

# **Entwicklung eines Systems zur Entfernungsabschätzung für Phasen basiertes UHF RFID Tracking durch Verwendung evolutionärer Berechnungsverfahren**

Masterthesis eingereicht zur Erfüllung  
der Anforderungen zum Erwerb des akademischen Grades  
Master of Science der Medizintechnik

Erstellt von  
**Christoph Gnip**

Fachbereich Elektrotechnik und angewandte Naturwissenschaften  
Westfälische Hochschule

September 2013



## Master's Thesis

Titel: Entwicklung eines Systems zur Entfernungsabschätzung für Phasen basiertes UHF RFID Tracking durch Verwendung evolutionärer Berechnungsverfahren

Title: Development of a Distance Estimation System for Phase-Based UHF RFID Tracking by Utilizing Methods of Evolutionary Computation

University: Westphalian University of Applied Sciences  
Department Electrical Engineering and Applied Sciences  
Neidenburger Str. 43  
45897 Gelsenkirchen  
Germany

In Cooperation with: Amedo Smart Tracking Solutions GmbH  
Universitätstraße 142  
Bochum

Author: Christoph Gnip  
Luggendelle 28  
48954 Gelsenkirchen  
Germany

Matrikelnumber: 200720362

Supervisor: Prof. Dr. Frank Bärmann  
Co-supervisor: Dipl.-Ing. Volker Trösken

## **Erklärung**

Hiermit erkläre ich, dass ich die Vorliegende Arbeit selbstständig und lediglich mit den angegebenen Hilfsmittel verfasst habe...

## Dank und Anerkennung

Danke...

# Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>1</b>
1.1. Allgemein . . . . .	1
1.2. Motivation . . . . .	3
1.3. Anforderungen . . . . .	5
1.4. Ziel und Herangehensweise . . . . .	6
<b>2. Grundlagen</b>	<b>7</b>
2.1. Mathematische Voraussetzungen . . . . .	7
2.1.1. Kondition . . . . .	7
2.1.2. SVD . . . . .	8
2.2. Optimierung . . . . .	9
2.2.1. Objektfunktion . . . . .	9
2.2.2. Arten von Optima . . . . .	10
2.3. Auffinden von Minima . . . . .	11
2.3.1. Optimierungsräume . . . . .	12
2.4. Evolutionäre Strategien . . . . .	13
2.4.1. Evolutionsstrategien - Grundlagen . . . . .	13
2.4.2. Strategien mit mehreren Populationen . . . . .	17
2.5. Covariance Matrix Adaption - Anpassung der Kovarianzmatrix . . . . .	18
2.6. Technische Voraussetzungen . . . . .	19
2.6.1. Positionsgenauigkeit auf Funk basierender Verfahren . . . . .	20
2.6.2. RFID . . . . .	20
2.6.3. PRPS-Messsystem . . . . .	22
2.6.4. Phase und Wellenzahl . . . . .	23
<b>3. Hauptteil</b>	<b>25</b>
3.1. Vorüberlegung zur Komplexität . . . . .	25
3.2. Entwicklung des Modells . . . . .	28
3.3. Antennen Permutationen . . . . .	35
3.4. Erweiterte Betrachtung der Kondition . . . . .	36
3.4.1. Weitere Anwendung der Konditionszahl . . . . .	38
3.5. Einsatz des Modells . . . . .	38
3.6. Betrachtung der Komplexität . . . . .	38
3.7. Realisierung der Kalibrierung . . . . .	42
3.7.1. Implementation . . . . .	42
3.7.2. Ergebnis . . . . .	43
3.8. Software . . . . .	51
3.8.1. Shark . . . . .	51

---

3.8.2. Implementation . . . . .	51
3.8.3. Paralleler Ablauf . . . . .	56
3.8.4. Schnittstellen für Dateneingabe . . . . .	57
3.8.5. Ablaufdiagramme . . . . .	58
<b>4. Ergebnisse</b>	<b>61</b>
4.1. Ergebnisse . . . . .	61
<b>5. Diskussion</b>	<b>63</b>
<b>6. Schluss</b>	<b>65</b>
6.1. Ausblick . . . . .	65
<b>A. Abbildungen</b>	<b>67</b>
A.1. Messaufbauten . . . . .	67
<b>B. Quellcodeauszüge</b>	<b>69</b>
B.1. ObjectiveFunktion - Evolutionary Calibration Header . . . . .	69
B.2. ObjectiveFunktion - Evolutionary Calibration Source . . . . .	72
<b>C. Gnuplot Skripte</b>	<b>73</b>
C.1. Boxplot . . . . .	73
C.2. Lineplot . . . . .	77
C.3. Scatterplot . . . . .	80
<b>D. Fitness Plots</b>	<b>85</b>

# Abbildungsverzeichnis

1.1.	Anforderungsspinne . . . . .	6
2.1.	Übersicht numerische Verfahren . . . . .	10
2.2.	Übersicht nichtlinearer Optimierungsstrategien . . . . .	14
2.3.	Ablauf Evolutionsstrategie . . . . .	15
2.4.	Konzept direkter Optimierung mittels CMA-ES . . . . .	19
2.5.	Beispiele für Transponder und Lesegeräte . . . . .	21
2.7.	PRPS der amedo GmbH . . . . .	22
2.8.	Messaufbau der Amedo GmbH . . . . .	23
2.9.	Zusammenhang Wellenlänge - Wellenzahl . . . . .	23
3.1.	Profil einer Phasenmessung . . . . .	25
3.2.	Reale Messwerte visualisiert . . . . .	27
3.3.	Normierte Messwerte von Kalibriermessung . . . . .	28
3.4.	Antennen-Szene mit einem Tag . . . . .	29
3.5.	Ablauf libPermuataate . . . . .	36
3.6.	Ergebnisse der Konditionsanalyse alle Permutationen . . . . .	37
3.9.	Fitness Ebenen Heatmap . . . . .	40
3.10.	Fitness Ebenen Heatmap, vergrößert . . . . .	41
3.11.	Übrige Ebenen für Antenne 1 . . . . .	41
3.13.	Ablauf der Kalibrierung . . . . .	44
3.14.	Ablauf der libCalibration . . . . .	45
3.15.	Box-Plot der Endergebnisse der Kalibrierung . . . . .	46
3.16.	Linien-Plot der Endergebnisse der Kalibrierung . . . . .	47
3.17.	Kalibrierung Scatter-Plot . . . . .	48
3.18.	Statistisch verteilte Ergebnisse der Kalibrierung mittels ES . . . . .	49
3.19.	Visualisierung des Kalibrierendergebnis . . . . .	50
3.20.	Kalibrierwerkzeuge . . . . .	50
3.22.	Ablauf Programmstart . . . . .	58
3.23.	Finde Modelllösung . . . . .	59
3.24.	Modelllösung bestimmen . . . . .	60
6.1.	Anforderungsspinne . . . . .	66
A.1.	PRPS-Kalibiersystem . . . . .	67
A.2.	Übersicht Kalibrieraufbau . . . . .	68



# Tabellenverzeichnis

1.1.	Anforderungen Trackingsysteme . . . . .	2
1.2.	Übersicht Navigationsverfahren . . . . .	2
3.1.	Parameter der Fitness Ebenen . . . . .	39
3.2.	Finale Antennen Koordinaten . . . . .	43
4.1.	Randbedingungen für Optimierung . . . . .	61



# Listings

3.1.	Quellcodeschnipsel für die Deklaration einer Objektfunktion . . . . .	52
3.2.	Quellcodeschnipsel für die Implementation einer Objektfunktion für verschiedene Optimizer in Shark . . . . .	54
3.3.	Quellcodeschnipsel für die Implementation einer Objektfunktion für verschiedene Optimizer in Shark . . . . .	55
3.4.	Quellcodeschnipsel für die Implementation einer Objektfunktion für verschiedene Optimizer in Shark . . . . .	55
3.5.	Quellcodeschnipsel für die Implementation einer Objektfunktion für verschiedene Optimizer in Shark . . . . .	56
3.6.	Erstellen von mehreren Threads bei gleichzeitiger Übergabe der Parameter . . . . .	56
3.7.	Abfragen der Asynchronen Ergebnisse mit Hilfe des auto-Datentyp. Ausgabe des Ergebnis in einer Datei. . . . .	57
C.1.	Gnuplot Boxplot-Skript . . . . .	73
C.2.	Gnuplot Lineplot-Skript . . . . .	77
C.3.	Gnuplot Scatterplot-Skript . . . . .	80

## Verwendete Abkürzungen

<b>ES</b>	Evolutionäre Strategie ( <i>Evolutionary Strategy</i> )
<b>CMA-ES</b>	Covariance Matrix Adaption - Evolutionary Strategy
<b>C++11</b>	Programmiersprache C++ in der Version 11
<b>MRT</b>	Magnetresonanztomografie
<b>RFID</b>	Radio-Frequency Identification
<b>LOS</b>	Line of Sight
<b>CSV</b>	Comma seperated Values
<b>Rö</b>	Röntgen
<b>CT</b>	Computertomografie
<b>EM</b>	Elektromagnetismus
<b>TAG</b>	Transponder/ Receiver für Funk Kommunikation
<b>Tracking</b>	Positionsbestimmung
<b>PRPS</b>	Passiv RFID Positioning System - Produkt der amedo GmbH
<b>TOF</b>	Time Of Flight
<b>PD</b>	Phasendifferenz
<b>RSSI</b>	Indikator für die empfangene Signalstärke (Received Signal Strength Indication)
<b>CMake</b>	Cross-Platform make; Werkzeug für Erstellung von Software
<b>ETSI</b>	European Telecommunications Standards Institute

**Verwendete Symbole**

<b>A</b>	Matrizen werden mit fetten Großbuchstaben notiert
<b>b</b>	Vektoren werden mit fetten Kleinbuchstaben notiert
<b>0</b>	Nullvektor
<i>k</i>	ist der Index der Antennen im Aufbau verwendeten Antennen
$r_k$	Abstand vom Tag zur indizierten Antenne
$d_{k0}$	Abstand zur Landmarke (Index 0) zur indizierten Antenne
$\mu$	Eigenwert; Es wird von dem gebräuchlicheren Symbol $\lambda$ abgewichen, um Mehrdeutigkeiten im Rahmen der Arbeit zu vermeiden
$(\mu, \lambda)$	"Komma"- Evolutionsstrategie
$(\mu + \lambda)$	"Plus"- Evolutionsstrategie
$\varrho$	Phase



# 1. Einleitung

## 1.1. Allgemein

Mit der Entwicklung der minimal-invasiven Chirurgie, einer Operationsmethode bei der durch sehr kleine Einschnitte in den Körper mit besonders filigranen Operationsinstrumenten operiert wird, verändert sich die Art Operationen durchzuführen grundlegend. Eingriffe können schneller, schonender und effizienter durchgeführt werden. Möglich wird diese Entwicklung durch eine Vielzahl neuartiger technischer Systeme. Die Vorteile gegenüber herkömmlichen Operationstechniken begründen die weite Verbreitung und den häufigen Einsatz der minimal-invasiven Techniken.

Mit fortschreitender Miniaturisierung der Instrumente und der Verlagerung des Operationsgebietes in den Patienten geht die optische Kontrolle über das Operationsgebiet sowie Instrumentarium verloren. Diese ist unabdingbar für einen Erfolg der Operation und die Information (z.B. Lage der Instrumente) müssen dem Operierenden jeder Zeit zur Verfügung stehen. Eine Übersicht über das Operationsgebiet im Inneren des Körpers kann durch Kameratechniken (Endoskope) gewonnen werden, allerdings wird dafür viel Platz benötigt. Die Kenntnis über den Aufenthaltsort und die Lage des Instrumentariums ist, gerade bei schwierigen Operationen, unabdingbar. Um an diese Informationen zu gelangen ist es Stand der Technik, bildgebende Verfahren intraoperativ, d.h. während der Operation, Bildvolumina zu generieren und auszuwerten.

Beispielsweise werden bei kardiologischen Interventionen (z.B. Platzierung eines Stents durch die Arteria iliaca interna<sup>1</sup> in den Coronargefäßen des Herzens) permanente Lagekontrolle der Katheter mittels Röntgentechnik durchgeführt. Dabei wird der Patient und das Operationsteam permanent ionisierender Bestrahlung ausgesetzt. Es können auch Bilder durch Magnetresonanztomografie oder durch andere bildgebende Verfahren erzeugt werden. Die Gewinnung dieser Bilddaten ist schwierig (MRT) oder nicht gut geeignet (Ultraschall) ist. Ein weiteres Erschwernis ist, dass der Patient dafür häufig mitsamt den Instrumenten umgelagert werden muss. Das Umlagern bringt weitere Risiken mit sich und ist mit weiterem Aufwand verbunden. Besonders könne die so gewonnenen Informationen nicht als genau angenommen weden.

Eine Lösung für diese Probleme bringen sog. Trackingsysteme. Diese sind in der Lage die Position, z.B. eines Instrumentes, zu ermitteln und stellen die benötigten Informationen für den Arzt zur Verfügung. Dabei kommen bei den verfügbaren Systeme unterschiedliche physikalische Prinzipien zu Einsatz, die verschiedene Vor- und Nachteile bieten.

---

<sup>1</sup>innere Beckenarterie- Standardzugang für diese Art von Operationen

Die Anwendung solcher Systeme erlaubt außerdem eine softwaregestützte Planung und assistierte Durchführung der Operation. Die Kombination dieser Techniken wird Navigation genannt. Die Möglichkeit der Planung und Kontrolle macht diese Systeme im Zuge der stets steigenden Ansprüche an das Qualitätsmanagement interessant. Die Anforderungen die vom Anwender im klinischen Alltag an die Systeme gestellt werden sind:

- Gute Genauigkeit
- Hohe Verfügbarkeit
- Leichte Bedienbarkeit
- Einfache Einbindung Workflow
- Geringe Kosten
- Sicherheit

Tabelle 1.1.: Anforderungen an ein medizintechnisches Messsystem. Nicht vollständig.

Die Anforderungen an ein solches System sind somit sehr hoch. Sie müssen über eine entsprechende technische Leistungsfähigkeit verfügen (Genauigkeit, Verfügbarkeit etc.) und gleichzeitig muss der Umgang mit ihnen leicht sein und dürfen keine hohen Kosten für Anschaffung und Betrieb generieren.

### Stand der Technik

Es befinden sich Trackingsysteme unterschiedlicher Hersteller am Markt. Sie beruhen auf unterschiedlichsten Messprinzipien und unterliegen den daraus resultierenden Limitierungen. Die wichtigsten Technischen Unterschiede sind im Folgenden tabellarisch zusammengefasst:

Arbeitsweise	Optisch	Magnetisch	Ultraschall	Funk (UHF)
Genauigkeit	gut	ausreichend	gut	sehr gut <sup>2</sup>
Frequenz	mittel	hoch	gering	hoch
Volumen	mittel	klein	mittel	groß
LOS	Ja	Ja	Nein	Ja
IV <sup>3</sup>	Nein	Nein	Nein	Ja

Tabelle 1.2.: Grobe Übersicht und Einteilung verschiedener Navigationsverfahren anhand ihres physikalischen Messprinzips.

Die Tabelle 1.2 teilt die unterschiedlichen Systeme anhand ihres physikalischen Messprinzips ein. Herausgestellt werden vor Allem die wesentlichen Messparameter der betreffende Aspekte der Verfahren. Aus der Auflistung lassen sich Vor. und Nachteile ableiten.

Das größte Problem ist das eine direkten Sicht auf die Objekte vorausgesetzt wird,

Line of Sight (LOS) genannt. Dem sog. LOS-Problem unterliegen fast alle Verfahren, die ein großes Messvolumen abdecken. Auf Funk basierende Methoden erfahren dieses Problem nicht, ihre Wechselwirkungsmechanismen erlauben die Durchdringung von Materie. Andere Wechselwirkungen machen diesen Verfahren zu schaffen und verursachen Schwierigkeiten. Der größte Vorteil des auf Funk basierenden RFID-Verfahrens ist es verschiedene Objekte von einander zu unterscheiden, bzw. sogar zu identifizieren. Die Genauigkeit (im technischen Sinne: hohe Präzision plus hohe Richtigkeit) der Messung ist bei allen Verfahren mindestens ausreichend.

Diese Anforderungen allein sind eine große Herausforderung an die Technik. Hinzukommen weitere Anforderungen, die sich z.B. aus dem Ablauf einer Intervention ergeben. Ein System muss eine einfache Integrationsmöglichkeit in diesen Arbeitsablauf bieten. Zusätzlich dürfen durch die Anschaffung und das Material keine großen Kosten entstehen.

## 1.2. Motivation

Bestimmung der Position (im Folgenden "Tracking" genannt) mittels RFID ist eine vielversprechende Technik und konkurrierenden Verfahren in viele Punkten überlegen, vgl. 1.2. Dabei stehen zwei Unterscheidungsmerkmale heraus:

1. Es wird keine LOS benötigt
2. Separation und Identifikation mehrerer Objekte

Die Vorteile lassen sich aus dem zugrunde liegende physikalischen Messprinzip ableiten. Es werden elektromagnetische Signale ausgewertet, die anderen Wechselwirkungen unterliegen und in der Lage sind Materie zu durchdringen. Insbesondere im Vergleich mit optischen Verfahren ist die auf Funk basierende RFID damit überlegen. Die Eigenschaft Materie zu durchdringen erlaubt es Objekte im Patienten zu lokalisieren. Entsprechende Untersuchungen über die erreichbare Positionsgenauigkeit dieser Verfahren im Körper sind vielversprechend. [23]

Es können mehrere Objekte von einander unterschieden und identifiziert werden. Man kann zusätzliche Informationen auf den Objekten ablegen und abfragen. Durch das einfache Anbringen von RFID-Tags unterschiedlicher Bauarten kann (siehe 2.6a) nahezu jeder Gegenstand oder Person einem Tracking unterzogen werden. Dadurch wächst das Anwendungsspektrum weiter. Besonders das Auslesen von zusätzlichen Informationen ist mit keiner der anderen Technologien möglich.

Auf Funk basierende Verfahren bieten zudem ein sehr gute Ortsauflösung. Diese ist essentiell für eine sichere Positions- und Lagebestimmung von Objekten. Die Auflösung ist jedoch abhängig von dem Messprinzip und wird in Abschnitt 2.6.1 genauer beschrieben. Das von dem Messsystem der amedo GmbH verwendete Verfahren basiert auf der Messung der Phasendifferenz der Antwort eines Objekts. Aus

den dort aufgeführten Gründen verwendet das PRPS der amedo GmbH eine Phasendifferenzmessung. Die Phasenlage ist direkt proportional zu einer Entfernung. Die Messung erreicht in der Theorie eine sehr gute Auflösung ( $\leq 1 \text{ mm}$ ), sie ist jedoch nicht eindeutig (siehe 2.6.2). Das Problem kann umgangen werden, indem man ein ganzzahliges Vielfaches der Wellenlänge auf das Messergebnis aufaddiert. Man erhält die sog. *Wellenzahl*<sup>4</sup>. Ist diese für alle Objekte und allen Antennen bekannt kann die Postion sicher bestimmt werden.

Die Annahme, dass die Wellenzahl für alle Zeiten  $t$  bekannt ist, ist naiv und höchstens unter Laborbedingungen richtig. In der Praxis müssen hier starke Einschränkungen, aufgrund der komplexen Wechselwirkungen auf EM-basierende Verfahren, gemacht werden. Die Betrachtung der Komplexität wird in Abschnitt 3.1 und 3.6 behandelt. Kann ein Objekt für kurze Zeit nicht abgefragt werden ist sofort ersichtlich, dass die Postion nicht mehr bestimmt werden kann. Die Wellenzahl(en) muss neu ermittelt werden.

Bisherige Ansätze die Wellenzahl nach Verlust des Objektes zu ermitteln basieren häufig auf Methoden der Statistik. Diese scheitern an der Komplexität des Problems oder benötigen sehr aufwändige Messreihen mit großer Anzahl an Messpunkten [3]. Eine weitere Methode die einen Schätzwert auf Basis des RSSI-Wertes berechnet wurde in vielen Ansätzen implementiert, z.B. [36]. Die Praxistauglichkeit dieser Methoden ist limitiert, die Abschätzung anhand des RSSI-Wertes funktioniert im Labor ausreichend gut, auch hier kommt es zu komplexen Wechselwirkungen. Ein weiteres Problem ist ihr große Anzahl an Wellenzahlen die bei Messaufbau vorkommen, die ein großes Volumen abdecken. Der Zusammenhang wird mit der Herleitung der Wellenzahl später beschrieben.

Zusammenfassend lassen sich über das Problem folgende Aussagen treffen. Das Problem ist:

1. Sehr komplex
2. Hochdimensional
3. Nicht linear<sup>5</sup>
4. Ohne analytische Lösung

Traditionell werden Probleme mit diesen Eigenschaften durch Methoden der Stochastik, Optimierung oder des maschinellen Lernens behandelt. Eine vollständige Übersicht und Abgrenzung der verschiedenen Gebiete ist im Rahmen dieser Arbeit nicht möglich bzw. sinnvoll. Eine Übersicht findet ist in Abbildung ?? zu finden.

---

<sup>4</sup>Diese ist nicht identisch mit der in der Physik gebräuchlichen Wellenzahl zur Beschreibung der Eigenschaften einer Welle.

<sup>5</sup>Positionsbestimmung aus superposition von EM-Wellen

Ein Teilgebiet der Optimierung stellen evolutionäre Berechnungsverfahren dar. Dies sind eine Klasse von Algorithmen, die sich für komplexe Problemfälle eignen. Sie stellen kaum Forderungen an die mathematische Formulierung des Problems, wie z.B. Stetigkeit, vollst. Differenzierbarkeit etc. Unter dem Begriff der Evolutionären Berechnungsverfahren fallen viele verschiedene Techniken, eine Übersicht über die Teilgebiete gibt die Abbildung ???. In dieser Arbeit wird das Teilgebiet der Evolutionären Optimierung verwendet um die Problemstellung zu lösen. Später wird diese Klasse von Verfahren ausführlich dargestellt.

