	0	0	0	0
Pad 4	0.724	0.359	-0.050	0.000	0.106	0.127	0.032	-0.212	0.271	0.254	0.023	0.034	-0.072	-0.409	-0.001	-0.011	0.193	-0.029	-0.539	-0.079
Pad 1	0.726	0.271	-0.061	-0.299	-0.006	0.127	0.032	-0.212	0.271	0.254	0.023	0.034	-0.072	-0.409	-0.001	-0.011	0.193	-0.029	-0.539	-0.079
Pad 2	0.726	0.271	-0.061	-0.299	-0.006	0.135	0.080	-0.183	0.377	0.105	0.023	0.034	-0.072	-0.409	-0.001	-0.011	0.193	-0.029	-0.539	-0.079
Pad 3	0.726	0.271	-0.061	-0.299	-0.006	0.135	0.080	-0.183	0.377	0.105	0.021	0.024	-0.071	-0.060	-0.009	-0.011	0.193	-0.029	-0.539	-0.079
Pad 4	0.726	0.271	-0.061	-0.299	-0.006	0.135	0.080	-0.183	0.377	0.105	0.021	0.024	-0.071	-0.060	-0.009	-0.031	0.092	-0.093	-0.158	0.003

	Bester Funktionswert $f(x_*)$	Funktions-auswertung
Startwert	2.9121	1
Pad 1	$7.8394 \cdot 10^{-1}$	25
Pad 2	$3.0889 \cdot 10^{-1}$	25
Pad 3	$1.6702 \cdot 10^{-1}$	25
Pad 4	$1.3827 \cdot 10^{-1}$	25
Pad 1	$1.3032 \cdot 10^{-1}$	25
Pad 2	$1.1501 \cdot 10^{-1}$	25
Pad 3	$1.0767 \cdot 10^{-1}$	25
Pad 4	$8.0133 \cdot 10^{-2}$	25