Es sollte in jedem Fall möglich sein eine Lösung über diesen Ansatz zu finden. Vorüberlegungen und Machbarkeitsstudien wurden vom Institut für Neuroinformatik (INI) an der Ruhr-Universität Bochum angestellt. [41, 26]. An dem Institut wird seit mehreren Jahren an dieser Art von Algorithmen geforscht und unterschiedlichste Problemstellungen sind mit diesen Algorithmen gelöst worden, bspw. [33, 43]

In dieser Arbeit soll mittels evolutionärer Optimierung das beschriebene Problem gelöst werden. Im Endergebnis soll dabei eine Abschätzung der Entfernung zu einem Referenzpunkt möglich sein. Darüber lässt sich im Anschluss die Wellenzahl ermitteln.

### 1.3. Anforderungen an die Lösung

Aus den bisher vorgestellten Überlegungen können nun folgende Anforderungen abgeleitet werden:

1. Lösung muss schnell (ideal < 1 Sekunde) gefunden werden
2. Unabhängigkeit von Stütz- und Kalibrierpunkten
3. Eindeutigkeit der Lösung
4. Eignung für ein großes Messvolumen
5. Nahtlose Integration in das bestehende Software Ökosystem
6. Stand der Softwaretechnik entsprechend

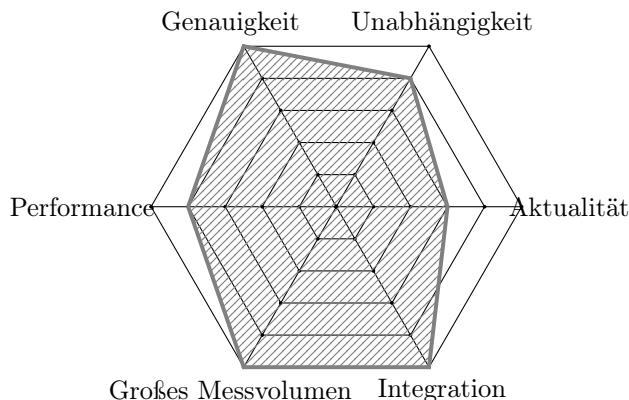


Abbildung 1.1.: Grafische Übersicht der Anforderungen an das System

## 1.4. Ziel und Herangehensweise

Das Ziel der Arbeit ist die Entwicklung eines Systems zur Abschätzung der Position eines Tags. Das Auffinden der Lösung soll die oben abgeleiteten Anforderungen erfüllt. Die Ermittelung einer korrekten Lösung ist jedoch das Wichtigste. Das System wird im Kern die Lösung über ein numerisches Optimierungsverfahren finden, im speziellen kommt das sog. '*Covarianz Matrix Adaption - Evolutionary Strategy*' (CMA-ES) zum Einsatz. Bei diesem Verfahren handelt es sich um ein stochastische, Ableitungsfreies Verfahren, dass für nicht lineare, nicht konvexe, kontinuierliche Probleme geeignet ist. Dazu wird zuerst ein Modell entworfen werden, dass sich für einen Einsatz in diesem Verfahren eignet. Das eingesetzte Lösungsverfahren stellt praktisch keine Anforderungen an ein solches Modell. Daher soll es mit möglichst wenig Annahmen/ Einschränkungen auskommen und dennoch ein relativ sicheres, reproduzierbares Ergebnis liefern. Weiterhin soll eine eine Integration der Lösung in das Software-Ökosystem der amedo GmbH erfolgen. Ferner sollen Methoden in weiteren Projekten zum Einsatz kommen, beider Umsetzung ist auf eine größtmögliche Wiederverwendbarkeit zu achten. Es werden verschiedene Implementationen des CMA-ES-Algorithmus recherchiert, verglichen und die geeignetste gewählt. Das System soll unmittelbar in den Produkten der amedo GmbH zum Einsatz kommen können, daher wird eine entsprechende Schnittstelle für andere Software implementiert werden. Im Rahmen dieser Arbeit wird eine Methode entwickelt werden, um die Position von frei im Raum angeordnete Antennen zu ermitteln und dem Messaufbau zu kalibrieren.

## 2. Grundlagen

Im Rahmen dieses Kapitels werden die Grundlagen für diese Arbeit beschrieben. Zielgruppe dieser Arbeit sind fachkundige Personen die ein gewisses Grundwissen/-verständnis mitbringen oder solche die es werden wollen. Die Ausführungen werden in der für das Verständnis dieser Arbeit angebrachten Tiefe beschrieben. Da das Spektrum der Themen zu groß ist, kann nicht auf Alles im Detail eingegangen werden. Allgemeine Zusammenhänge und Techniken, denen einen großen Stellenwert in dieser Arbeit zukommt, werden zusammengefasst präsentiert. Für detaillierte Beschreibungen wird stets auf entsprechende Fachliteratur verwiesen.

### 2.1. Mathematische Voraussetzungen

Dieser Abschnitt behandelt die mathematischen Voraussetzungen für diese Arbeit.

#### 2.1.1. Kondition

Gegeben ist ein lineares Gleichungssystem der Form:

$$\mathbf{Ax} - \mathbf{b} = \mathbf{0}$$

Eine numerische Lösung führt in der Regel zu einer von  $\mathbf{0}$  verschiedenen Lösung (insbesondere bei überbestimmten Systemen), so das wir:

$$\mathbf{A}\tilde{\mathbf{x}} - \mathbf{b} = \mathbf{r}$$

schreiben. Man nennt  $\mathbf{r}$  den Residuumvektor. Es ist offensichtlich, dass ein kleines Residuum nicht hinreichend ist um von einem kleinen relativen Fehler auszugehen. Weiter folgt aus  $\mathbf{Ax} - \mathbf{b} = \mathbf{0}$  und  $\mathbf{A}\tilde{\mathbf{x}} - \mathbf{b} = \mathbf{r}$ , dass

$$\mathbf{A}\Delta\mathbf{x} = \mathbf{r}$$

und damit:  $\|\mathbf{b}\| = \|\mathbf{Ax}\| \leq \|\mathbf{A}\|\|\mathbf{x}\|$ ,  $\|\Delta\mathbf{x}\| = \|-\mathbf{A}^{-1}\mathbf{r}\| \leq \|\mathbf{A}^{-1}\|\|\mathbf{r}\|$ . Wir können nun für den relativen Fehler schreiben:

$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \frac{\|\mathbf{A}^{-1}\|\|\mathbf{r}\|}{\|\mathbf{b}\|/\|\mathbf{A}\|} = \|\mathbf{A}\|\|\mathbf{A}^{-1}\| \frac{\|\mathbf{r}\|}{\|\mathbf{b}\|}$$

Der Term  $\|\mathbf{A}\|\|\mathbf{A}^{-1}\| := \text{cond}(\mathbf{A})$  heißt Konditionszahl. Auch der Begriff Konditionsmaß ist gebräuchlich und bezieht sich auf die gewählte Matrixnorm. Es kann gezeigt werden, dass  $\text{cond}(\mathbf{A}) \gg 1$  für eine schlechte Konditionierung der Matrix

steht. Wird im Folgenden von einer speziellen Matrixnorm gesprochen schreiben wir  $\text{cond}(\mathbf{A})$  zu

$$\text{cond}_k(\mathbf{A}) = \|\mathbf{A}\|_k \|\mathbf{A}^{-1}\|_k$$

Der Index  $k$  wird entsprechend für die verwendete Norm ersetzt. Beispielsweise ergibt sich für die Konditionszahl der Spektralnorm<sup>1</sup>:

$$\text{cond}_2(\mathbf{A}) = \|\mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2 = \sqrt{\frac{\mu_{\max}}{\mu_{\min}}}$$

Die Symbole  $\mu_{\max}$  und  $\mu_{\min}$  stehen für die Eigenwerte des Systems.

Die Konditionszahl ermöglicht eine Analyse der Güte einer Lösung, die mittels Numerischer Verfahren ermittelt wurde. Nach [18] kann man folgende Aussage über die Konditionszahl treffen:

"Wird ein lineares Gleichungssystem  $Ax = b$  mit  $t$ -stelliger dezimaler Gleitpunktarithmetik gelöst und beträgt die Konditionszahl  $\text{cond}(A) \approx 10^\alpha$ , so sind auf Grund der im allgemeinen unvermeidbaren Fehler in den Eingabedaten  $A$  und  $b$  nur  $t - \alpha - 1$  Dezimalstellen der berechneten Lösung  $\tilde{x}$  (bezogen auf die betragsgrößte Komponente) sicher."

### 2.1.2. SVD

Bei dem Verfahren der Singular Value Decomposition (oder auch Singulärwertzerlegung), kurz SVD, handelt es sich um eine Faktorisierung einer Matrix. Die Matrix wird dabei als Produkt von drei Matrizen dargestellt. Diese Matrizen enthalten die sog. Singulärwerte und können aus einer der Matrizen abgelesen werden. Die Eigenschaften des Systems sind, ähnlich den Eigenwerten, aus den Singulärwerten bestimmbar. Besonders an der SVD ist, die Existenz für jede Form von Matrix - einschließlich nicht quadratischer Matrizen.

Die SVD basiert auf folgender Theorie der linearen Algebra: Jede  $M \times N$  Matrix  $\mathbf{A}$  kann als Produkt einer  $M \times N$  Spalten-orthogonalen Matrix  $\mathbf{U}$ , einer  $N \times N$  Diagonalmatrix  $\Sigma$  mit Werten  $\geq 0$  und einer dritten adjungierten  $N \times N$ -Matrix  $\mathbf{V}^*$ , so ergibt sich:

$$\mathbf{A} = \mathbf{U} \Sigma \mathbf{V}^* = \mathbf{U} \Sigma \mathbf{V}^T \quad (2.1)$$

Ist  $\mathbf{A}$  eine reellwertige Matrix gilt:  $\mathbf{V}^* = \mathbf{V}^T$ . Die Matrix  $\Sigma$  ist im Rahmen dieser Arbeit von besonderem Interesse, denn sie enthält die Singulärwerte  $\sigma_r$ . Ihre

---

<sup>1</sup><http://de.wikipedia.org/w/index.php?title=Spektralnorm&oldid=118988565>

Gestalt ist wie folgt:

$$\Sigma = \left( \begin{array}{ccc|cc|c} \sigma_1 & & & & & \vdots \\ & \ddots & & & 0 & \dots \\ & & \sigma_r & & \vdots & \\ \hline & & & & & \\ \vdots & & & & & \vdots \\ \dots & 0 & \dots & \dots & 0 & \dots \\ & \vdots & & & & \vdots \end{array} \right)$$

$$, wobei \sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > 0$$

Da die  $\sigma_r$  der Matrix mit den Eigenwerten in Verbindung stehen, kann aus dieser Matrix die Konditionszahl bestimmt werden. Sie ist durch folgendes Verhältnis gegeben:

$$cond(\mathbf{A}) = \frac{\max(\sigma_r)}{\min(\sigma_r)} = \frac{\max(\sigma_1)}{\min(\sigma_r)} \quad (2.2)$$

Es gibt bereits viele Implementationen des Verfahrens, z.B. [28]. Diese Implementation wird durch den Erwerb der entsprechenden Lizenz im Rahmen dieser Arbeit verwendet.

Weitere Informationen zum Verfahren sind in [4, Kapitel 4.6.3] zu finden.

## 2.2. Optimierung

Dieser Abschnitt führt in die Optimierung ein und gibt einen Überblick über die Grundbegriffe, die im Rahmen dieser Arbeit verwendet werden. Tiefe Einsichten und die theoretischen Grundlagen sind z.B. in [2, 37] zu finden.

### 2.2.1. Objektfunktion

Für eine Optimierung wird eine Größe benötigt, die optimal werden soll. Dazu bedarf es einer Formulierung der Funktion die das Optimierungsziel enthält. Diese Funktion wird Objektfunktion<sup>2</sup>, Fitnessfunktion oder Zielfunktion genannt. Auch Güte- oder Qualitätsfunktion sind gebräuchlich. Es sind eine Reihe von Optimierungskriterien denkbar, z.B. können Gewicht, Größe, Fläche etc. optimiert werden. Es kann auch mehr als ein Ziel Bedingung sein, in diesem Fall spricht man von Mehrzieloptimierung<sup>3</sup>.

Damit man eine Optimierung durchführen kann, muss die Objektfunktion freie Parameter (Variablen) enthalten.

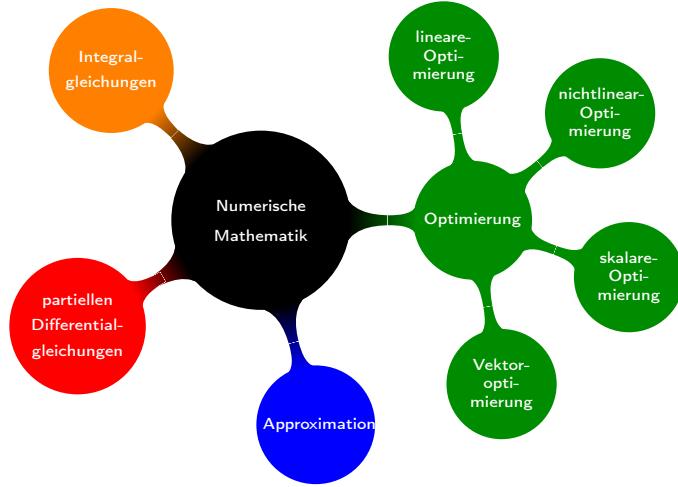
$$y = f(x), \quad x \in \mathbb{R}$$

---

<sup>2</sup>Wird in dieser Arbeit verwendet

<sup>3</sup>Keine weitere Erläuterung im Rahmen dieser Arbeit

Abbildung 2.1.: Übersicht über numerische Verfahren (unvollständig). Interessant für diese Arbeit ist der teil *nichtlineare Optimierung*. Dieser wird an andere Stelle ausführlicher gezeigt.



Diese Formulierung ist eine eindimensionale Objektfunktion. Dabei ist  $x$  der freie Parameter den man variieren kann, um ein Optimum der Funktion  $f(x)$  zu finden. In der Regel ist der Wert des erreichten Optimums nicht sehr interessant. Vielmehr ist man an dem auffinden der Optimalen Einstellung der freien Parameter interessiert.

### 2.2.2. Arten von Optima

Man kann leicht verstehen, dass es unterschiedliche Optima gibt. Es kann das z.B. das geringste Gewicht oder der größte Gewinn Ziel der Optimierung sein. Dazu notieren wir:

$$y = \min\{f(\mathbf{x})\} = -\max\{-f(\mathbf{x})\}, \quad \mathbf{x} \in \mathbb{R}^n$$

Die Gleichung drückt aus, dass ich jedes Maximierungsproblem in ein äquivalentes Minimierungsproblem überführen lässt. Daraus kann eine allg. Formulierung des Optimierungsproblem angegeben werden.

#### Allgemeine Formulierung des Optimierungsproblems

Ableitung der allgemeinen Formulierung aus den bisherigen Ausführungen. Die Funktion:

$$y = f(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n$$

sei zu optimieren. Dabei ist eine Minimierungsaufgabe gleich der Maximierungsaufgabe. Das bring die Formulierung:

$$\min\{ y(\mathbf{x}) \mid \mathbf{x} \in \mathbf{X} \} \tag{2.3}$$

In der Formulierung ist eine Einschränkung enthalten. Wir fordern, dass  $\mathbf{x} \in \mathbf{X}$  sein soll. Diese Einschränkung entstammt den Nebenbedingung des Problems. Herleitung:

$$g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, n$$

Ergibt den zulässigen Bereich:

$$\mathbf{X} = \{\mathbf{x} \in \mathbb{R}^n | g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, n\}$$

### Allgemeine von Minima

Eine Objektfunktion kann zwei Arten von Minima enthalten:

- **Lokales Minimum** - Ein zulässiger Punkt, bei dem in der Nachbarschaft keine niedrigeren Funktionswerte zu finden sind
- **Globales Minimum** - Ein zulässiger Punkt, der den geringsten Funktionswert des gesamten zulässigen Bereichs aufweist. Gleichzeitig ein lokales Minimum.

Wir haben ein mathematisches Modell der Optimierungsaufgabe erstellt. Wir können noch keine Aussage über die Komplexität des Problem treffen. Um den Schwierigkeitsgrad bestimmen zu können muss man die Zielfunktion analysieren. Eine Analyse der in dieser Arbeit verwendeten Objektfunktion wird im Hauptteil gezeigt.

Ein Optimierungsproblem mit Nebenbedingungen wird registriertes Optimierungsproblem genannt. Ohne spricht man von unregistrierten Optimierungsproblemen.

## 2.3. Auffinden von Minima

Die Kenntnis der Zielfunktion und des mathematischen Modells erlaubt die Anwendung eines Algorithmus, der das Minima bestimmt. Über gibt man die Zielfunktion und die Fragestellung an einen solchen Algorithmus berechnet dieser auf unterschiedlichen Wegen mögliche Optima. Man kann zwei Hauptklassen von Verfahren unterscheiden:

- Lineare Optimierungsverfahren
- Nichtlineare Optimierungsverfahren

Auf die linearen Verfahren wird im Rahmen dieser Arbeit nicht eingegangen. Sie eignen sich nicht für komplexe Fragestellungen. Eine Übersicht über die nichtlinearen Verfahren ist in der Abbildung 2.2

### 2.3.1. Optimierungsräume

In der Optimierung kann man verschiedene Optimierungsräume betrachten. Der häufigste und auf den alle Algorithmen Anwendbar sind ist der reellwertige oder kontinuierliche Raum. Daneben kommen Probleme vor, die auf einem ganzzahligen (diskreten) Raum ablaufen sowie gemischte Probleme. Im Folgenden wird die Modellformulierung für alle Räume besprochen.

#### Kontinuierliche Optimierung

Es wurde bereits das Modell für die kontinuierliche Optimierung besprochen siehe Gleichung 2.3. Wird im Folgenden von kontinuierlichen Problemen gesprochen, wird ein  $k$  an die Bezeichnungen angehängt.

#### Diskrete Optimierung

Darf ein Variablenvektor nur Werte einer diskreten Wertemenge annehmen, spricht man von einer diskreten Optimierung. Als Spezialfälle lassen sich eine ganzzahlige sowie eine binäre Optimierung ableiten, siehe unten. Die Definition von kontinuierlichen und diskreten Optimierungsproblem sind gleich:

$$\min\{ y(\mathbf{x}) \mid \mathbf{x} \in \mathbf{X}_D \} \quad (2.4)$$

Dabei stammt der zulässige Bereich  $\mathbf{X}_D$  aus der Menge diskreter Werte:

$$\mathbf{X}_D = \{ \mathbf{x} \in \mathbb{R}^n \mid x_i \in \mathbf{D}_i, \quad i = 1, 2, \dots, m; \quad g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, n \}$$

Für den diskreten Charakter sorgt die Auswahl von  $m$  Wertemengen aus  $\mathbb{R}$ :

$$\mathbf{D}_i = \{x_{i1}, x_{i2} \dots x_{id_i}\}, \quad \mathbf{D}_i \subset \mathbb{R}, \quad i = 1, 2, \dots, m;$$

#### Spezialfall - Ganzzahlige Optimierung

Der Spezialfall einer diskreten Optimierung ist die ganzzahlige

$$\min\{ y(\mathbf{x}) \mid \mathbf{x} \in \mathbf{X}_Z \} \quad (2.5)$$

$$\mathbf{X}_Z = \{ \mathbf{x} \in \mathbb{Z}^n \mid x_i \in \mathbf{Z}_i, \quad i = 1, 2, \dots, m; \quad g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, n \}$$

Dadurch entstehen die  $m$  Wertemengen:

$$\mathbf{Z}_i = \{-d_1^-, \dots, -1, 0, +1, \dots d_i^+\}, \quad \mathbf{Z}_i \subset \mathbb{Z}, \quad i = 1, 2, \dots, m;$$

Ein weiterer Spezialfall ist die binäre Optimierung. Die Erweiterung ist für diesen Fall trivial und wird hier nicht gezeigt.

### Gemischte Optimierung

Die auch als diskret-kontinuierliche Optimierung bekannte Methode formuliert das Modell für einen gemischten Variablenvektor. Man kann in einem solchen Fall den Variablenvektor aus zwei Teilen zusammensetzen:

$$\mathbf{x} = [\mathbf{x}_K \ \mathbf{x}_D]^T$$

Die Notation meint mit

- $\mathbf{x}_K$  den Vektor der  $m_K$  kont. Variablen und mit
- $\mathbf{x}_D$  den Vektor der  $m_D$  disk. Variablen.

Analog dazu lässt sich der Suchraum in zwei Teile Zerlegen:

$$\mathbf{x}_K \in \mathbf{X}_K \quad \text{und} \quad \mathbf{x}_D \in \mathbf{X}_D$$

Nun lautet das disk.-kont. Optimierungsproblem wie folgt:

$$\min\{ y(\mathbf{x}) \mid \mathbf{x}_K \in \mathbf{X}_K, \mathbf{x}_D \in \mathbf{X}_D \} \quad (2.6)$$

Diskret-kontinuierliche Probleme sind nur schlecht lösbar. In dieser Arbeit wird ein rein reellwertiges Problem behandelt.

## 2.4. Evolutionäre Strategien

Folgende Informationen entstammen im Wesentlichen aus [24, Kapitel 7],[4] sowie [14] und sind auf den folgenden Seiten zusammengefasst und neu arrangiert. Das erleichtert die Einarbeitung in die Thematik und das Verständnis der Arbeit. Evolutionsstrategien nach natürlichem Vorbild gehen auf die Arbeiten von RECHENBERG [31] und SCHWEFEL [35] zurück. Ihr Ansatz war von Anfang an für die Optimierung gedacht, anders als z.B. Evolutionäre Programmierung (EP). Im Gegensatz zu Genetischen Algorithmen spielt die Repräsentationsform der Daten keine Rolle. Der Variablenvektor kann sowohl diskrete als auch kontinuierliche Komponenten haben.

### 2.4.1. Evolutionsstrategien - Grundlagen

Nach dem Vorbild natürlicher Evolution entworfene stochastische Optimierungsverfahren werden als Evolutionsstrategie bezeichnet. Sie verwenden die Prinzipien der Mutation, Rekombination und Selektion analog zu der natürlichen Evolution. Die Vorgänge werden dabei abstrahiert und vereinfacht implementiert, bilden die Prozesse der Evolution jedoch nach. Durch ihre Allgemeingültigkeit haben diese Verfahren ein großes Anwendungsspektrum in Wissenschaft und Industrie. Einige Anwendungsmöglichkeiten und Beispiele sind in [24, Kapitel 11] zu finden.

Die Abstrahierung der Prozesse verwendet dabei die selbe Nomenklatur, wie im biologischen Kontext üblich. Dabei bezeichnet im Folgenden:

Abbildung 2.2.: Klassifizierung von nicht linearen Optimierungsstrategien. Die evolutionären Strategien gehören zu den ableitungsfreien Verfahren. Diese grenzen sich zu anderen Verfahren ab, da sie keine stetige Differenzierbarkeit erwarten. Das ist ein Vorteil, da damit praktisch alle Probleme gelöst werden können.

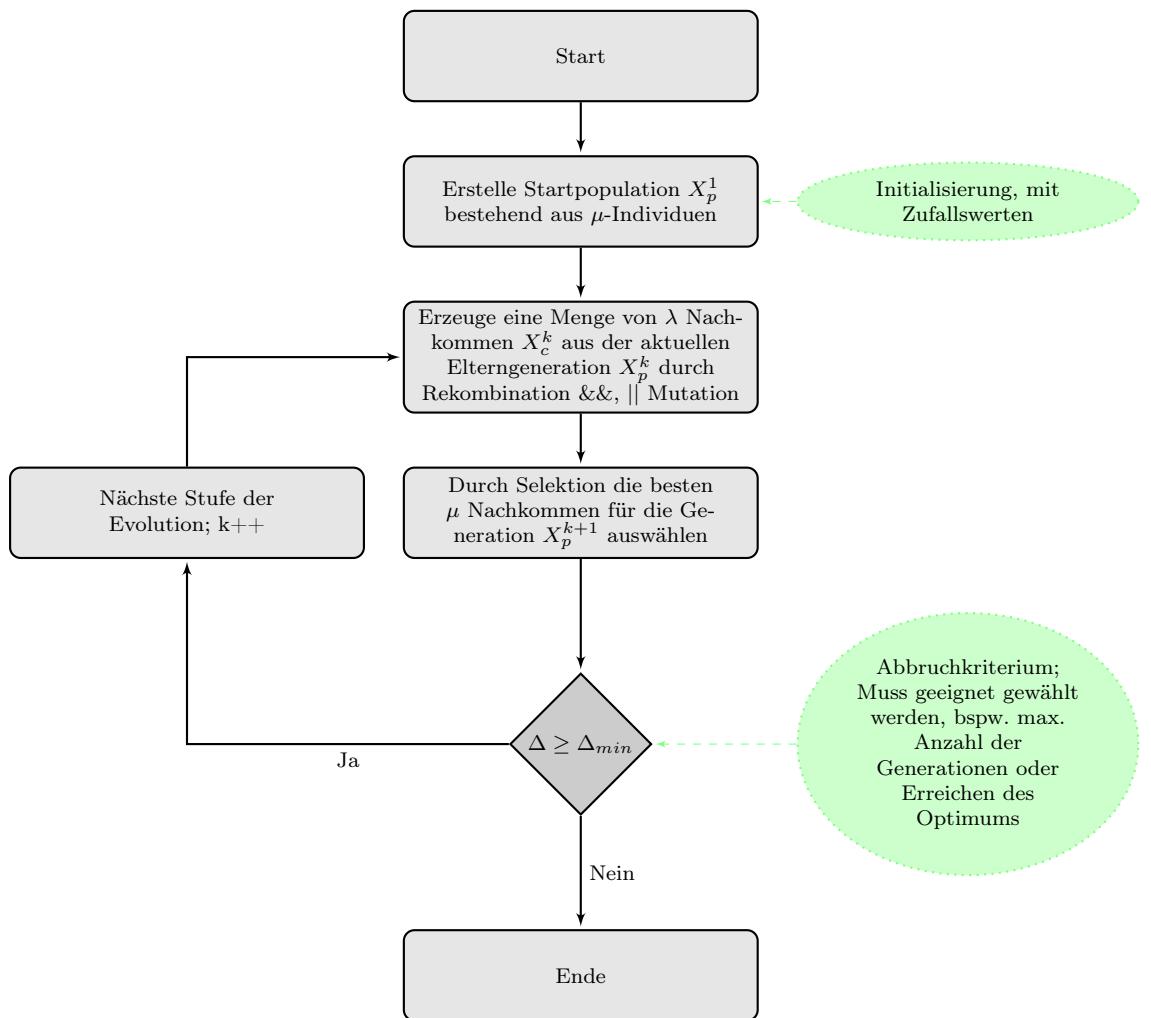


- $\mu$  die Anzahl der Eltern ( $\Rightarrow$  Größe der Population)
- $\lambda$  die Anzahl der Eltern die bei Rekombination neue Kinder erzeugt; Die Anzahl der erzeugten Nachkommen einer neuen Generation<sup>4</sup>
- $\mathbf{x}_p$  Eltern (Parent)
- $\mathbf{x}_c$  Nachkommen einer Generation (Child)
- $X_p^1$  Die Menge aller Eltern der ersten Generation  $X_p = \{\mathbf{x}_{p_1}^1, \dots, \mathbf{x}_{p_\mu}^1\}$
- $X_p^k$  Die Menge aller Eltern der k-ten Generation  $X_p = \{\mathbf{x}_{p_1}^k, \dots, \mathbf{x}_{p_\mu}^k\}$

Wir wollen nun in Abbildung 2.3 einen Blick auf den prinzipiellen Ablauf dieses Algorithmus werfen und anschließend auf die Details eingehen.

<sup>4</sup>Anmerkung: Die Verwendung des Symbols  $\lambda$  ist in diesem Kontext nicht eindeutig. Im Rahmen dieser Arbeit steht dieses Symbol auch für die Wellenlänge. In diesem Abschnitt wird jedoch weiterhin  $\lambda$  verwendet um die gleiche Nomenklatur wie bei dieser Thematik üblich zu verwenden.

Abbildung 2.3.: Der Ablauf des  $(\lambda, \mu)$ -Evolutionsalgorithmus ist in dieser Abbildung gezeigt. Dies ist die einfachste Variante der Algorithmen<sup>5</sup>. Die wesentlichen Schritte gleichen sich in den Varianten.



### Mutation

Ein Nachkomme  $\mathbf{x}_C$  wird aus seinem Elternteil  $\mathbf{x}_P$  und einer zufälligen Variation  $\mathbf{d}$  gebildet.

$$\mathbf{x}_c = \mathbf{x}_P + \mathbf{d} \quad (2.7)$$

Dabei ist  $\mathbf{d}$  ein bei jeder Mutation neu zu bestimmender  $(0, \sigma^2)$ -normalverteilte Zufallszahl  $Z(0, \sigma^2)$ :

$$\mathbf{d} = \begin{pmatrix} d_1 \\ \vdots \\ d_n \end{pmatrix} = \begin{pmatrix} Z(0, \sigma_1^2) \\ \vdots \\ Z(0, \sigma_n^2) \end{pmatrix} = \begin{pmatrix} Z(0, 1)\sigma_1 \\ \vdots \\ Z(0, 1)\sigma_n \end{pmatrix} \quad (2.8)$$

Die Normalverteilung der Variation ist nützlich, da kleine Änderungen wahrscheinlicher sind als Große. Die maximale Größe der Variation wird durch die Standardabweichung  $\sigma_i$  bestimmt. Sie steuert somit die Schrittweite von Generation zu Generation.

### Rekombination

Durch Rekombination zweier oder mehr Eltern aus der Menge aller  $\mu$ -Eltern  $X_\varrho \subset X_E$ . Die Wahl der Eltern sollte zufällig erfolgen um *Inzuchtprobleme*<sup>6</sup> zu verhindern.

Zwei Arten der Rekombination sind denkbar:

Die *intermediär Rekombination* erstellt einen Nachkommen durch das gewichtete Mittel von  $\varrho$  Eltern.

$$\mathbf{x}_c = \sum_{i=1}^{\varrho} \alpha_i \mathbf{x}_{p_i}, \sum_{i=1}^{\varrho} \alpha_i = 1, 2 \leq \varrho \leq \mu \quad (2.9)$$

Bei der *diskreten Rekombination* vom  $\varrho$ -Eltern wird die  $i$ -te Komponente  $x_{ic}$  eines Nachkommen  $\mathbf{x}_c$  mit der  $i$ -te Komponente eines zufällig gewählten Elternpunktes gleichgesetzt.

$$\mathbf{x}_{ic} = \mathbf{x}_{ip_j}, j \in \{1, \dots, \varrho\}, i = 1, \dots, n \quad (2.10)$$

### Selektion

Die durch Rekombination und/oder Mutation erzeugten Nachkommen werden in dem Schritt ausgewählt um einen Evolutionsfortschritt zu erreichen. Dies erfolgt anhand des Vergleichs mit dem Zielfunktionswert  $f(\mathbf{x})$ . Das beste Individuum, bzw. die besten, werden für die nachfolgende Generation ausgewählt. Dabei gibt es Strategien, bei denen zum einen nur die Nachkommen an der Auswahl beteiligt sind oder Andere bei denen Eltern und Kinder teilnehmen.

### Evolutionsalgorithmus

Der eigentliche Evolutionsalgorithmus ist in Abbildung 2.3 dargestellt. Er enthält im Wesentlichen die in den vorherigen Abschnitten beschriebenen Schritte. Der prinzipielle Ablauf ist für alle Evolutionsalgorithmen gleich. Eine Unterscheidung der Verfahren kann durch verschiedene Parameter beschrieben werden. Wesentlich dabei sind die Populationsgröße  $\mu$ , die Anzahl an der Rekombination beteiligten

---

<sup>6</sup>d.h. sich schlechte Ergebnisse weiter durchsetzen

Eltern  $\varrho$ , die gewählte Selektionsstrategie sowie die Anzahl der Nachkommen  $\lambda$ . Im Folgenden sind zuerst einige Beispiele für die Nomenklatur der Selektionsstrategie aufgeführt, die im Anschluss genauer beschrieben werden.

Für Strategien die nur auf Mutation für die Erzeugung von Nachkommen setzten sind folgende Nomenklaturen gebräuchlich:

- $(\mu + \lambda)$  Elternelemente werden in der Selektion berücksichtigt
- $(\mu, \lambda)$  Ausschließlich Nachkommen nehmen an der Selektion teil

Die Strategien werden Plus- bzw. Komma-Strategie genannt. bei der Plus-Strategie wird zusätzlich noch ein Gewichtungsfaktor benötigt, der das *Altern* der Eltern-generation darstellt. Dieser Mechanismus soll verhindern, dass die Eltern für alle Zeiten überlegen. Somit werden sie nach einer gewissen Anzahl an Generationen, nicht mehr berücksichtigt werden. Sie sind zu *alt*. Wird die Rekombination eingesetzt kann auch die Anzahl der beteiligten Elternelemente angegeben werden:

- $(\mu/\varrho + \lambda) \& (\mu/\varrho, \lambda)$  Angabe der Anzahl beteiligter Eltern bei der Rekombination.

Mit Hilfe der hier beschriebenen Klassifikationen werden die Algorithmen im Folgenden stets angegeben. In Abbildung 2.3 wird der Ablauf einer Optimierung mit evolutionären Verfahren dargestellt. Es wird die Komma-Strategie gezeigt, ein Struktogramm der Plus-, oder anderer Strategien ist nicht gezeigt. Die Unterschiede würden sich in dem Punkt Rekombination zeigen.

#### 2.4.2. Strategien mit mehreren Populationen

Es ist möglich die Strategien auf die Ebene von Populationen zu erweitern. Das bedeutet, man lässt ganze Populationen miteinander in Wettstreit treten und nur diejenigen überleben, die die besten Ergebnisse liefern. Das mündet in einem zweistufigen Evolutionsprozess. Man kann die Notation um diesen Umstand erweitern und erhält so:

$$[\mu_2/\varrho_2,^+ \lambda_2(\mu_1/\varrho_1,^+ \lambda_1)]$$

Sprich aus  $\mu_2$ -Elternpopulationen werden durch Rekombination mit jeweils  $\varrho_2$  Populationen,  $\lambda_2$  Nachkommenpopulationen generiert. Innerhalb der Populationen erfolgt die Optimierung anhand einer  $(\mu_1/\varrho_1 + \lambda_1)$  oder  $(\mu_1/\varrho_1, \lambda_1)$ -Strategie. Nun können nach einer bestimmten Zahl von Generationen die besten Populationen für die nächste Generation ausgewählt werden. Auch hier stehen verschiedene Auswahlkriterien zur Verfügung. Man kann z.B. die Population anhand des Zielfunktionswert des besten Individuums wählen oder den Mittelwert über alle Individuen wählen.

#### Nicht behandelte Themen

Das Gebiet der Evolutionären Optimierung/- Algorithmen ist groß. Daher wurden in diesem Abschnitt nur die für das Verständnis unbedingt notwendigen Grundlagen erläutert. Eine Reihe von Themen, die in der Evolutionären eine Rolle spielen, wurde ausgelassen. Beispielsweise:

- Schrittweitensteuerung
- Diskrete Optimierungsprobleme/ Variationsmechanismen
- Neustart mit andere Population
- Behandlung von Nebenbedingungen

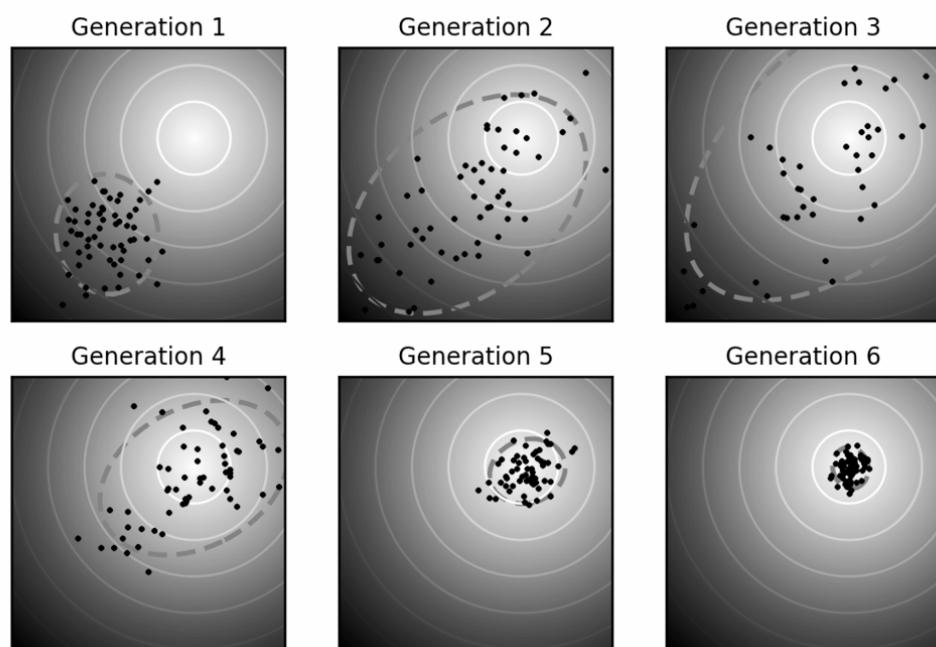
## 2.5. Covariance Matrix Adaption - Anpassung der Kovarianzmatrix

Der Folgende Abschnitt behandelt ein spezielles Verfahren der Evolutionären Optimierung. Das CMA-ES-Verfahren. Es stellt den "State of the Art"- der Evolutionären Berechnungsverfahren dar. Für spezialisierte Probleme gibt es bessere Lösungen, als allgemeiner Solver ist dieses Verfahren mehr als tauglich. Es wurde um die Jahrtausendwende entwickelt und veröffentlicht. Typische Einsatzgebiete der CMA-ES sind unregistrierte Optimierungsprobleme mit einem Suchraum zwischen drei und einhundert Dimensionen. Der Algorithmus kann eingesetzt werden wenn auf Ableitung angewiesene Verfahren (z.B. Newtonverfahren) aufgrund zerklüftete Suchräume, Rauschen, viele lokale Optima oder Ausreißern zu Fehlern neigen. Die Methode ist geeignet für nicht separierbare, schlecht konditionierte Probleme. Das Verfahren benötigt keine Gradienten, oder erfordert ihre Existenz. Dadurch kann der Algorithmus auch auf nicht stetige Probleme angewandt werden. Es existieren verschiedene Abwandlungen des Algorithmus, die für spezielle Fragestellungen geeigneter sind, und sogar eine Lösung zur Multiobjekt-Optimierung (MO-CMA-ES) wurde beschrieben [16]. Das Verfahren wird aktuell stets weiterentwickelt [10, 9]. Das Verfahren wurde in vielen Fragestellungen angewendet, eine unvollständige, etwas veraltete Übersicht ist unter [13] zu finden.

Durch die Anpassung der Kovarianz-Matrix wird eine enge Abschätzung der Konturlinien der Objektfunktion  $f$ .

Der Algorithmus steht in verschiedene Implementationen, Programmiersprachen und Umgebungen zur Verfügung [15]. Teilweise sind die Implementationen proprietär (z.B. Matlab), teilweise quelloffen. Die in dieser Arbeit zur Anwendung kommende Umsetzung ist die Shark-Library. Diese Bibliothek ist eine in *C++* geschriebene, quelloffene Software, die am Institut für Neuroinformatik der Ruhr Universität Bochum entwickelt wird. Detailliert wird Shark im Rahmen des Hauptteils in Abschnitt 3.8.1 vorgestellt.

Abbildung 2.4.: Die Abbildung zeigt sechs Lösungsschritte des CMA-ES-Verfahrens. Eingangs wird eine Population mit einer Gauß-Verteilung generiert. Diese bewegt sich in jeder Generation näher an das globale Optimum heran. Die gestrichelte Linie zeigt dabei eine Isolinie der Wahrscheinlichkeitsdichte. Wir befinden uns auf einem 2D-Problem, somit wird das vorankommen der Population über zwei  $\sigma$ 's gesteuert. Dadurch kommt die Ellipse zu Stande. Die Linie bedeutet nicht, dass nur in diesem Bereich Nachkommen erzeugt werden, siehe z.B. Generation 4. Je nach Beschaffenheit des Problems und nach Nähe zum Optimum werden die steuernden  $\sigma$ 's kleiner. [38]



## 2.6. Technisch-Physikalische Voraussetzungen

In diesem Abschnitt werden die technischen und physikalischen Grundlagen für diese Arbeit vorgestellt und das Wichtigste erörtert. Es wird auf die Besonderheiten und Merkmale des auf Funk basierenden RFID-Verfahrens eingegangen. Die anderen Trackingverfahren werden, aufgrund der Unterschiedlichkeit der Systeme wird im Rahmen dieser Arbeit nicht weiter behandelt. Es kann nicht im vollem Umfang auf die Details der Technik eingegangen werden ohne den Rahmen dieser Arbeit zu sprengen. Interessierte sei die referenzierte Literatur für eine weite Lektüre empfohlen.

### 2.6.1. Positionsgenauigkeit auf Funk basierender Verfahren

Die Positionsgenauigkeit eines auf EM basierenden Systems ist von dem Messprinzip abhängig. Dabei bieten sich im Wesentlichen drei Möglichkeiten:

Eine Laufzeitmessung des Signals kommt aufgrund der Ausbreitungsgeschwin-

- |                              |             |
|------------------------------|-------------|
| 1. Laufzeitmessung           | <b>TOA</b>  |
| 2. Messung der Signalsträrke | <b>RSSI</b> |
| 3. Phasendifferenzmessung    | <b>PD</b>   |

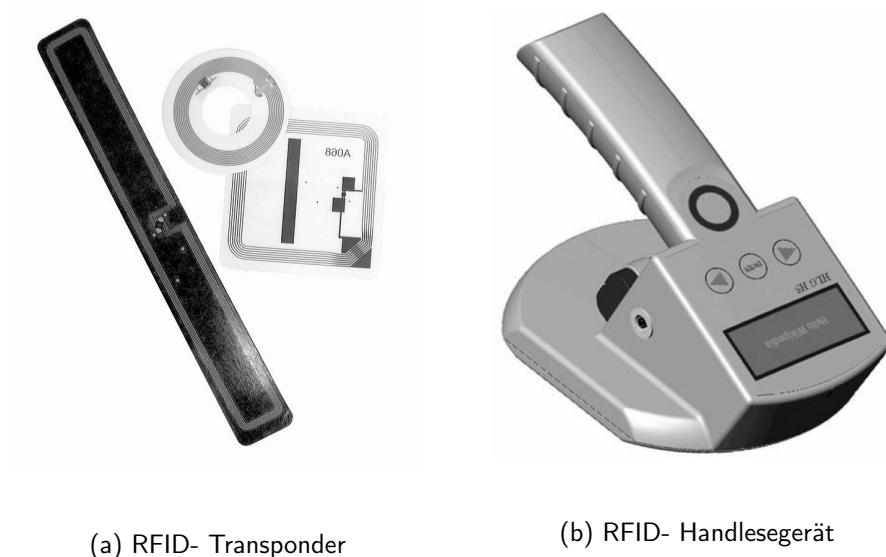
digkeit der EM-Welle nicht in Frage, da diese typischerweise gleich der Lichtgeschwindigkeit ist und die Distanz zwischen Sender und Empfänger zu gering ist. Das reduziert die Möglichkeiten auf zwei Verfahren. Bei der RSSI wird die Stärke des empfangenen Signals ausgewertet. Dies stellt eine einfache Art der Positionsermittlung dar. Jedoch kann die Signalstärke stark schwanken und erlaubt nur eine geringe Ortsauflösung. Bei der PD wird die Postion anhand der zurückgestrahlten Welle ermittelt, genauer der Phase der Welle.