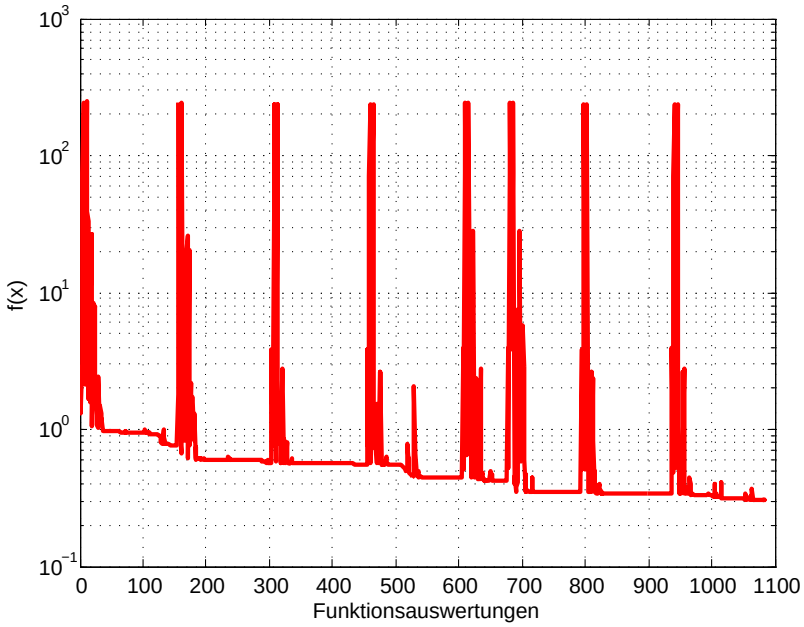


Abbildung A.1: Konvergenz für das Coordinate-Descent Verfahren in rad

A.8 Lösungen der realen Messung

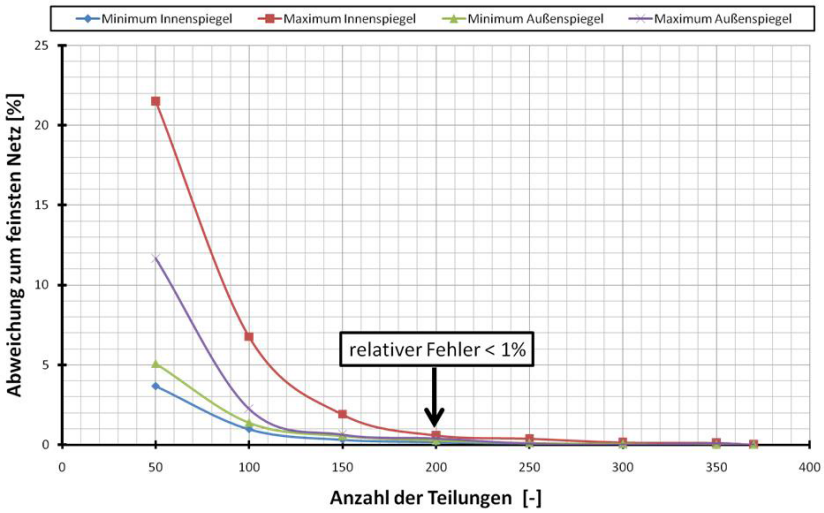


Abbildung A.2: Prozentualer Fehler unterschiedlicher Netzwqualitäten [Kle 11]

A.8.1 Lösungen des Multilevelansatzes

Tabelle A.20: Translationen und Rotationen der Pads

	Pad 1	Pad 2	Pad 3	Pad 4
x [mm]	0	0.8301	0.5412	0.3802
y [mm]	0	−0.2498	0.2346	−0.0339
z [mm]	0	2.3164	1.4593	−0.2645
α_x [mrad]	−0.0026	1.2616	−1.2455	−1.2938
α_y [mrad]	2.0531	1.1084	1.5845	−3.2066
α_z [mrad]	2.0593	−0.5129	0.6359	1.1622

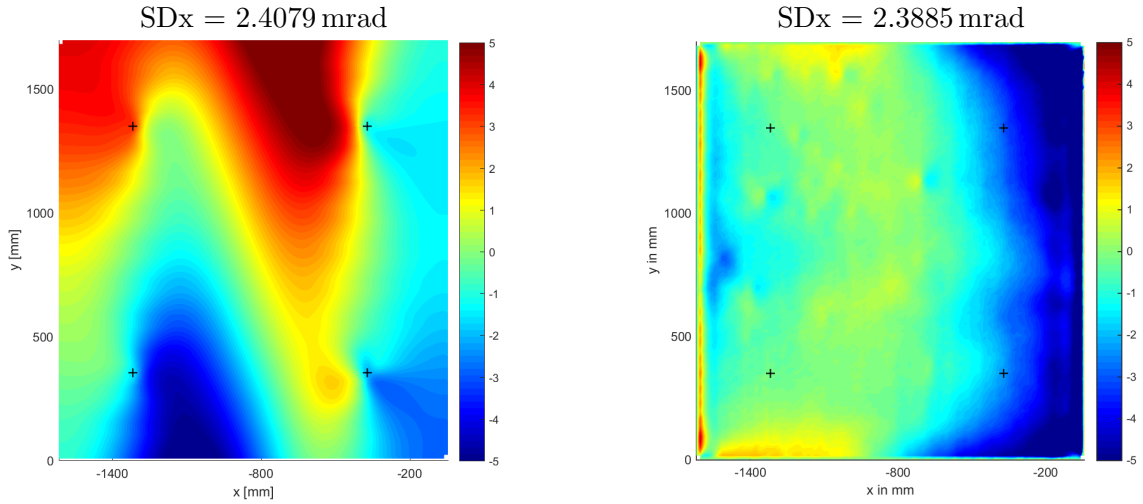


Abbildung A.3: Slope Deviation Plots des Optimierungsmodells (links) und herstellungsbedingte Formabweichungen (rechts) der Multileveloptimierung