### 2.6.2. RFID

Bei *Radio-Frequency Identification* (RFID) handelt es sich um einen Funkstandard der die kontaktlose Identifikation bei gleichzeitiger Erfassung zusätzlicher Informationen ermöglicht (Payload). Zur Technik gehört ein Auslesegerät (Reader) und ein oder mehrere Transponder (Tags). Eine sehr grobe Übersicht über typische Bauformen von Tags und Reader ist in 2.5 zu finden. Die dargestellten Tags sind für verschiedene Frequenzbänder. Heute verfügbare Transponder lassen sich auf nahezu jeder beliebigen Oberfläche anbringen. Das ermöglicht ein großes Anwendungsspektrum, praktisch wird die Technik in jeder Umgebung eingesetzt in der es erforderlich oder nützlich ist, Dinge kontaktlos zu identifizieren. Eine gute Übersicht über Branchen und Anwendungsgebiete für RFID ist in [32] zu finden. Im Rahmen dieser Arbeit wird kein umfassender Überblick über die Technik geboten, da die Bauformen und Spezifikationen sehr stark variieren. Ein Umfassendes Werk, gute Einführung und Übersicht zur Technik bietet [7]. Dort werden auch detailliert die physikalischen Grundlagen verschiedener Antennenbauformen und Tags erläutert. Aufgrund des großen Anwendungsspektrums und der weiten Verbreitung ist die Technik in die Kritik geraten. Unter dem Dach des Vereins *digitalcourage e. V.* existiert die Kampagne *StopRFID*. Die Kampagne hat sich zum Thema gemacht über die Anwendungsmöglichkeiten und Risiken von RFID aufzuklären [20]. Die Seiten der Kampagne bieten eine sehr weitgehende Auflistung der Anwendungen für RFID.

Die Messung der Position erfolgt über die Auswertung der Phasenlage des empfangenen Signals in Bezug auf ein Referenzsignal. In der EU gibt es verschiedene zulässige RFID-Frequenzen sie reichen von 865,0 MHz bis 868,0 MHz [6]. Man kann man die Wellenlänge mit:  $\lambda \simeq 0,35m$  angeben. Daraus folgt, dass alle 35 cm

Abbildung 2.5.: Hier gezeigt sind Beispiele für RFID Transponder und Lesegeräte. Das linke Bild zeigt drei typische Tags, nahezu jede Gestalt ist mittlerweile erhältlich. Die hier gezeigten Tags eignen sich für eine Anbringung an glatten Oberflächen. Es gibt zog weitere Bauformen, die unterschiedlichste Anwendungsspektren bedienen und sogar eine Implantation ermöglichen (nicht gezeigt). Im rechten Bild ist ein Handlesegerät gezeigt. Zum Mobilen Auslesen über mittlere bis kurze Distanzen. Auch bei den Readern gibt es unterschiedlichste Bauformen, die je nach Anwendungsfall ausgewählt werden.



die gleiche Konfiguration der Phase vorliegt. Im Rahmen dieser Arbeit wird dabei von *Isphasen* gesprochen. Daraus folgt, dass die gewonnene Information aus der Phase ist nicht eindeutig ist. D.h. es lässt sich durch die Kenntnis der Phase nicht unmittelbar auf die korrekte Postion des Tags schließen. Man kann das Problem umgehen in dem man auf die errechnete Position ein ganzzahliges Vielfaches der Wellenlänge addiert, siehe Gleichung 2.11. Die dort beschriebene Konstante  $n$  wird Wellenzahl genannt.

Das System der Amedo STS verwendet eine spezielle Antennenanordnung um die Position zu ermitteln. Dabei wird eine Antennenanzahl  $>4$  eingesetzt. Für jede dieser Antennen muss eine eigene Wellenzahl bestimmt werden. Durch Auslösung des Signals, Absorption etc. kann es dazu kommen, dass eine Antenne eine unbestimmte Zeit lang kein Signal vom Tag empfängt. Wenn die Antenne nach dieser Zeit erneut ein Signal empfängt ist die ihr zugehörige Wellenzahl unbekannt und muss neu bestimmt werden.

In realen Umgebungen treten zusätzlich noch Reflektionen und ein sog. Multipath-

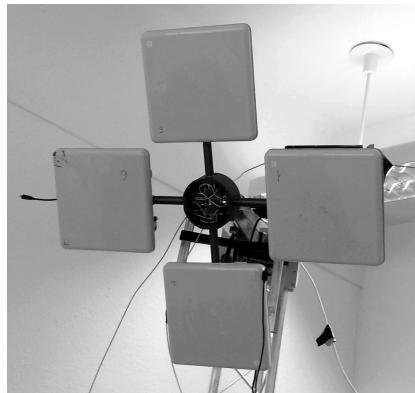
Effekt auf. Dabei wird das Signal nicht auf dem Direkten Weg Antenne-Tag-Antenne empfangen sondern über einen unbekannten, längeren Weg. Dadurch kommt es zu einem Fehler in der Phase. Zusätzlich ist dieser Effekt individuell für jede Antenne.

### 2.6.3. PRPS-Messsystem

Diese Arbeit wird für das Messsystem der amedo GmbH entwickelt. Die Entwicklung wird Teil der Softwarekomponenten des Systems werden. Aktuelle befindet sich das gesamte System in der Revision, in diesem Abschnitt wird der Stand der Entwicklung dargestellt. Details über Hardware und Software werden im Rahmen der Arbeit nicht besprochen, sofern sie diese Arbeit nicht unmittelbar betreffen.

Das auf RFID basierende PRPS (Pasiv RFID Positioning System) besteht aus mehreren frei positionierbaren Antennen, einer Mess- und Steuereinheit, sowie einem Rechner zur Kommunikation mit Endkundensoftware. Die Systemkomponenten für die Messwertaufnahme und Steuerung sind in einem Gehäuse untergebracht, dieses zeigt Abbildung 2.7. Eine typische Installation ist in Abbildung 2.8 gezeigt. Dort wurde ein System mit insgesamt acht Antennen installiert. Vier Antennen sind frei aufgestellt, vier Weitere in einer festen Anordnung (*Spinne*) installiert. Der Aufbau ist auf ein Messvolumina gerichtet.

Abbildung 2.7.: Abgebildet ist der Messaufbau aus vier Antennen. In dem Aufbau verbaut sind die wesentlichen elektronischen Komponenten wie Auswerte- und Steuereinheit. Nicht einzeln gezeigt.



### Leistungsmerkmale

Das PRPS erlaubt eine Identifikation mehrerer Objekte oder genauer: RFID-Tags, die auf den Objekten angebracht sind. Wird ein einzelner Tag ausgewählt, kann seine Position mit einer sehr hohen Frequenz ausgegeben werden. Die momentan Erreichbare Werte liegen bei 60 Hz. Je nach Umgebung kann dieser Wert jedoch Variieren, siehe Kapitel 3.1. Sollen mehrere Tags ausgegeben werden, verringert sich die Frequenz entsprechend. Bei der Verwendung von drei Tags ist eine Leserate von 20 Hz pro Tag realistisch. Die Positionsgenauigkeit liegt aktuell bei  $\approx 5$  mm. Die Antennen können frei Arrangiert werden. Das erlaubt eine Anpassung an nahezu jeden Raum und jeden Kundenbedarf. Als Tags kommen alle handelsüblichen RFID-Tags im verwendeten Frequenzband im Bereich der ETSI-Frequenzen [6] in Frage. Der Messbereich liegt bei mehreren Kubikmetern

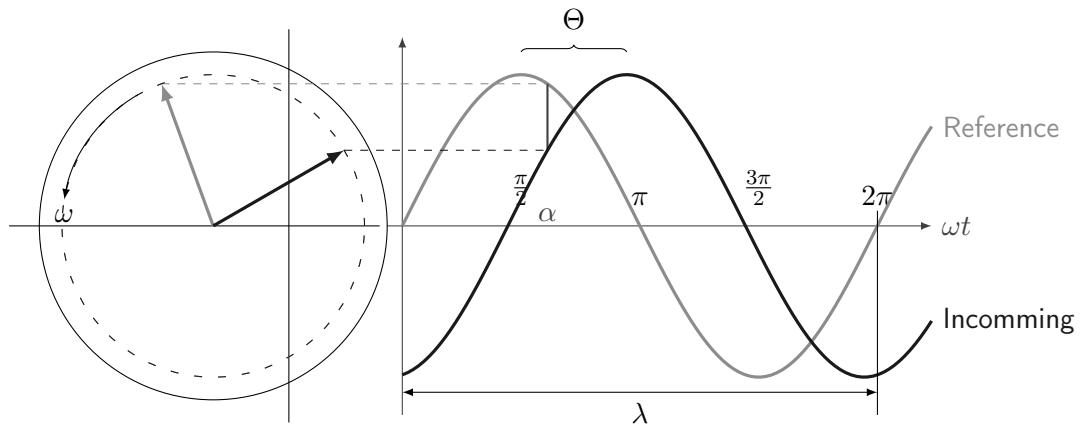
und einer erprobten maximalen Entfernung von  $6 - 7$  Metern. Theoretisch ist das maximale Volumen noch größer. Die leistungstarke Messeinheit besteht aus einer sehr flexiblen Messelektronik. Diese ist in der Lage eine sehr hohe Güte und Frequenz der Messwerte zu realisieren.

Abbildung 2.8.: Abgebildet ist der Messaufbau mit unterschiedlichen Antennen.  
Der Aufbau ist auf ein Messvolumina von mehreren Kubikmetern  
ausgerichtet



#### 2.6.4. Phase und Wellenzahl

Abbildung 2.9.: Dargestellt ist der Zusammenhang zwischen der Wellenlänge  $\lambda$  und der Wellenzahl  $n$ . Da die Phase alle  $2\pi$  den gleichen Wert annimmt, wird mit dem Faktor  $n$  ein Vielfaches der Wellenlänge aufaddiert. Dadurch erhält man die Entfernung zu dem Tag.



Aus der Abbildung 2.9 lässt sich folgender Zusammenhang ableiten.

$$d(\Theta, n) = \lambda(\Theta/2\pi + n) \quad (2.11)$$

# 3. Hauptteil

Im Folgenden werden ausführlich die Methoden und Lösungen zur beschriebenen Problemstellung vorgestellt. Zuerst wird eine Betrachtung der Komplexität des Problems präsentiert. Es werden die Modelle vorgestellt die zum Auffinden der Lösung verwendet wurden. Im Anschluss wird die Implementation der ES und die Schnittstellen zum PRPS beschrieben.

## 3.1. Vorüberlegung zur Komplexität

In diesem Abschnitt wird eine Übersicht über die Komplexität des Problems gegeben. Abbildung 3.1 zeigt die Visualisierung einer typischen Kalibriermessung. Der verwendete Aufbau ist in Abbildung A.1. gezeigt. Er besteht aus vier Antennen, die in einer Ebene angeordnet sind. Es wurde eine reproduzierbare Aufstellung verwendet (Abbildung A.2) und eine Fläche von 1 × 1 Meter vermessen. Alle 10 cm wurde eine Messung gespeichert. In der Abbildung kann man deutlich das Verhalten der Phasendaten sehen. Um diesen Verlauf deutlicher zu zeigen, wurden die Phasenwerte normiert und als Oberfläche in den Plot gelegt. Am Boden gezeigt ist der Kontur-Plot der Werte. Zwischen den Werten wurde interpoliert um die Nulldurchgänge deutlicher zu zeigen. Die Übersicht aus der Sicht aller Antennen ist in Abbildung 3.2 gezeigt.

In der Abbildung 3.3 werden die Daten ohne Interpolation dargestellt. Es wurden die Höhenlinien eingezeichnet. Die Anordnung der Plots soll ein Gefühl dafür vermitteln, wie die Messwerte eines Tags sich an unterschiedlichen Positionen und aus Sicht verschiedener Antennen verhalten.

Die Darstellung echter Messwerte lässt Rückschlüsse auf die Komplexität des Problems zu. Es ist leicht nachzuvollziehen, dass das Zusammenspiel der Messwerte eine sehr komplexe Szene ergibt. Hier dargestellt ist bereits das Verhalten bei der Verwendung von vier Antennen. Der aktuelle Messaufbau erlaubt sogar acht

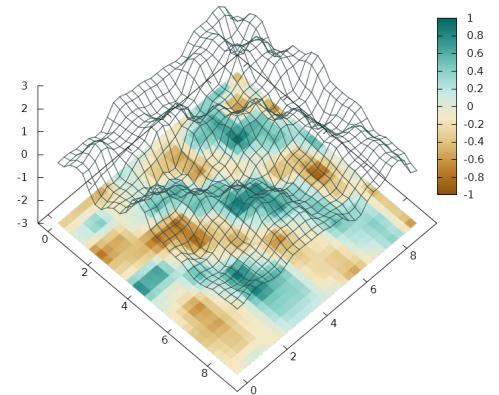


Abbildung 3.1.: Normiertes Höhenprofil einer Phasenmessung aus der Sicht von Antenne 1

Antennen. Das ergibt insgesamt eine komplexe Szenerie.

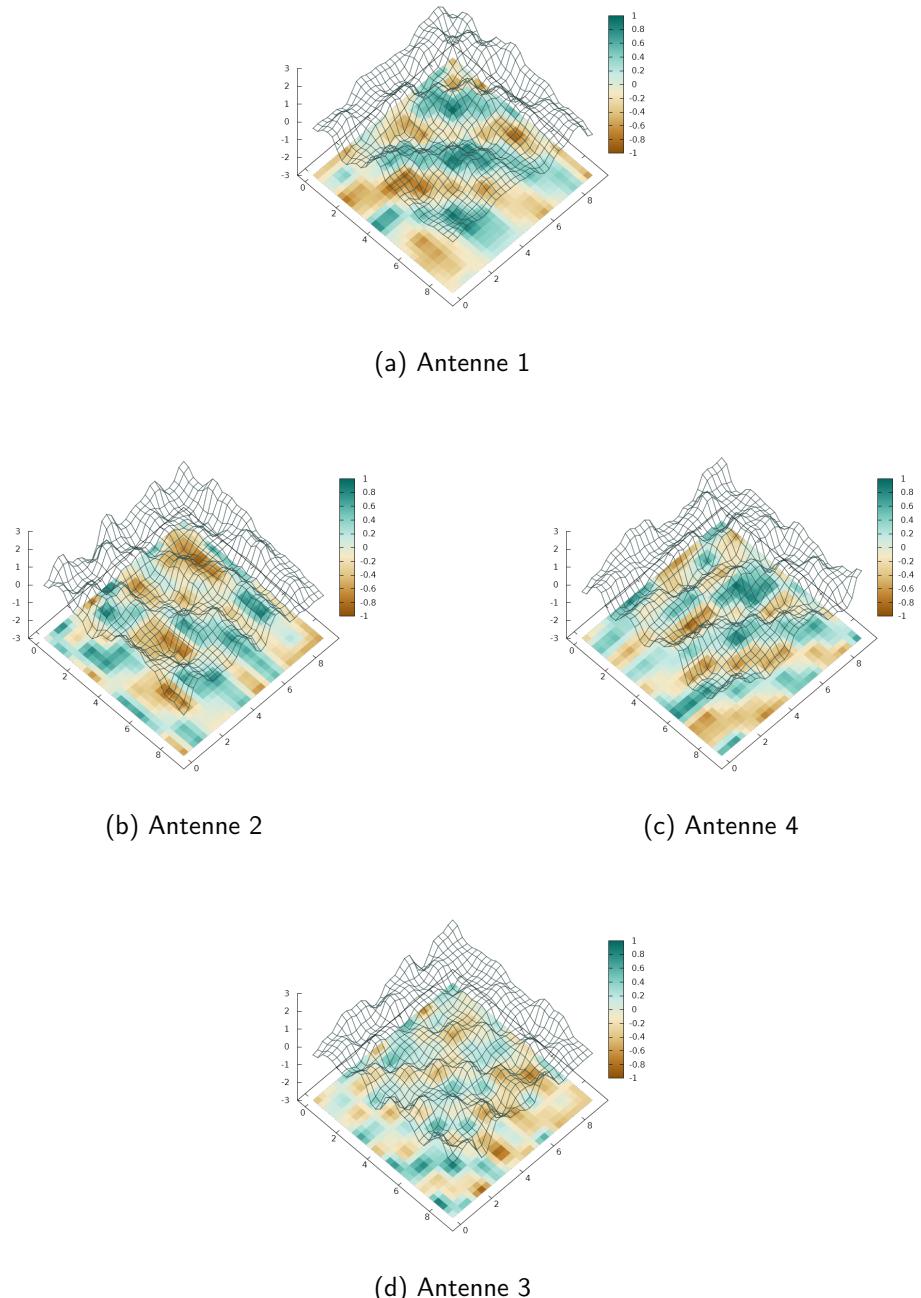


Abbildung 3.2.: Blick auf die Messwerte der Kalibrierplatte aus der "Sicht" der Antennen. Dabei zeigt sich deutlich der Wellencharakter der Messung, dieser ist zu erwarten. Die Messungen wurden mit einer Frequenz von 865,7 MHz unter Laborbedingungen aufgenommen.

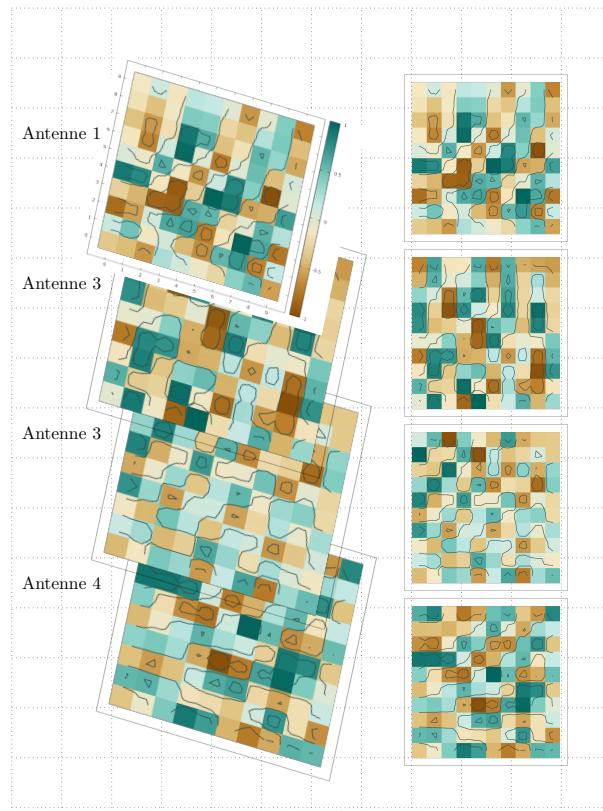


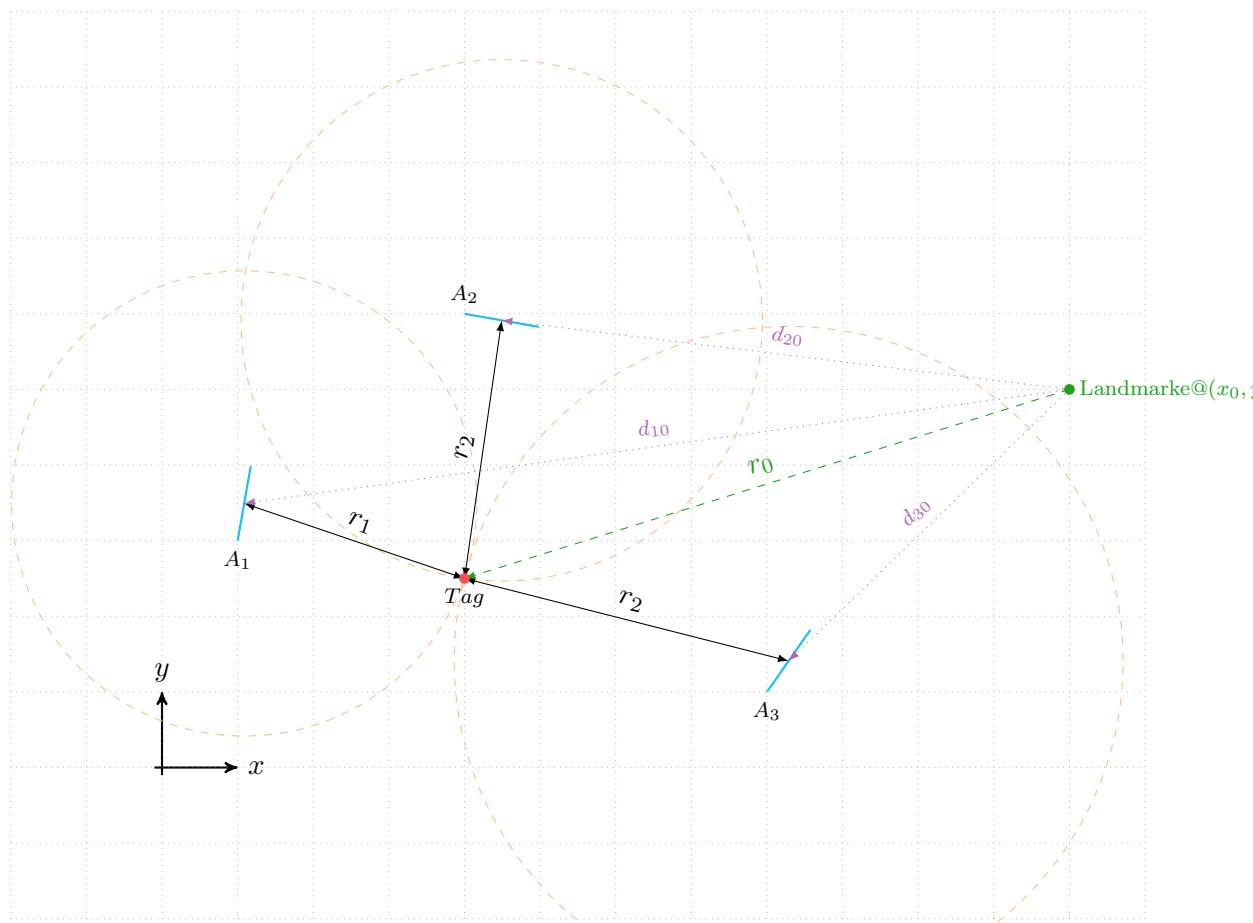
Abbildung 3.3.: Diese Grafik zeigt die Visualisierung von realen Phasen-Messwerten. Die Daten wurden durch Vermessung einer  $1 \times 1$ -Kalibrierplatte mit reproduzierbarer Aufstellung<sup>1</sup> gewonnen. Die Daten wurden normiert. In jeder Dimension wurden  $10 \times 10$  Werte aufgenommen. Die Darstellung der Phasenwerte erfolgt als Heatmap, es soll qualitativ der Verlauf der Phasenwerte gezeigt werden. Zur Orientierung sind in jedem Plot Höhenlinien eingezeichnet. Pro Plot werden die Daten einer Antenne dargestellt. Die Antenne von der die Daten stammen ist angegeben.

### 3.2. Entwicklung des Modells

Im folgenden Abschnitt wird das Modell für die Lösung des Zusammenhangs entwickelt. Zur Veranschaulichung des Sachverhalts dient die Abbildung 3.4. Dort skizziert ist der Messaufbau mit einem Tag. Die Szene ist in 2D dargestellt, die Ableitung des Modells erfolgt direkt für drei Raumkoordinaten. Folgende Nomenklatur und Symbole gelten für diesen Abschnitt:

- $r_k :=$  Abstand vom Tag zur Antenne
- $d_{kJ} :=$  Abstand zur Landmarke
- $N_0 :=$  Menge der verfügbaren Antennen  $N = \{1, \dots, 8\}$

Abbildung 3.4.: 2D-Übersicht auf die Szene mit drei Antennen, einem Tag und einer Landmarke. Die Position von  $\{A_1, A_3, A_3\}$ , sowie der Landmarke, zum Koordinatenursprung sind bekannt. Die Vektoren  $r_1, r_2, r_3$  sind die gemessene Entfernung zu einer Antenne. Die Landmarke wird im späteren Verlauf eine Antenne sein, die ihrerseits eine gemessene Entfernung  $r_0$  produziert. Der Schnittpunkt aller Kreise ist die Lösung der gemessenen Entfernung und der geom. Anordnung, die sich für die Position des Tags ergibt.



- $N :=$  Menge der Antennen für die Optimierungen verfügbar sind<sup>2</sup> ( $N \subseteq N_0$ )
- $N' :=$  Menge der Antennen für die Optimierung ( $N' \subseteq N$ )
- $j$  ist der Index der Referenzantenne, es gilt  $j = \{1, 2, \dots, 8\}$
- $k$  ist der Index der Antennen einer Messung, es gilt  $k = 1, 2, \dots, |N'| - 1$

Wir starten mit der Überlegung über den geometrischen Zusammenhang zwischen der Antennenposition von Antenne  $k$  zu der Position des Tags  $r_k$ :

$$r_k^2 = (x - x_k)^2 + (y - y_k)^2 + (z - z_k)^2 \quad (3.1)$$

Diese Gleichung stellt die euklidische Vektornorm dar und entspricht der Strecke Antenne-Tag. Für die Ermittlung einer Postion (mit drei Raumkoordinaten) sind drei Antennen Notwendig. Daraus ergibt sich:

- 3 Gleichungen
- 3 Unbekannte
- Quadratisches Gleichungssystem

Das Gleichungssystem sieht wie folgt aus:

$$\begin{aligned} r_1^2 &= (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 \\ r_2^2 &= (x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 \\ r_3^2 &= (x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2 \end{aligned}$$

Es ist trivial und es wird in verschiedenen Beispielen gezeigt<sup>3</sup>, dass man die Koordinaten aus dem quadratischen Gleichungssystem unmittelbar berechnen kann. Es muss jedoch ein quadratisches Gleichungssystem gelöst werden, was zu den bekannten Problematiken führt, insbesondere der Ausschluss mehrdeutiger Ergebnisse. Der Messaufbau der amedo GmbH erlaubt die Verwendung von mehr als 3 Messwertgebern. Diese zusätzlichen Informationen lassen sich für eine Linearisierung des Gleichungssystems verwenden. Dieser Ansatz wird für ein Modell im Rahmen dieser Arbeit verwendet und wird im Folgenden beschrieben.

Von den Antennen sind die Raumkoordinaten ( $x, y, z$  – Koordinaten) bekannt, bzw. wurden durch Kalibrierung (vgl. Abschnitt 3.7) in einem vorherigen Schritt bestimmt. Wir können zusätzlich zu notieren:

$$d_{kj}^2 = (x_k - x_0)^2 + (y_k - y_0)^2 + (z_k - z_0)^2 \quad (3.2)$$

Linearisierung des Modells. Dazu wird Gleichung 3.1 in mehreren Schritten umgebaut. Zuerst wird eine neutrale Erweiterung durchgeführt und die Terme zusam-

---

<sup>2</sup>d.h. ein Messergebnis liefern

<sup>3</sup>z.B. <http://en.wikipedia.org/w/index.php?title=Trilateration&oldid=553215995>

mengefasst. Das führt zu:

$$\begin{aligned}
 r_k^2 &= (x - x_k)^2 + (y - y_k)^2 + (z - z_k)^2 \\
 &= (x - x_k + x_0 - x_0)^2 + (y - y_k + y_0 - y_0)^2 + (z - z_k + z_0 - z_0)^2 \\
 &= ((x - x_0) - (x_k - x_0))^2 + ((y - y_0) - (y_k - y_0))^2 + ((z - z_0) - (z_k - z_0))^2 \\
 &= (x - x_0)^2 - 2(x - x_0)(x_k - x_0) + (x_k - x_0)^2 \quad \underbrace{+ \dots + \dots}_{\text{y- \& z-Terme analog}} \quad (3.3)
 \end{aligned}$$

Um Platz zu sparen sind die y- und z-Terme nicht explizit notiert. Sie ergeben sich durch einfaches Ersetzen der Indizes und werden im finalen Modell eingefügt. Durch Umstellen von (3.3) erhalten wir:

$$\begin{aligned}
 (x - x_0)(x_k - x_0) + \dots + \dots &= -\frac{1}{2}[r_k^2 - (x_k - x_0)^2 - (x - x_0)^2 + \dots + \dots] \\
 (x - x_0)(x_k - x_0) + \dots + \dots &= \frac{1}{2}[(x_k - x_0)^2 + (x - x_0)^2 + \dots + \dots - r_k^2] \\
 (x - x_0)(x_k - x_0) + (y - y_0)(y_k - y_0) + (z - z_0)(z_k - z_0) &= \\
 \frac{1}{2}[(x_k - x_0)^2 + (x - x_0)^2 - (y_k - y_0)^2 + (y - y_0)^2 \\
 &\quad - (z_k - z_0)^2 + (z - z_0)^2 - r_k^2] \quad (3.4)
 \end{aligned}$$

Vergleich von (3.4) mit (3.2) bringt:

$$\begin{aligned}
 (x - x_0)(x_k - x_0) + (y - y_0)(y_k - y_0) + (z - z_0)(z_k - z_0) &= \\
 \frac{1}{2}[\underbrace{(x_k - x_0)^2 + (z_k - z_0)^2 + (y_k - y_0)^2}_{d_{kj}^2} \\
 &\quad + \underbrace{(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 - r_k^2}_{r_j^2}] \quad (3.5)
 \end{aligned}$$

$$(x - x_0)(x_k - x_0) + (y - y_0)(y_k - y_0) + (z - z_0)(z_k - z_0) = \frac{1}{2}[d_{kj}^2 + r_j^2 - r_k^2] \quad (3.6)$$

mit

$$\mathbf{c}_{kj} = \frac{1}{2}[d_{kj}^2 + r_j^2 - r_k^2] \quad (3.7)$$

können wir für das lineare Gleichungssystem abschließend schreiben:

$$\mathbf{0} = \begin{pmatrix} x_1 - x_j & y_1 - y_j & z_1 - z_j \\ x_2 - x_j & y_2 - y_j & z_2 - z_j \\ x_3 - x_j & y_3 - y_j & z_3 - z_j \end{pmatrix} \begin{pmatrix} x - x_j \\ y - y_j \\ z - z_j \end{pmatrix} - \begin{pmatrix} c_{1j} \\ c_{2j} \\ c_{3j} \end{pmatrix} \quad (3.8)$$

Das Gleichungssystem ist linear und hat die allg. Form:  $\mathbf{0} = \mathbf{Ax} + \mathbf{b}$ . Es lässt sich daher mit bekannten Methoden lösen.

### Zusammenhang mit der Wellenzahl

Wie gezeigt wurde ergibt sich für den Fall der Trilateration und der Annahme, dass vier Antennen Messwerte liefern, die Gleichung:

$$\mathbf{0} = \begin{pmatrix} x_k - x_0 & y_k - y_0 & z_k - z_0 \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{pmatrix} - (c_{kj}) \quad (3.9)$$

Wir stellen fest, dass dieses Modell rein geometrisch ist. Es erlaubt bereits einen Einsatz im Rahmen der Kalibrierung (siehe 3.7). Es wird im Folgenden eine Erweiterung dieses Modells gezeigt. Ziel ist es, einen Zusammenhang zwischen diesem Modell, der gemessenen Phase und der Wellenzahl zu erzeugen. Folgender Ansatz wird gewählt:

$$r(\varrho, n) = \frac{\lambda}{2} \left( \frac{\varrho}{2\pi} + n \right), \lambda = \frac{c}{f}, n := \text{Wellenzahl} \quad (3.10)$$

In dem Modell steht  $\varrho_k$  für die gemessene Phase vom Messsystem und  $n_k$  ist die gesuchte Wellenzahl. Der Index  $k$  deutet eine Existenz der beiden Parameter für jede Antenne an. Durch einsetzen von (3.10) in (3.7), erhalten wir:

$$c_{kj}(\varrho_0, \varrho_k, n_0, n_k) = \frac{1}{2} \left[ d_{kj}^2 + \frac{\lambda^2}{4} \left( \frac{\varrho_j}{2\pi} + n_0 \right)^2 - \frac{\lambda^2}{4} \left( \frac{\varrho_k}{2\pi} + n_k \right)^2 \right] \quad (3.11)$$

Wir stellen Gleichung (3.11) um:

$$c_{kj}(\varrho_0, \varrho_k, n_0, n_k) = \frac{1}{2} \left\{ d_{kj}^2 + \frac{\lambda^2}{4} \left[ \left( \frac{\varrho_j}{2\pi} \right)^2 + 2 \frac{\varrho_j}{2\pi} n_0 + n_0^2 - \left( \frac{\varrho_k}{2\pi} \right)^2 - 2 \frac{\varrho_k}{2\pi} n_k - n_k^2 \right] \right\} \quad (3.12)$$

$$= \frac{1}{2} \left\{ d_{kj}^2 + \frac{\lambda^2}{4} \left[ \left( \frac{\varrho_j}{2\pi} \right)^2 - \left( \frac{\varrho_k}{2\pi} \right)^2 + 2 \frac{\varrho_j}{2\pi} n_0 - 2 \frac{\varrho_k}{2\pi} n_k + n_0^2 - n_k^2 \right] \right\} \quad (3.13)$$

$$= \frac{1}{2} d_{kj}^2 + \frac{\lambda^2}{8} \left[ \frac{1}{(2\pi)^2} (\varrho_0^2 - \varrho_k^2) + \frac{1}{\pi} (\varrho_0 n_0 - \varrho_k n_k) + (n_0^2 - n_k^2) \right] \quad (3.14)$$

Führen wir nun:

$$\begin{aligned} a_{0k} &:= \frac{1}{2} d_{kj}^2 \\ a_1 &:= \frac{\lambda^2}{8} \\ a_2 &:= a_1 \frac{1}{\pi} \end{aligned}$$

$$a_{3kj} := a_1 \frac{1}{(2\pi)^2} (\varrho_j^2 - \varrho_k^2)$$

in Gleichung (3.14) ein, erhalten wir die finale Form der Gleichung:

$$c_{kj}(\varrho_0, \varrho_k, n_0, n_k) = a_{0k} + a_1(n_0^2 - n_k^2) + a_2(\varrho_0 n_0 - \varrho_k n_k) - a_{3kj} \quad (3.15)$$

Die Einführung der Konstanten macht zum einen die Gleichung übersichtlicher. Zum anderen können so in der spätere Softwareimplementation Rechenschritte gespart werden, was sich günstig auf den Rechenaufwand auswirkt. Im Weiteren erkennt man, dass in Gleichung (3.15), für  $\varrho_k = \text{const.}$  &  $\varrho_0 = \text{const.}$  gilt. Der Grund dafür liegt darin, dass  $\varrho$  zwar die Messwerte beschreibt, diese jedoch nur in dem Modell eingeführt werden. Im Sinne der später durchgeführten Optimierung sind diese Parameter keine Variablen. Es ermöglicht uns zu schreiben:

$$c_{kj}(\varrho_0, \varrho_k, n_0, n_k) = c_{kj}(n_0, n_k) \quad (3.16)$$

Im engeren Sinne einer mathematischen Funktion sollten wir die Parameter alle als Argument aufnehmen. Diese Form soll darstellen, welche Größen von Interesse sind. Im späteren Gebrauch wird diese Gleichung in der Optimierung eingesetzt werden.

Für unser Gleichungssystem aus (3.9) ergibt sich:

$$\mathbf{0} = \begin{pmatrix} x_k - x_0 & y_k - y_0 & z_k - z_0 \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{pmatrix} - \begin{pmatrix} c_{kj}(n_0, n_k) \end{pmatrix} \quad (3.17)$$

### Konkretes Beispiel

Für ein konkretes Beispiel Betrachten wir nun (3.17). Dabei wählen wir  $|N'| = 4$  (d.h. wir verwenden 4 Antennen) und setzen  $j = 0$ . Diese exemplarische Konfiguration kann wie folgt beschrieben werden: Antenne 0 ist die Referenz-Antenne und Antennen 1, 2 und 3 sind Messwertgeber für die Phaseninformation. Im praktischen Gebrauch werden die Konfigurationen anders zusammengestellt. Strategien für die Zusammenstellung werden später beschrieben.

Für die gewählte Konfiguration ergibt sich explizit:

$$\mathbf{0} = \underbrace{\begin{pmatrix} x_1 - x_0 & y_1 - y_0 & z_1 - z_0 \\ x_2 - x_0 & y_2 - y_0 & z_2 - z_0 \\ x_3 - x_0 & y_3 - y_0 & z_3 - z_0 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{pmatrix}}_{\mathbf{x}} - \underbrace{\begin{pmatrix} c_{10}(n_0, n_1) \\ c_{20}(n_0, n_2) \\ c_{30}(n_0, n_3) \end{pmatrix}}_{\mathbf{b}} \quad (3.18)$$

Wir wollen den Vektor  $\mathbf{b}$  nun explizit betrachten:

$$\mathbf{b} = \begin{pmatrix} a_{01} + a_1(n_0^2 - n_1^2) + a_2(\varrho_0 n_0 - \varrho_1 n_1) - a_{310} \\ a_{02} + a_1(n_0^2 - n_2^2) + a_2(\varrho_0 n_0 - \varrho_2 n_2) - a_{320} \\ a_{03} + a_1(n_0^2 - n_3^2) + a_2(\varrho_0 n_0 - \varrho_3 n_3) - a_{330} \end{pmatrix} \quad (3.19)$$

Das Ergebnis ist ein um  $\varrho$  und  $n$  erweitertes Gleichungssystem. Zusätzlich enthält es mehrere geometrische Konstanten ( $a_{0k}$ ), mehrere Phasen-Konstanten ( $a_{3k0}$ ),

sowie zwei Systemparameter abhängige Konstanten ( $a_1$  und  $a_2$ ). Allgemeiner formuliert ergibt sich:

$$0 = \begin{pmatrix} x_k - x_0 & y_k - y_0 & z_k - z_0 \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{pmatrix} - \left( a_{0k} + a_1(n_0^2 - n_k^2) + a_2(\varrho_0 k_0 - \varrho_k n_k) - a_{3kj} \right) \quad (3.20)$$

### Hinzufügen von Antennen - Der allgemeine Fall

Aus dem oben beschriebenen Beispiel, Gleichung (3.20), und der dort getroffene Wahl von  $|N'| = 4$  ergibt sich wie viele Veränderliche sich für eine gewählte Konstellation an Antennen ergeben. Leiten wir daraus nun einen allgemeinen Fall ab. Für  $k$  gilt in diesem Fall  $k = \{1, \dots, N' - 1\}$ , wir wählen die Referenzantenne  $j = 0$  und die Menge an verwendeten Antennen gleich der Anzahl der verfügbaren ( $N' = N$ ). Es ist leicht ersichtlich, dass sich die Anzahl der verwendeten Antenne unmittelbar auf die Zahl der Variablen auswirkt. Es ergeben sich für das Modell mit vier Antennen insgesamt 7 Variablen ( $\mathbf{x}, n_0, n_1, n_2, n_3$ ), wobei sich für ein Modell mit allen 8 Antennen, 11 Variablen ( $\mathbf{x}, n_0, \dots, n_7$ ) ergeben. Andere Konfigurationen verhalten sich analog dazu.

### Relevanz dieses Modells

Dieses Modell hat unmittelbare Relevanz für die Praxis. Es trägt dem Umstand Rechnung, dass zu einem Messzeitpunkt ein Teil der Antennen keine Messwerte liefern könnte. Das Modell erlaubt daher, dass die Anzahl und die Auswahl der Antennen variieren kann. Damit ist das Modell uneingeschränkt tauglich für den Einsatz in dem PRPS-Messsystem.

Abschließend soll das bisher verwendete Modell umgeschrieben werden, damit die Allgemeingültigkeit darin enthalten ist.

$$\mathbf{A} = \begin{pmatrix} x_k - x_0 & y_k - y_0 & z_k - z_0 & \sum_{i=1,j=0}^k (-a_1 \delta_{ij}) & -a_2 \Theta_0 & \sum_{i=1,j=0}^k (a_2 \Theta_k \delta_{ij}) \end{pmatrix}$$

$$\mathbf{x} = \begin{pmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \\ n_0^2 - n_k^2 \\ n_0 \\ n_k \end{pmatrix}$$

$$\mathbf{b} = a_{0k} - a_{3kj} = c'_{kj}$$

Dabei steht  $\delta_{ij}$  für den bekannten Kronecker-Operator und bedeutet:

$$\delta_{ij} = \begin{cases} 1 & \text{für } i = j \\ 0 & \text{für } i \neq j \end{cases}$$

Im Expliziten sehen die Matrix  $\mathbf{A}$  und der Vektor  $\mathbf{b}$ , für den Fall  $N' = 3$  und  $k = \{1, 2, 3\}$ , wie folgt aus:

$$\mathbf{A} = \begin{pmatrix} x_1 - x_0 & y_1 - y_0 & z_1 - z_0 & -a_1 & 0 & 0 & -a_2\Theta_0 & a_2\Theta_3 & 0 & 0 \\ x_2 - x_0 & y_2 - y_0 & z_2 - z_0 & 0 & -a_1 & 0 & -a_2\Theta_0 & 0 & a_2\Theta_3 & 0 \\ x_3 - x_0 & y_3 - y_0 & z_3 - z_0 & 0 & 0 & -a_1 & -a_2\Theta_0 & 0 & 0 & a_2\Theta_3 \end{pmatrix}$$

$$\mathbf{x} = \begin{pmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \\ n_0^2 - n_1^2 \\ (\dots) \\ n_0^2 - n_3^2 \\ n_0 \\ n_1 \\ (\dots) \\ n_3 \end{pmatrix}$$

### Bemerkungen - Finales Modell

Das Ergebnis ist eine  $3 \times 10$  Matrix und ein  $1 \times 10$  Vektor. Es ist möglich diesem Modell eine beliebige Anzahl an Antennen hinzuzufügen. Fügt man eine Antenne zur Berechnung hinzu, würde sich die Matrix  $\mathbf{A}$  um zwei Spalten und eine Zeile erweitern, der Vektor  $\mathbf{x}$  analog um 2 Zeilen.

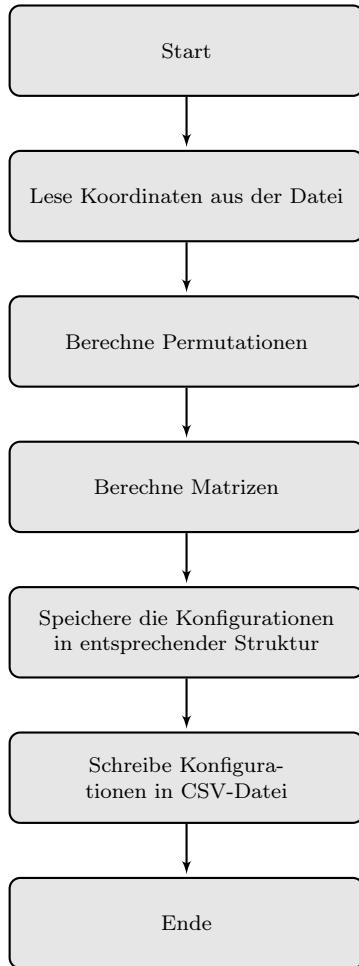
### 3.3. Antennen Permutationen

Um die Beurteilung der Konditionszahl der Matrizen effizient durchführen zu können, wurde im Rahmen dieser Arbeit ein Programm geschrieben. Es erstellt automatisch, auf Basis der durch die Kalibrierung bestimmten Koordinaten, alle möglichen Permutationen von Antennen. Die sich ergebenden Matrizen sind immer auf eine Referenzantenne bezogen. Es ergeben sich, bezogen auf eine Referenzantenne, folgende Anzahl an Matrizen:

$$\frac{7!}{3!(7-3)!} = 35 \quad (3.21)$$

Für einen Aufbau mit acht Antennen ergeben sich so  $8 \times 35 = 280$  mögliche Anordnungen. Die Implementation der Berechnungen der Permutationen findet sich in dem Modul '*libPermute*'. Das Modul generiert bei der Instanziierung automatisch alle möglichen Kombinationen von Antennen und speichert diese in einer geeigneten Struktur für den späteren Gebrauch. Das Ablaufdiagramm ist in Abbildung 3.5 zu finden.

Abbildung 3.5.: lorem



### 3.4. Erweiterte Betrachtung der Kondition

Die vorgestellte erweiterte Form des Modells erleichtert die Implementation und Verifikation, da große Teile vorberechnet und in geeigneten Strukturen abgelegt werden können. Diese statischen Teile des Modells sind in Gleichung 3.22 ersichtlich. Es sind nun auch die gemessenen Phasenwerte Teil des Modells, genauer: der Matrix  $\mathbf{A}$ . Im Folgenden werden die Auswirkungen auf die Kondition der Matrix betrachtet, wenn man diese Phasendaten hinzurechnet. Weiterhin wird Untersucht inwieweit die Zerlegung in Blockmatrizen und die Untersuchung der Kondition dieser eine Abschätzung der vollständigen Konditionszahl im Allgemeinen darstellt.

$$\mathbf{A} = \begin{pmatrix} \mathbf{Z} & \mathbf{P} & \mathbf{V} \end{pmatrix} \quad (3.22)$$

Dabei ist:

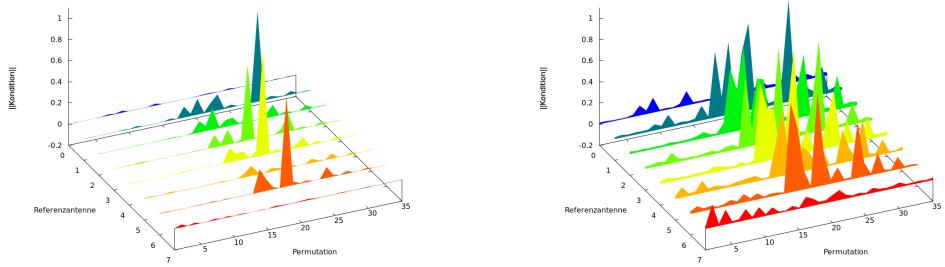
$$\mathbf{Z} \in \mathbb{R}^{3 \times 3} \quad \mathbf{P} \in \mathbb{R}^{3 \times 3} \quad \mathbf{V} \in \mathbb{R}^{4 \times 3} \quad (3.23)$$

Die Matrizen  $\mathbf{Z}$  und  $\mathbf{P}$  sind statisch. Hingegen enthält die Matrix  $\mathbf{V}$  die gemessenen Phasenwerte  $\Theta_k$  der Antennen für diese Konfiguration.

Die Abbildung 3.6 zeigt die bereits angestellte Untersuchung zu dieser Überlegung. Abbildung 3.7a stellt die Konditionszahl der rein geometrischen  $3 \times 3$ -Matrix dar. In der Abbildung 3.7b sehen wir die Kondition der erweiterten Matrix. Neben der geometrischen sind auch die beiden anderen Blockmatrizen in diese Konditionsbetrachtung eingeflossen. Als zusätzliche Angabe sind die Skalierungsfaktoren angegeben. Legt man beide Grafiken übereinander erkennt man:

1. Geometrisch gut konditionierte Konfigurationen (linke Grafik), bleiben im erweiterten Modell (rechte Grafik) weiterhin gut konditioniert.
2. Die Konditionszahl der *schlechten* Konfiguration ist wesentlich kleiner (ca. Faktor 10) als im rein geometrischen Modell

Abbildung 3.6.: Analyse der Konditionszahlen aller möglichen Matrizen für den Messaufbau. Die Konditionszahl ist für jede mögliche Permutation an Messantennen für eine Referenzantenne angegeben



(a) Konditionszahl der rein geometrischen  $3 \times 3$  Matrix normiert auf den größten vorkommenden Wert ( $= 2149, 16$ ). Auf den Achsen finden sich der Index der Referenzantenne sowie die Nummer der Permutation. Die z-Achse enthält die normierte Kondition

(b) Konditionszahl der  $10 \times 3$  Matrix normiert auf den größten vorkommenden Wert ( $= 257, 13$ ); In dieser Konfiguration sind die Konstanten ( $a_1$  &  $a_2$ ) sowie die variablen, gemessenen Phasen  $\Theta_k$  enthalten

Aus der Grafik lässt sich entnehmen, dass für jede Referenzantenne aus der Geometrie alleine gute Konfigurationen existieren. Aus diesen Erkenntnissen kann in späteren Aufbauten die Position der Antennen optimiert werden. Dieses Verfahren wird in Abschnitt 6.1 weiter beschrieben. Die Grafik lässt erkennen, dass eine Konfiguration, die ohne Phasendaten eine gut Kondition aufwies, eine ähnliche Kondition behält, wenn diese Daten in das Modell einfließen.

### 3.4.1. Weitere Anwendung der Konditionszahl

Weitere Anwendungen, die sich aus der Konditionszahl der Matrix ableiten, sind denkbar. Die Steuereinheit wird, parallel zu diesem Projekt, um eine intelligente Umschaltung der Antennen erweitert. Das Problem ist dabei die 'wissende' Umschaltung auf andere Antennen. Dazu lässt sich die Kondition der Antennenkombinationen nutzen. Die Kondition der geometrischen Matrix verändert sich nach dem Kalibrieren nicht mehr. Dadurch und durch die oben beschriebenen Überlegungen kann statisch eine Abschätzung für die Konditionszahl, von zwei der drei Blockmatrizen, im Vorfeld erstellt werden. Die Konditionszahl dient zum Steuern der Umschaltung. Ordnet man die möglichen Konfiguration anhand ihrer Konditionszahl (niedrigste zuerst) in einer statischen Liste an, so kann in der Steuereinheit eine einfache, schlaue Umschaltung implementiert werden. Diese würde immer dafür sorgen, dass Messdaten von einer Konfiguration bevorzugt werden, die eine niedrige Konditionszahl hat und somit relativ sicher zu einer guten Lösung führen. Diese Überlegungen werden im Rahmen dieser Arbeit nicht näher beschrieben. Eine weitere Anwendung ergibt sich für die Kalibrierung. Der Aufbau der Antennen kann unter Berücksichtigung der Kondition optimiert werden. Ziel der Optimierung wäre es, durch eine geeignete Positionierung der Antennen, die Anzahl der Antennenpermutationen mit kleiner Konditionszahl zu maximieren.

## 3.5. Einsatz des Modells

Im vorherigen Abschnitt wurde das Modell aus den geometrischen Gegebenheiten hergeleitet. Das Modell ist in seinen Eingabeparametern flexibel und erlaubt verschiedene Arten des Einsatzes. Diese werden im Folgenden erläutert.

## 3.6. Betrachtung der Komplexität

Im Folgenden wird eine Betrachtung der Komplexität des in Abschnitt 3.2 entwickelten Modells präsentiert. Diese Betrachtung ist wichtig für die Parametrisierung des Optimierungsverfahrens, sowie für eine Beurteilung der generellen Lösbarkeit mit den verwendeten Verfahren. Es wird eine Visualisierung der Fitness-Landschaft<sup>4</sup> vorgestellt.

Um einen Ausschnitt der Fitnesslandschaft zu erstellen, die sog. Fitness-Ebenen, wurde ein eigener Programmteil (der *FitnessPlaneCalculator*) entwickelt. Das Programm nimmt ein in dieser Arbeit entwickeltes Modell, fürttert es mit vorgegebenen Daten und schreibt den Rückgabewert in eine Datei. Das Programm lässt sich per Eingabedatei steuern und erlaubt die Definition der Ebenen die dargestellt werden sollen. Es lassen sich immer zwei Variablen der Objektfunktion variieren und der Rest wird dabei auf feste Werte gesetzt. Das erlaubt eine Visualisierung durch eine 2D-Heatmap (siehe folgende Plots). Formal lässt sich das vorgehen wie folgt

---

<sup>4</sup>Auch Fitness-Raum genannt

beschreiben:

$$f(\mathbf{x}) \in \mathbb{R}^N \rightarrow \mathbb{R}^2$$

Aus der Visualisierung können Rückschlüsse auf die Gestalt der Fitnessebene gezogen werden.

Die Parameter wurden für diese Plots so gewählt, dass sie über einen Bereich iterieren, indem die richtige Lösung liegen sollte. Eingezeichnet sind verschiedene Höhenlinien, die zur Orientierung dienen sollen. Diese sind für die Werte des diskreten Intervalls:  $[0, 1, 5, 10, 50, 100, 200]$  ausgeführt. Sie tragen zur besseren Übersicht bei. Die Farbskala das Heatmap skaliert den Plot auf den Wertebereich  $[0, 1]$  und addiert das im Plot vorkommende Minimum dazu. So kann man leicht eine qualitative Beurteilung der Ebene ablesen. Da nur drei Parameter variiert werden, erhalten die übrigen vier analytisch bestimmte, wahre Werte. Diese sind in Tabelle 3.1 zusammen mit den wahren Werten aufgeführt.

In den Abbildungen 3.9 und 3.10 zeigen sich erste Herausforderungen für den Algorithmus. Eine große, sehr flache Fitnessebene für verschiedene Parameter ist nicht leicht zu handhaben. Eine Möglichkeit dieses Problem zu umgehen ist eine große Anzahl an Nachkommen zu generieren. Im Jargon der EA: großes  $\lambda$ . In den Ergebnissen wurde mit unterschiedlichen Populationsgrößen experimentiert, dabei zeigte sich, dass die Anzahl der Nachkommen groß sein sollte, damit eine gute Güte der Lösung gefunden werden kann. Ein anderer, steuernder Faktor ist die Schrittweite  $\sigma$ . Diese darf nicht zu klein werden, sonst besteht die Gefahr mit der Population auf der flachen Ebene liegen zu bleibt. Was zu mehrdeutigen Ergebnissen führt. Die Fitness-Ebenen der anderen Antennen zeigen das gleiche Verhalten. Aufgrund der Komplexität des in dieser Arbeit erstellten Modells (siehe Abschnitt 3.3), ist bereits der Fall mit vier Antennen sehr hochdimensional. Er erreicht 7 Dimensionen und er kann im Rahmen dieser Arbeit nicht vollständig untersucht werden. Die Fitness-Plots aller Antennen und für drei Ebenen sind in Anhang D zu finden. Eine vollständige Untersuchung ist indes auch nicht notwendig, da der Algorithmus den Suchraum natürlicherweise untersucht.

Tabelle 3.1.: Tabellarisch sind hier die Parameter (Intervalle) der Fitness Ebenen aufgelistet. Zusätzlich sind die Werte angegeben, in denen eine optimale Lösung liegen sollte. Notation: [Start:Inkrement:Ende]

Ebene	x	y	z	n <sub>0</sub>	n <sub>1</sub>	n <sub>2</sub>	n <sub>3</sub>
x-y	[-10:2:10]	[-10:2:10]	[-7:1:7]	[7:0:7]	[10:0:10]	[13:0:13]	[9:0:9]
x-z	[-10:2:10]	[-7:1:7]	[-10:2:20]	[7:0:7]	[10:0:10]	[13:0:13]	[9:0:9]
y-z	[-7:1:7]	[-10:2:10]	[-10:2:10]	[7:0:7]	[10:0:10]	[13:0:13]	[9:0:9]
Wahre	0.479	-1.012	0.607	7	10	13	9

Abbildung 3.9.: Diese Grafik zeigt die Fitnessebenen  $\mathbb{R}^2$  des Problems für eine Anordnung aus vier Antennen und einem Sweep über die  $x - y$ -Ebene. Jeder Plot ist für einen festen Wert  $z$  erstellt worden. In der oberen linken Ecke beginnend und nach rechts laufend. Die  $x, y$ -Werte wurden über ein Intervall von  $[-5, 5]$  m und mit einem Inkrement von 0.2 variiert. Daraus ergibt sich eine für die Abschätzung der Gestalt der Fitnessebene ausreichende Datengrundlage. Die  $z$ -Werte der 15-Plots stammen aus dem Intervall  $[-7, -7]$  mit einem Inkrement von 1. Bereits in dieser Ansicht ist zu erkennen, dass der Verlauf sehr flach ist. Eine Schlucht bildet sich etwa in Nord-Süd-Richtung aus. Jeweils verzeichnet ist das lokale Minima (+) eins Plots, sowie die Höhenlinien<sup>5</sup>. Die farbliche Kodierung gibt den Fitnesswert an diesem Punkt an. Die Fitnesswerte wurden normiert und um den Wert ihres Minimums verschoben.

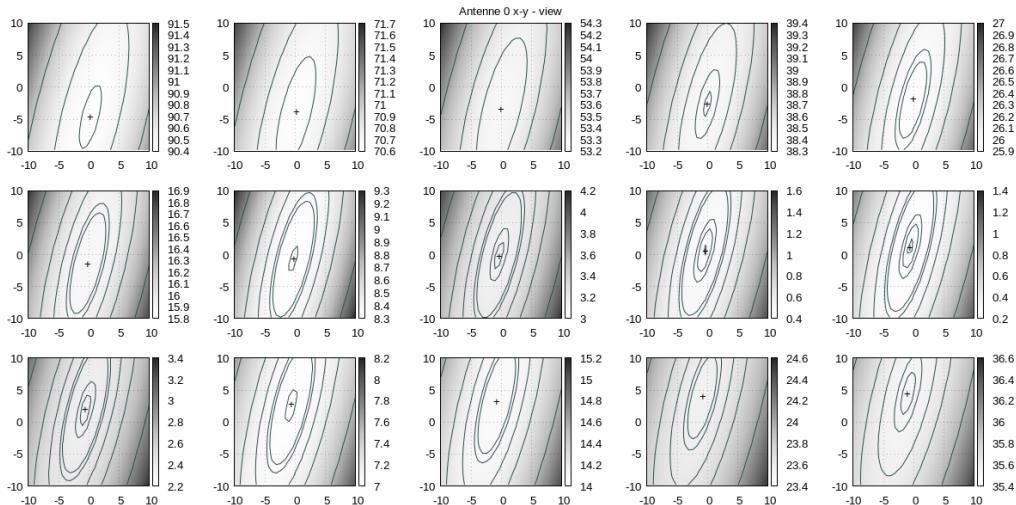


Abbildung 3.10.: Vergrößerung der Fitnessebenen der Antenne 1. Es ist hier deutlich zu erkennen, wie gering die Funktionen ansteigen. Das ist ein Problem für die meisten Algorithmen. Es bleibt zu klären wie sensitiv der Algorithmus auf diesen Umstand reagiert. Es wurde um das lokale Minima zentriert und der Bereich auf

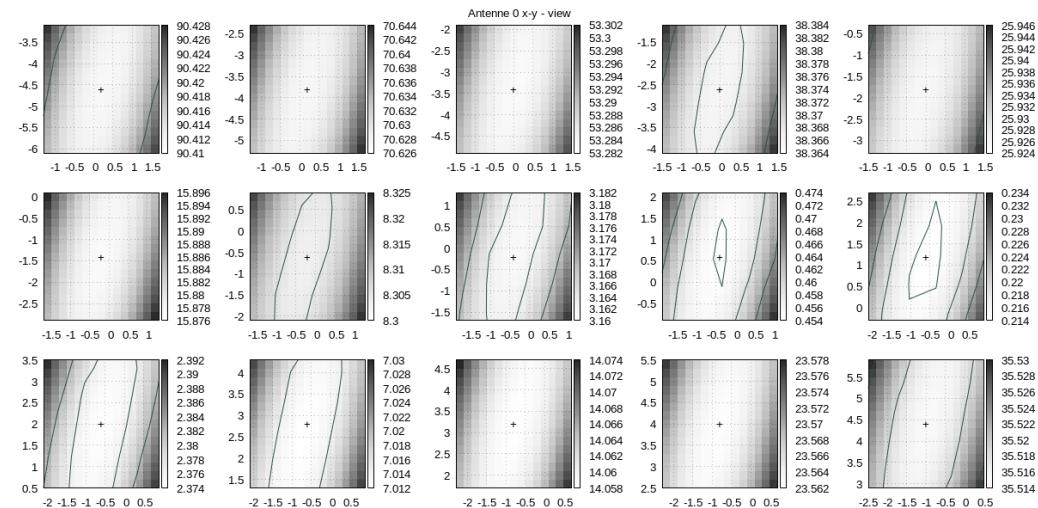
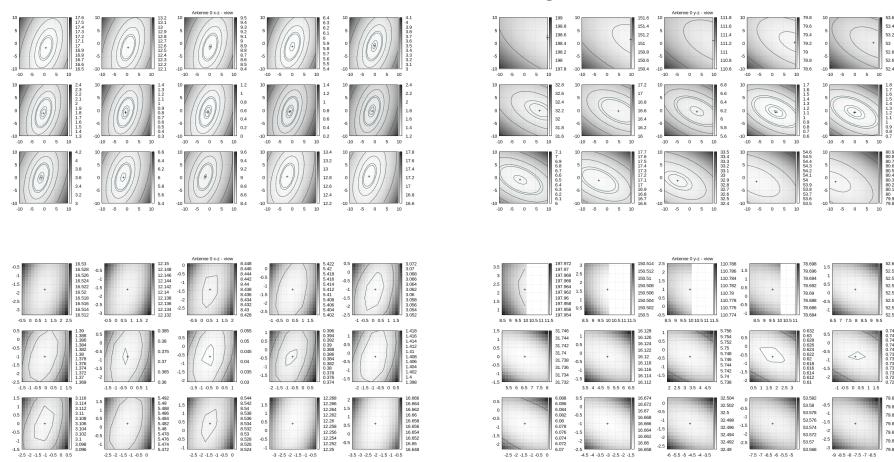


Abbildung 3.11.: Auf diesen Abbildungen zeigen sich die Fitness-Ebenen für die übrigen Ansichten, x-z und y-z. In der oberen Reihe sind Ebenen über den gesamten Bereich, in der Unteren vergrößert dargestellt. Zu erkennen ist ein zum Verlauf der x-y-Ebene sehr ähnliches Bild. Ein flaches, längliches Tal mit Minimum.



### 3.7. Realisierung der Kalibrierung

In diesem Abschnitt wird die Implementierung der Kalibrierung des Messaufbaus präsentiert und die Ergebnisse werden kurz zusammengefasst. Es werden zwei unterschiedliche Berechnungsverfahren vorgestellt. Zuerst die Berechnung über das SVD-Verfahren (numerisch), danach durch das CMA-ES-Verfahren (evolutionär). Es ist sinnvoll zu erwarten, dass beide Ergebnisse die gleichen Koordinaten liefern.

#### 3.7.1. Implementation

Der Ablauf der Kalibrierung ist in Abbildung 3.13 in Form eines Ablaufdiagramms dargestellt. Beschrieben werden die wesentlichen Schritte. Es sind sowohl Interaktion mit der Person enthalten, die die Kalibrierung durchführt, als auch die Schritte, die von den beteiligten Softwarekomponenten ausgeführt werden enthalten. Es wurden im Rahmen der Arbeit zwei unterschiedliche Wege implementiert, um ein Ergebnis für die Kalibrierung zu berechnen. Diese Wege werden im Folgenden vorgestellt und die Ergebnisse miteinander verglichen. Die Präsentation der Resultate wird vor allem dazu verwendet werden, die gewählte Form der Diagramme zu erläutern. Diese werden in den Ergebnissen des komplexeren Modells ebenfalls verwendet.

#### SVD

Das unter 2.1.2 vorgestellte Verfahren der Singular-Value-Decomposition kann dazu verwendet werden eine Lösung eines linearen Gleichungssystems zu berechnen. Das Modell, das zur Kalibrierung verwendet wird, ist ein Gleichungssystem der Form  $\mathbf{Ax} = \mathbf{b}$  und hat drei Gleichungen mit drei Unbekannten. Daher kann sofort eine Lösung mit dem Verfahren hergeleitet werden. Das Ergebnis eines Messaufbaus mit 3 Antennen ist in Tabelle 3.2 und in Abbildung 3.19 gezeigt. Die Implementation des Algorithmus stammt aus [28] und wurde für diese Arbeit anschafft.

#### CMA-ES

Da in dieser Arbeit der CMA-ES-Algorithmus eingesetzt wird und damit ohnehin eine Implementation vorgenommen wird, kann die Kalibrierung dazu verwendet werden die Umsetzung des Algorithmus zu verifizieren. Dazu vergleichen wir die Lösung der SVD-Methode mit der des CMA-ES.

Das über den evolutionären Algorithmus gefundene Ergebnis gleicht dem des SVD-Verfahrens (siehe Tabelle 3.2). Der SVD-Algorithmus ist um ein vielfaches effizienter<sup>6</sup> beim Lösen des Gleichungssystems. Die Gründe, warum an dieser Stelle die Berechnung mit dem evolutionären Verfahren durchgeführt und hier dargestellt wird sind folgende:

1. Die Komplexität ist gering, daher kann der Ablauf des evolutionären Verfahrens besser dargestellt und verstanden werden

---

<sup>6</sup>d.h. weniger Rechenzeit ist erforderlich

2. Der Vergleich der beiden Ergebnisse ermöglicht die Verifizierung der Implementation beider Verfahren.

Dem ersten Punkt kommt im Rahmen dieser Arbeit eine besondere Stellung zu. Es ist einfacher anhand dieses übersichtlichen Problems (mit nur drei Unbekannten) den Ablauf des Algorithmus sowie die Visualisierung der Ergebnisse zu veranschaulichen. Die verwendete Darstellung gleicht der, die später bei der Präsentation und Beurteilung der komplexeren Modell verwendet wird.

### 3.7.2. Ergebnis

Es werden nun die Ergebnisse der Kalibrierung vorgestellt. Für eine der vermessenen Antennenkonfigurationen sind in der folgenden Tabelle die Koordinaten der Antennen gezeigt. Die Visualisierung der Konfiguration zeigt die Abbildung 3.19. Eine Berechnung mit dem evolutionären Verfahren dauerte ca. 170 ms mit dem

<b>Antenne</b>	<i>x</i>	<i>y</i>	<i>z</i>	<i>d<sub>meas</sub></i>	<i>d<sub>result</sub></i>	$\epsilon_{abs}$	$\epsilon_{rel}$
1	0.48	-1.01	0.60	1,259	1,274	0,015	1,14%
2	-0.77	-1.04	1.34	1,894	1,872	-0,022	1,19%
3	1.52	-1.05	1.37	2,334	2,307	-0,027	1,15%
4	-0.92	-0.19	1.32	1,661	1,628	-0,033	2,01%
5	1.92	0.03	1.39	2,399	2,375	-0,024	1,01%
6	-0.55	1.09	1.43	1,851	1,887	0,036	1,93%
7	1.06	1.07	1.35	2,055	2,031	-0,024	1,19%
8	0.45	1.35	0.67	1,574	1,578	0,004	0,26%

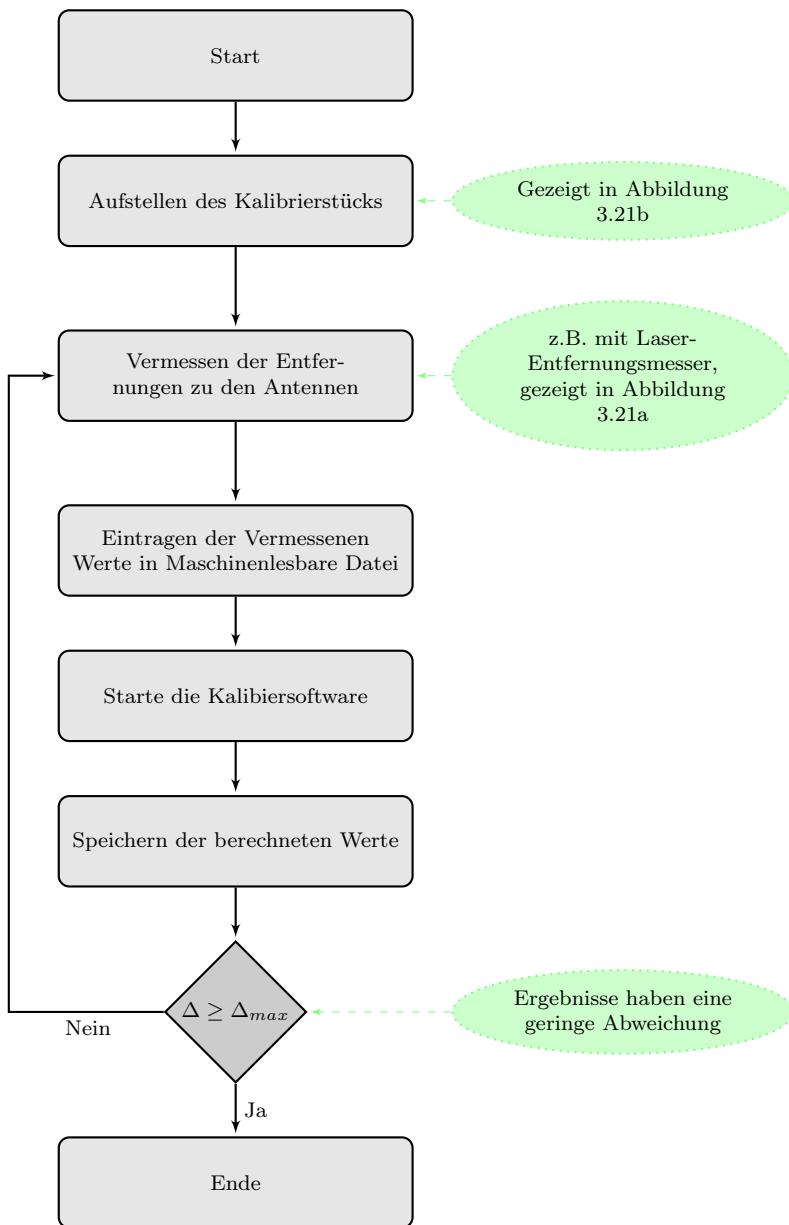
Tabelle 3.2.: Tabelle der finalen Antennenkoordinaten [m], berechnet mit dem in dieser Arbeit entwickelten Modell und dem SVD-Verfahren. Die Ergebnisse wurden auf zwei Nachkommastellen gerundet und sind identisch für beide Methoden. Die Spalte *d<sub>result</sub>* enthält die von der Berechnung gefundenen Distanz vom Referenzpunkt zur Antenne. Die Spalte *d<sub>meas</sub>* zeigt die gemessenen Werte. Die  $\epsilon$ -Spalten zeigen die Abweichung.

SVD-verfahren wurde eine Lösung in  $\leq 1$  ms gefunden. Für die in der Praxis eingesetzte Software wird es eine Implementation der Kalibrierung mit dem SVD-Verfahren geben. Das mit dieser Variante berechnete Ergebnis wird bei Bedarf mit einer Lösung des evolutionären Verfahrens verglichen. Das ermöglicht eine Build-In Verifikation der Kalibrierung.

### Visualisierung der Ergebnisse

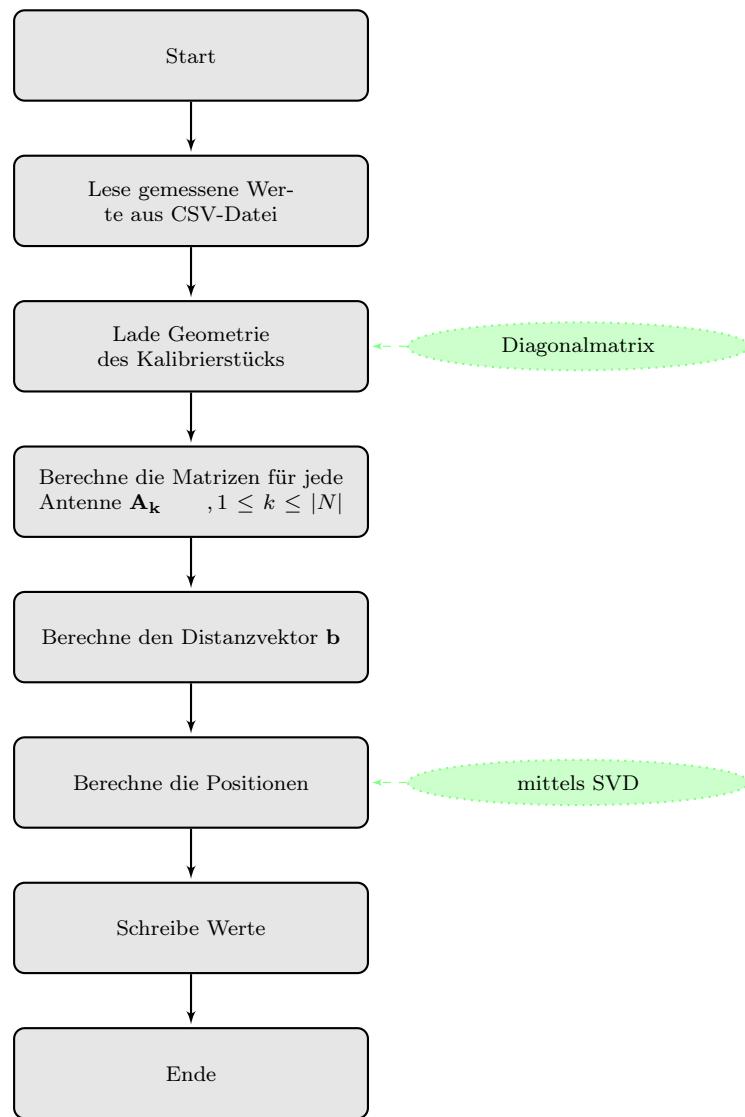
Für die in den folgenden Abbildungen präsentierten Ergebnisse wurden insgesamt 100 Durchläufe des Algorithmus erstellt. Die Ergebnisse wurden mit einem vom Algorithmus selbst erstellten  $\mu$  und  $\lambda$  gefunden. In Abbildung 3.15 wird eine statistische Auswertung der Ergebnisse gezeigt. In jedem Plot werden die Endwerte der

Abbildung 3.13.: Ablauf der Kalibrierung



Lösungen in einem sog. Boxplot gezeigt. Dabei wird die Verteilung mit Hilfe von Boxen dargestellt. Die Fähnchen der Boxen stellen die maximal- bzw. minimal- Werte dar. Die Größe der Boxen enthält das obere und untere Quartil der Daten,

Abbildung 3.14.: Ablauf der libCalibration



der horizontale Strich in der Box zeigt den Mittelwert der Daten. Ausreißer<sup>7</sup> in den Daten werden durch Punkte abseits der Box dargestellt.

Die Abbildung 3.15 zeigt den Verlauf der drei Objektvariablen ( $x, y, z$ -Koordinaten) sowie die Entwicklung der Fitness und des mittleren Sigmas. Als Darstellungsart wird der Linienplot verwendet und die Verläufe einzelner Lösungen überlagern sich

<sup>7</sup>Hier Werte die mindestens den 2-Fachen Wert des oberen- bzw. unteren-Quartils aufweisen

in diesem Plot. Das Abbruchkriterium war eine Fitness von  $\leq 10^{-25}$ . Für Darstellungszwecke wurde die  $x$ -Achse nach 500 Werten beschränkt, daher erreicht der Fitness-Plot diesen Wert in der Abbildung nicht. Der Verlauf ist typisch für den verwendeten Algorithmus. Deutlich zu erkennen ist eine Verbesserung des Ergebnisses mit steigender Zahl der Generationen. Der Verlauf der Variablen ist immer für den erfolgreichsten Nachkommen einer Generation dargestellt.

Abbildung 3.17 ist ein Scatter-Plot. Die Objektvariablen werden hier gegeneinander aufgetragen. Auf der Diagonalen befinden sich stets die Variable gegen sich selbst aufgetragen, daher zeigt sich dort immer eine Linie, bei streuenden Ergebnissen, bzw. ein einzelner Punkt, sollten die Ergebnisse nicht streuen. Der Plot ist praktisch um die Implementation des Algorithmus zu verifizieren. Er lässt Rückschlüsse auf Abhängigkeiten und Einflüsse der Objektvariablen zu. So können die Ergebnisse mit den Erwartungen an die Verläufe verglichen werden.

Abbildung 3.15.: Boxplot des Kalibierergebnis aus 100 Durchläufen. Im oberen Plot sind die  $x,y,z$ -Koordinaten gezeigt, diese landen in allen Durchläufen auf dem selben Ergebnis. Was nicht verwundert, das Problem ist eines der Art „drei Gleichungen und drei Unbekannte“. Die Streuung der Lösung zeigt sich in der Breite der Linien. Die unteren vier Plots zeigen die Anzahl der Evaluationen der Fitness-Funktion, den finalen Funktionswert, das Sigma für die Variablen und die Entfernung zum Referenzpunkt (v.l.n.r.). Das Ergebnis sind die Koordinaten für Antenne 1

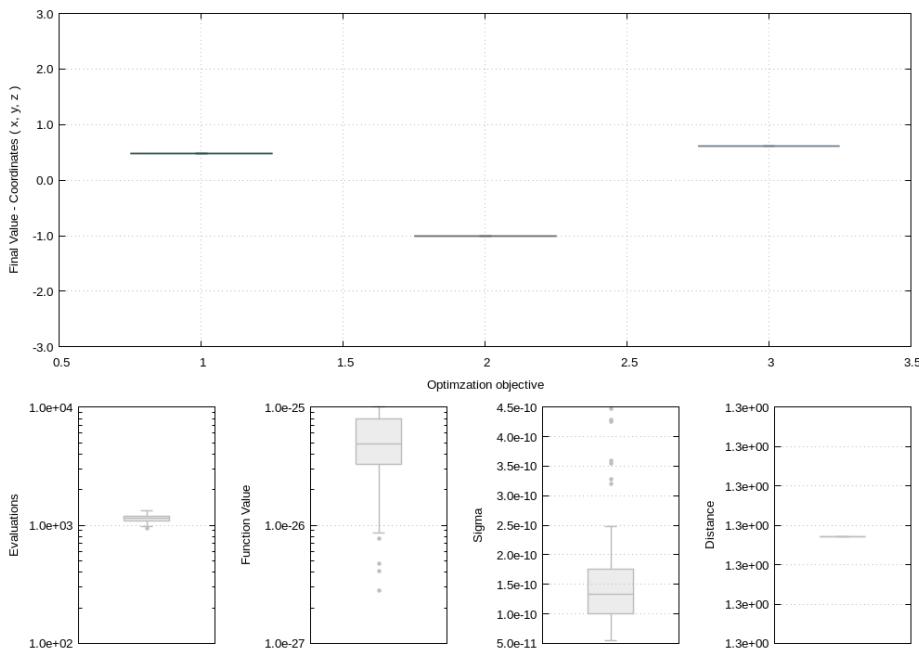


Abbildung 3.16.: Zu erkennen ist, dass nach ca. 300 Evaluationen der Zielfunktion keine großen Änderungen der Variablen zu erkennen sind. Bis zum erreichen des Abbruchkriteriums (Function Value  $\leq 10^{-25}$ ) werden noch ca 400 Evaluationen benötigt, vgl. korrespondierender Boxplot.

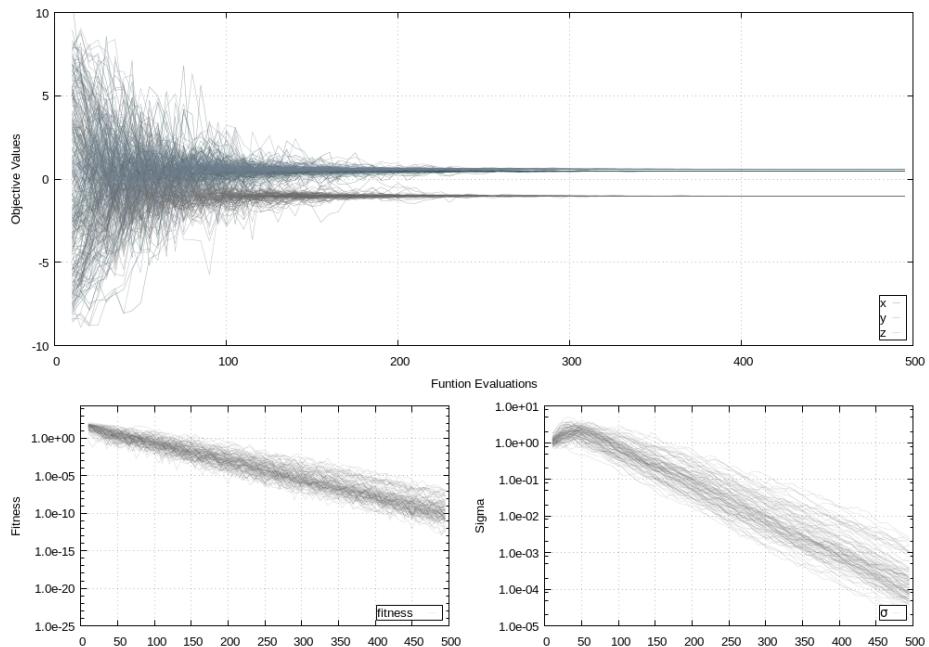
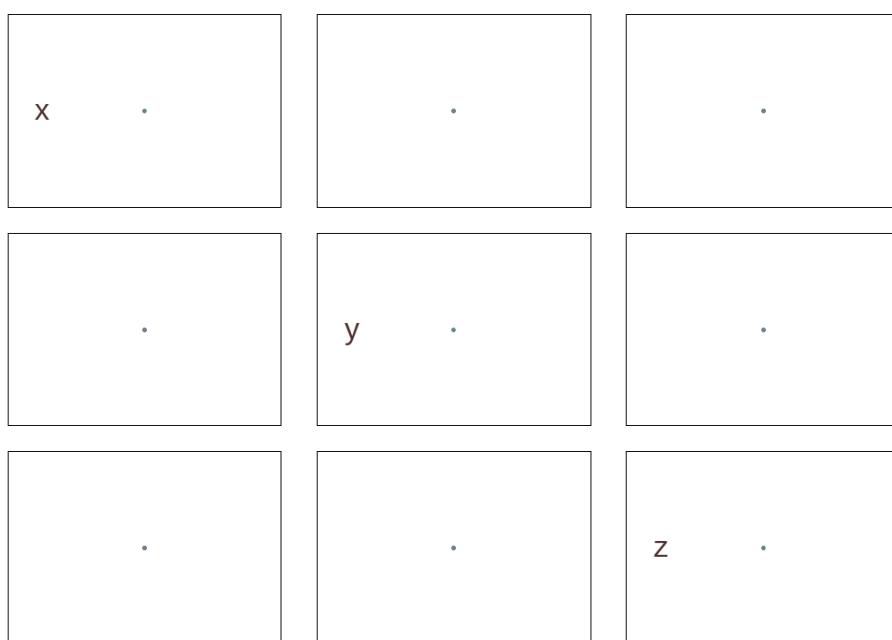


Abbildung 3.17.: Scatter-Plot der Ergebnisse der evolutionären Kalibrierung. Die Endergebnisse streuen in keiner Dimension, das wird aus dieser Darstellung deutlich.



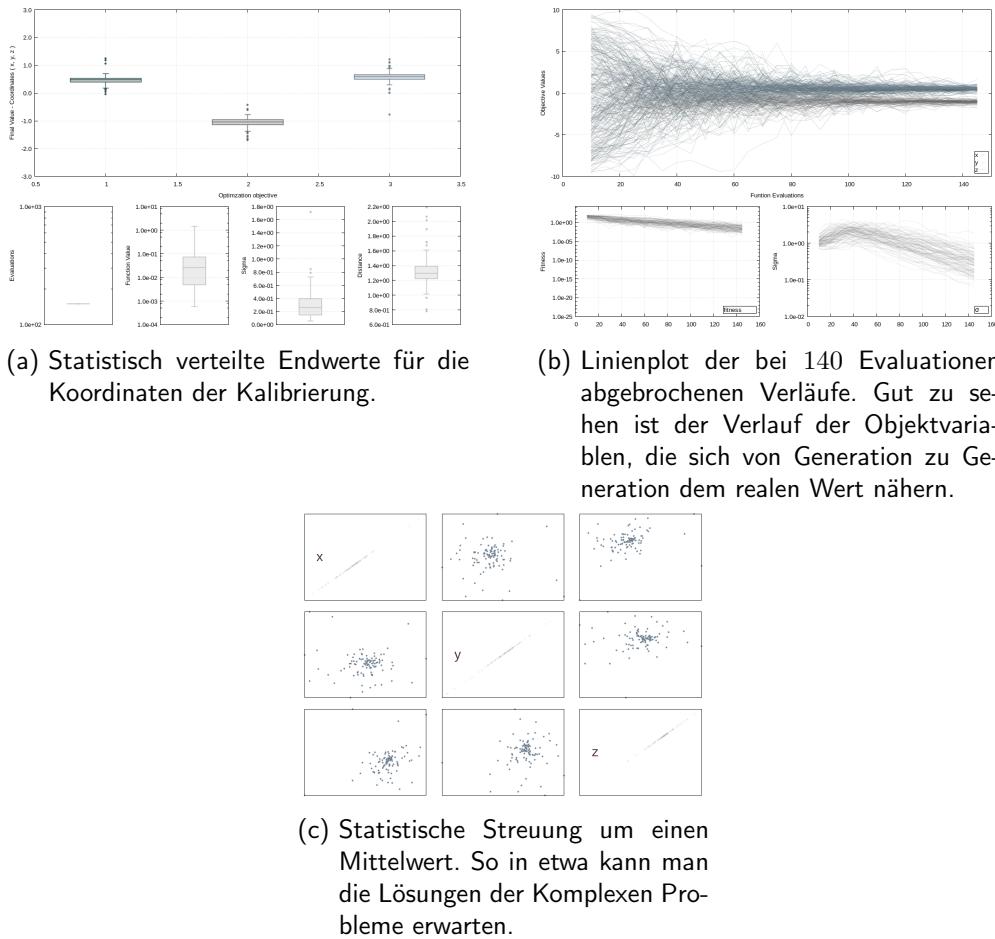


Abbildung 3.18.: Analog zu den Abbildungen 3.18b, 3.18a und 3.18c zeigen die Plots die gleichen Darstellungen. Hier gezeigt wird, wie sich eine statistische Verteilung in den Plots manifestieren würde. Um das zu demonstrieren wurde das Abbruchkriterium auf lediglich 150 Evaluationen der Zielfunktion eingestellt. Zu diesem Zeitpunkt können die Objektvariablen bereits einen passablen Wert erreicht haben oder noch abweichende Werte aufweisen (vgl. 3.16).

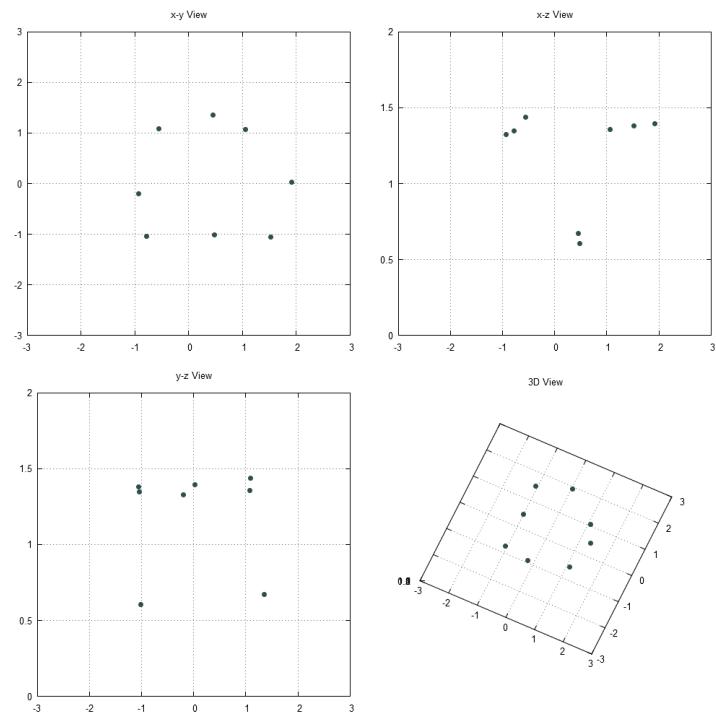


Abbildung 3.19.: Visualisierung des Kalibrierendergebnis. Abgebildet sind die gefundenen Antennenkoordinaten (Punkte) in drei Raumansichten. Die zusätzliche, dreidimensionale Ansicht dient der Übersicht. Das Ergebnis und die reale Anordnung decken sich sehr gut.

Abbildung 3.20.: Werkzeuge, die bei der Kalibrierung verwendet werden.



(a) Laser Distanzmesser



(b) Kalibrierstück mit vier Messpositionen.

## 3.8. Software

### 3.8.1. Shark

Shark ist eine Open-Source *C++*-Bibliothek für maschinelles Lernen und Optimierung [21]. Es implementiert Methoden für lineare und nicht lineare Optimierung, Kernel-basierte Lernverfahren, künstliche neuronale Netzwerke, und Weitere. Details der Umsetzung finden sich in [22]. Es wird sowohl in Forschung als auch in industriellem Umfeld eingesetzt und implementiert, nach eigenen Angaben, Algorithmen, die in anderen Bibliotheken nicht verfügbar sind. Shark baut auf den *Boost C++ Libraries* auf und verwendet CMake<sup>8</sup>. Dadurch ist es auf nahezu jeder Plattform verfügbar. Die Integration in ein Projekt ist sehr einfach und daher wird es in dieser Arbeit eingesetzt.

Eine Integration von Shark in dem Software Ökosystem ist aufgrund der vielen Features sehr sinnvoll. Weitere Entwicklungen können vom Funktionsumfang partizipieren.

### 3.8.2. Implementation

In diesem Abschnitt wird auf interessante Details der Softwareimplementation eingegangen. Es ist nicht möglich im vollen Umfang die Implementation der Software zu besprechen, dafür ist die Software zu Umfangreich<sup>9</sup>. Lediglich sollen hier die Implementationen verschiedener Evolutionsalgorithmen unter Verwendung von Shark gezeigt werden. Verschiedene Algorithmen wurden in Kapitel 2.4 besprochen. Es wird beispielhaft die Implementation einer Objektfunktion beschrieben, wie sie typischerweise in Shark vorgenommen wird.

#### Objektfunktion in Shark

Als Beispiel für die Implementation einer Objektfunktion wird im Folgenden das Modell der evolutionären Kalibrierung besprochen. Dieses Modell, bzw. diese Objektfunktion, hat eine überschaubare Komplexität und wurde bereits im Abschnitt 3.7 zur Veranschaulichung verwendet. Daher eignet es sich gut um die Implementation zu zeigen. Im Rahmen dieser Arbeit sind eine Vielzahl von Modellen entstanden, die nicht in aller Ausführlichkeit diskutiert werden können.

Das Listing 3.1 zeigt die Headerdatei für die Implementation einer Objektfunktion. Sofort ist zu erkennen, dass sie von der abstrakten Klasse *SingleObjectiveFunction* abgeleitet ist. Diese ist eine von Shark bereitgestellte Klasse. Sie beschreibt die Funktionen, die eine Objektfunktion implementieren muss, damit sie von Optimizern verwendet werden kann. Ein von dieser Klasse abgeleitetes Modell erlaubt die Verwendung in verschiedenen Algorithmen. Die Funktion:

---

<sup>8</sup>CMake (cross-platform make) ist ein plattformunabhängiges Programmierwerkzeug für die Entwicklung und Erstellung von Software.

<sup>9</sup>Für diese Arbeit wurden ca. 5000 Zeilen Quellcode erstellt – einzelne Modelle nicht mitgerechnet

```
double eval(const SearchPointType &p) const;
```

wird in von den Optimizern aufgerufen und implementiert die eigentliche Funktion des Modells.

```
void proposeStartingPoint(SearchPointType &x) const;
```

Diese Methode wird von einem Solver beim Start aufgerufen und liefert passende Startwerte für das Modell zurück. Der Rückgabewert der Funktion ist ein Vektor der Dimensionalität  $m\_numberOfVariables$ , dessen Werte in einem Intervall von  $[-10, 10]$  gleichverteilt sind. Der Wert für  $m\_numberOfVariables$  wird bei der Instanziierung des Modells, also beim Aufruf des Konstruktors, gesetzt. Dort wird auch das Flag *CAN\_PROPOSE\_STARTING\_POINT* gesetzt. Sie wird von den unterschiedlichen Optimizern verwendet, um zu erkennen, welche Features vom Modell unterstützt werden.

```
EvolutionaryCalibration( ) {
    m_numberOfVariables = Solve::ProblemDimensions::Calibration;
    m_features |= CAN_PROPOSE_STARTING_POINT;
}
```

Listing 3.1: Quellcodeschnipsel für die Deklaration einer Objektfunktion

```
struct EvolutionaryCalibration : public SingleObjectiveFunction {

    typedef AbstractOptimizer<shark::VectorSpace<double>,double,
        SingleObjectiveResultSet<typename shark::VectorSpace<double>:::
        PointType> > base_type;

    typedef typename base_type::ObjectiveFunctionType
        ObjectiveFunctionType;

    EvolutionaryCalibration( ) {
        m_numberOfVariables = Solve::ProblemDimensions::Calibration;
        m_features |= CAN_PROPOSE_STARTING_POINT;
    }

    /// brief From INameable: return the class name.
    std::string name() const
    { return "Evolutionary Calibration"; }

    std::size_t numberOfVariables() const{
        return m_numberOfVariables;
    }

    bool hasScalableDimensionality() const{
        return true;
    }

    void setNumberOfVariables( std::size_t numberOfVariables ){
        m_numberOfVariables = numberOfVariables;
    }

    void configure(const PropertyTree &node) {
        m_numberOfVariables = node.get("numberOfVariables", 51);
    }

    /**
```

```

 * Generate a starting value
 * @param[out] x The suggested search point
 *
 */
void proposeStartingPoint(SearchPointType &x) const {
    x.resize(numberOfVariables());

    for (unsigned int i = 0; i < 3; i++) {
        x(i) = Rng::uni(-10, 10);
    }
}

/**
 *
 */
double eval(const SearchPointType &p) const;

/**
 *
 */
void setParams( const NRmatrix< Doub > &M,
                 const NRvector< Doub > &v
                ) {
    setMat(M);
    setVec(v);
}

/**
 *
 */
void setMat( const NRmatrix< Doub > &M ) {
    A = M;
    A_isSet = true;
}

/**
 *
 */
void setVec( const NRvector< Doub > &v ) {
    b = v;
    b_isSet = true;
}

inline double mkII( const NRmatrix<Doub> &A, const double* x, const
                     NRvector<Doub> &b ) const;

private:
    std::size_t m_numberOfVariables;

    /**
     * The Matrices we need to solve the Problem */
    NRmatrix< Doub > A;
    bool A_isSet = false;

    /**
     * The b-vector needed to find a Solution */
    NRvector< Doub > b;
    bool b_isSet = false;
}

```

Die in Listig 3.2 gezeigte cpp-Datei ist die Implementation der oben beschriebenen Modellfunktionen. Aus diesen beiden Dateien ist ersichtlich, dass die Deklaration

des Modells sehr überschaubar ist. Praktische jedes Modell hat diese übersichtliche Struktur, was die Wartbarkeit enorm erhöht. Außerdem erlaubt es einen einfachen Austausch in der Implementation sowie die Verwendung in unterschiedlichen Algorithmen.

```
inline double EvolutionaryCalibration :: mkII( const NRmatrix<Doub> &A,
const double* x, const NRvector<Doub> &b ) const
```

Diese Funktion ist die eigentliche Berechnung des Gleichungssystems der Form  $\mathbf{Ax} = \mathbf{b}$ . Die Lösung wird an die Aufrufende Funktion übergeben. Als Eingabe wird der Variablenvektor *const double\*x* von Shark sowie die geom. Matrix **A** und der Distanzvektor **b** erwartet. Der vollständige Quellcode des Modells ist im Anhang B.1 und B.2 gelistet.

Listing 3.2: Quellcodeschnipsel für die Implementation einer Objektfunktion für verschiedene Optimizer in Shark

```
double EvolutionaryCalibration :: eval( const SearchPointType &p )
{
    m_evaluationCounter++;
    std :: vector<double> res;

    double x[ m_numberOfVariables ];

    x[ 0 ] = p[ 0 ];
    x[ 1 ] = p[ 1 ];
    x[ 2 ] = p[ 2 ];
    return mkII( A, x, b );
}

inline double EvolutionaryCalibration :: mkII( const NRmatrix<Doub>
    &A, const double* x, const NRvector<Doub> &b ) const
{
    double res;
    double prod_Ax[3] = { 0., 0., 0. };
    double x_[m_numberOfVariables];

    x_[0]=x[0];
    x_[1]=x[1];
    x_[2]=x[2];

    /* multiply the matrix with the vector */
    for( int i = 0; i < A.nrows(); i++ )
        for( int j = 0; j < A.ncols(); j++ )
            prod_Ax[i] += A[i][j]*x_[j];

    /* sum up */
    res = (prod_Ax[0] - b[0]) * (prod_Ax[0] - b[0]);
    res += (prod_Ax[1] - b[1]) * (prod_Ax[1] - b[1]);
    res += (prod_Ax[2] - b[2]) * (prod_Ax[2] - b[2]);

    return res;
}
```

### Process MkII – Finde die Lösung

Die Klasse *ProcessMkII* steht im Zentrum der Lösungsfindung. Sie erlaubt die Ausführung und Parametrierung von verschiedenen, in dieser Arbeit erprobten Testfunktionen. Die Klasse hat verschiedene Konstruktoren um den verschiedenen Modellen gerecht zu werden. Diese erwarten häufig unterschiedliche Parameter. Bei der Instanziierung werden gleichzeitig alle wichtigen Parameter übergeben. Dies zeigt folgendes Schnipsel Quellcode:

```
Solve :: Process_MkII process( A, b, names, MU, LAMBDA );
```

Das Listing 3.3 zeigt einen der implementierten Konstruktoren und seine Übergabeparameter. Dieser Konstruktor bereitet das Objekt auf das Model 'WholeToma-toMkII' vor.

Listing 3.3: Quellcodeschnipsel für die Implementation einer Objektfunktion für verschiedene Optimizer in Shark

```
Process_MkII(
    std :: vector<NRmatrix< Doub >> Mats ,
    std :: vector<NRvector< Doub >> Vectors ,
    std :: vector<std :: string> Names ,
    std :: vector<std :: vector<int>> IDs ,
    double Epsilon
) : A( Mats ), b( Vectors ), names( Names ), idxs( IDs ), epsilon(
    Epsilon )
{
    init();
}
```

Ein wichtiger Schritt ist noch für alle Solver durchzuführen. Eine Initiierung des Zufallszahlengenerator muss durchgeführt werden. Das übernimmt die *init*-Funktion, gezeigt in Listing 3.4. Dort wird über den Aufruf 'shark::Rng::seed( seed )' der Generator initialisiert. Die Variable *seed* wird über die Systemzeit gefüllt. Die Funktionen des in C++11 eingeführte Namespace 'std::chrono' erleichtern diese Aufgabe durch den einfachen Zugriff auf Zeitfunktionen.

Listing 3.4: Quellcodeschnipsel für die Implementation einer Objektfunktion für verschiedene Optimizer in Shark

```
void init()
{
    // Adjust the floating-point format to scientific and increase
    // output precision.
    std :: cout . setf( std :: ios_base :: scientific );
    std :: cout . precision( 10 );

    std :: chrono :: time_point<std :: chrono :: system_clock> now = std :: chrono
        :: system_clock :: now();

    auto duration = now.time_since_epoch();
    /* init the seed */
    auto seed = std :: chrono :: duration_cast<std :: chrono :: milliseconds>(
        duration ). count();

    shark :: Rng :: seed( seed );
}
```

}

Nach erfolgreicher Instanziierung kann einfach das gewünschte Modell ausgeführt werden. Beispielsweise das Modell der evolutionären Kalibrierung wird unter Angabe der Problemdimension ausgeführt:

```
process.EvoCal( Solve::ProblemDimensions::Calibration );
```

Intern wird im Anschluss an diesen Aufruf eine Instanz des Solvers erstellt, hier: *shark::CMA*. Nach der Initialisierung des Solvers kann dieser ausgeführt werden. Das zeigt das Listing 3.5. Dort wird ein anderes Modell '*WholeTomatoMkII*' ausgeführt. Anhand der *while*-Schleife kann man die Konvergenzkriterien erkennen. Der Algorithmus läuft so lange, bis das gewünschte epsilon unterschritten oder die maximale Anzahl an Evaluationen erreicht ist.

Listing 3.5: Quellcodeschnipsel für die Implementation einer Objektfunktion für verschiedene Optimizer in Shark

```
int WholeTomatoMkII( int dimension ) {
    auto dim = Solve::ProblemDimensions::WholeTomatoMkII;
    dim += dimension;
    PRPSEvolution::Models::WholeTomatoMkII model( dim );

    model.setNumberOfVariables( dim );
    model.setParams( A, b, names );

    /* init the algorithm */
    shark::CMA cma;
    cma.init( model );

    do {
        cma.step( model );
    } while( cma.solution().value > epsilon
        && model.evaluationCounter() < maxEvaluations );
}
```

### 3.8.3. Paralleler Ablauf

Die *ProcessMkII*-Klasse ist Multithreading-fähig. Eine Implementierung wurde im Rahmen der Arbeit durchgeführt. Durch das parallele Ablaufen verschiedener Optimierungen kann die Performance wesentlicher verbessert werden und so die Multi-Kern-Architektur aktueller Rechner ausgenutzt werden. Einen großen Beitrag zur schnellen Umsetzung lieferte der neue Standard C++11. Dieser implementiert ein High-Level Konstrukt für Threads und Tasks, sodass es einfacher ist, Abläufe zu Synchronisieren und Ergebnisse abzufragen. Zusätzlich ist die Implementation sehr leicht zu verwenden, wie im folgenden Listing gezeigt:

Listing 3.6: Erstellen von mehreren Threads bei gleichzeitiger Übergabe der Parameter

```
/* result vector; Solve::solveresult_t defines the return from the
model */
```

```

std::vector<std::future<Solve::solveresult_t<ChromosomeT<double>,
    ChromosomeT<double>,Doub>>> results;

/* create tasks */
for( int Solution = 0; Solution < NO_OF_SOLUTIONS; Solution++ ) {

    t_0 = steady_clock::now();

    results.push_back( std::async( std::launch::async ,
        &Solve::Process::findSolution<Solve::solveresult_t<
            ChromosomeT<double>,Doub>>,
        &process ,
        A,
        b,
        Solve::ESSStrategy::OnePlusOne,
        duration_cast<microseconds>(t_1-t_00).count() ) );

    results.push_back( std::async( std::launch::deferred , &Solve::
        Process::findSolution , &process , A, b ) );

    t_1 = steady_clock::now();
}

```

Das Abfragen der Ergebnisse ist ähnlich einfach:

**Listing 3.7: Abfragen der Asynchronen Ergebnisse mit Hilfe des auto-Datentyp.**  
**Ausgabe des Ergebnis in einer Datei.**

```

for( auto res = results.begin(); res != results.end(); ++res ) {
    while(res->wait_for(std::chrono::seconds(0)) != future_status::ready);

    auto r = res->get();
    f_fitness << r.iterations << " " << r.fitness << " in: " << r.
        duration << " (us)" << " " << r.converged << std::endl;

    for( auto values : r.valCont )
        f << values << "\t";
    for( auto values : r.valDis )
        f << values << "\t";

    f << std::endl;
}

```

### 3.8.4. Schnittstellen für Dateneingabe

Im Folgenden wird die implementierte Schnittstelle besprochen, mit der die Daten unter den Programmteilen ausgetauscht werden können. Die Schnittstelle umfasst im Wesentlichen zwei Teile:

1. Eingabe für die gemessenen Phasenwerte
2. Ausgabe für die ermittelten Wellenzahlen

Hinzukommt eine pseudo-Schnittstelle, über die die Systemparameter eingelesen werden können. Die aktuelle Implementation liest lediglich eine CSV-Eingabedatei mit den Systemparametern in eine entsprechende Struktur ein. Diese Parameter

stehen anschließend dem PRPS-Evolutions-System zur Verfügung. Die Kommunikation zwischen dem PRPS-Dienst und dem PRPS-Evolutions-System ist einfach. Der Dienst teilt die gemessenen Phasendaten mit, die vom PRPS-Evolutions-System für die Berechnung der Wellenzahlen verwendet werden. Das bedeutet, das der Ablauf des Auffindens der Wellenzahl vom PRPS-Dienst als 'Black-Box' angesehen wird.

### 3.8.5. Ablaufdiagramme

Vieles der Funktionalität der Software ist zu umfangreich für dieses Dokument. Es kann in diesem Rahmen keine vollständige Besprechung des Quellcodes durchgeführt werden. Wichtige Funktionalitäten und Abläufe werden in Ablaufplänen zusammenfassend dargestellt.

Abbildung 3.22.: Prozessschritte nach Start des Programms. Mit dem Punkt "Ende" ist nicht das Ende des Programms gemeint, sondern das Ende des Programmstarts. Nach diesen Schritten stehen alle statisch generierbaren verfügbaren Daten für das Programm zur Verfügung.

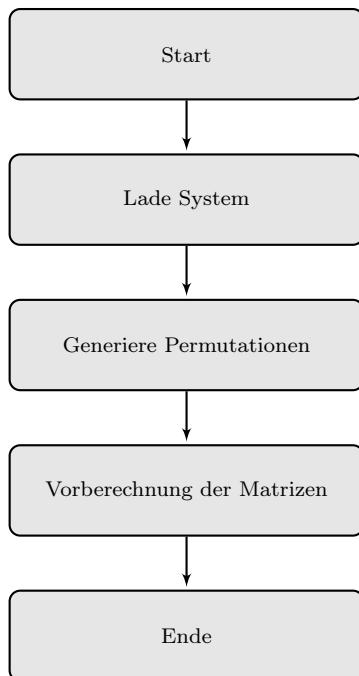


Abbildung 3.23.: Schritte die zur Lösungsfindung abgearbeitet werden. Dies ist ein Teil des Hauptprogramms, der die erstellten Informationen vom Programmstart in ein Modell zum Auffinden einer Lösung übergibt. Es werden eine beliebige Anzahl an Lösungen generiert.

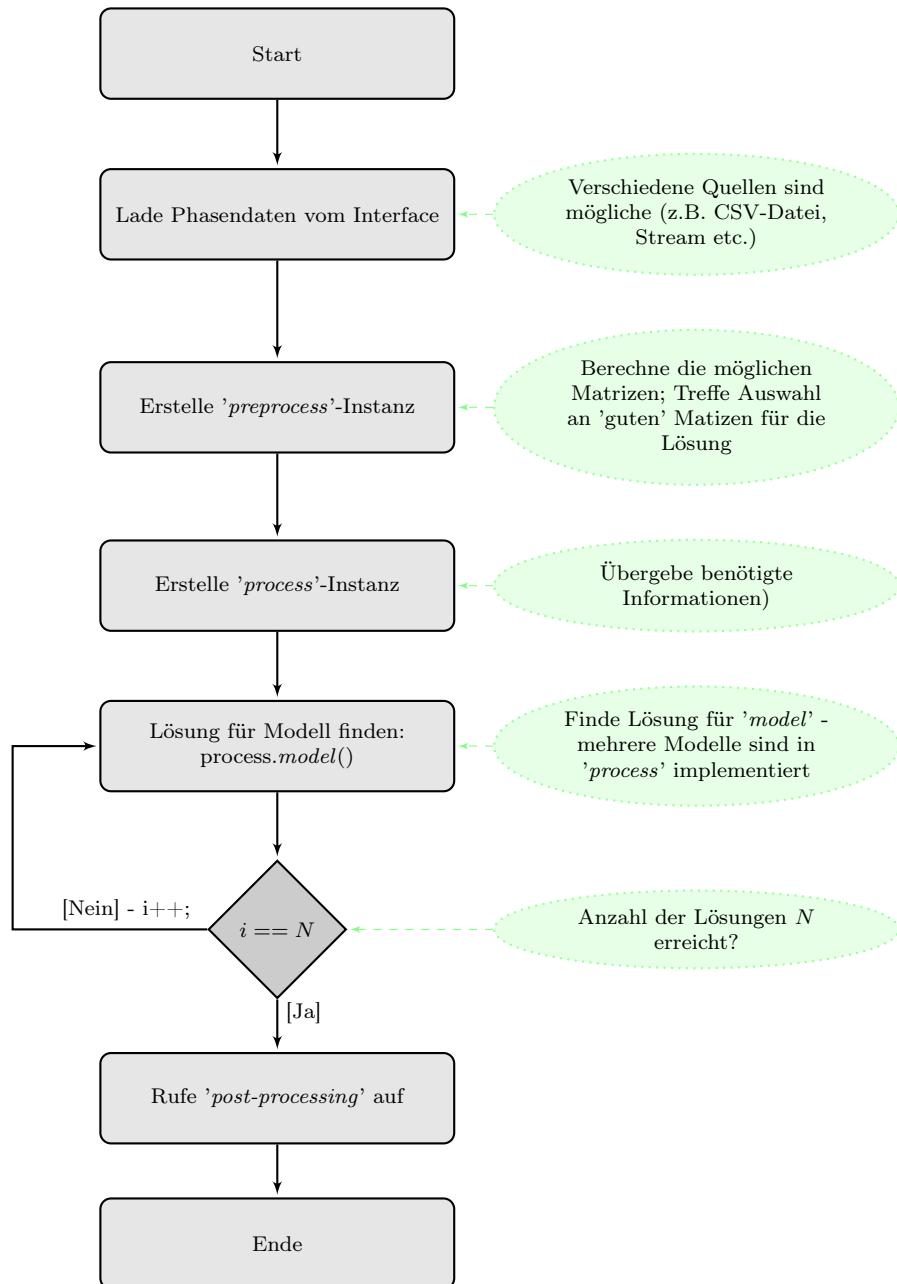
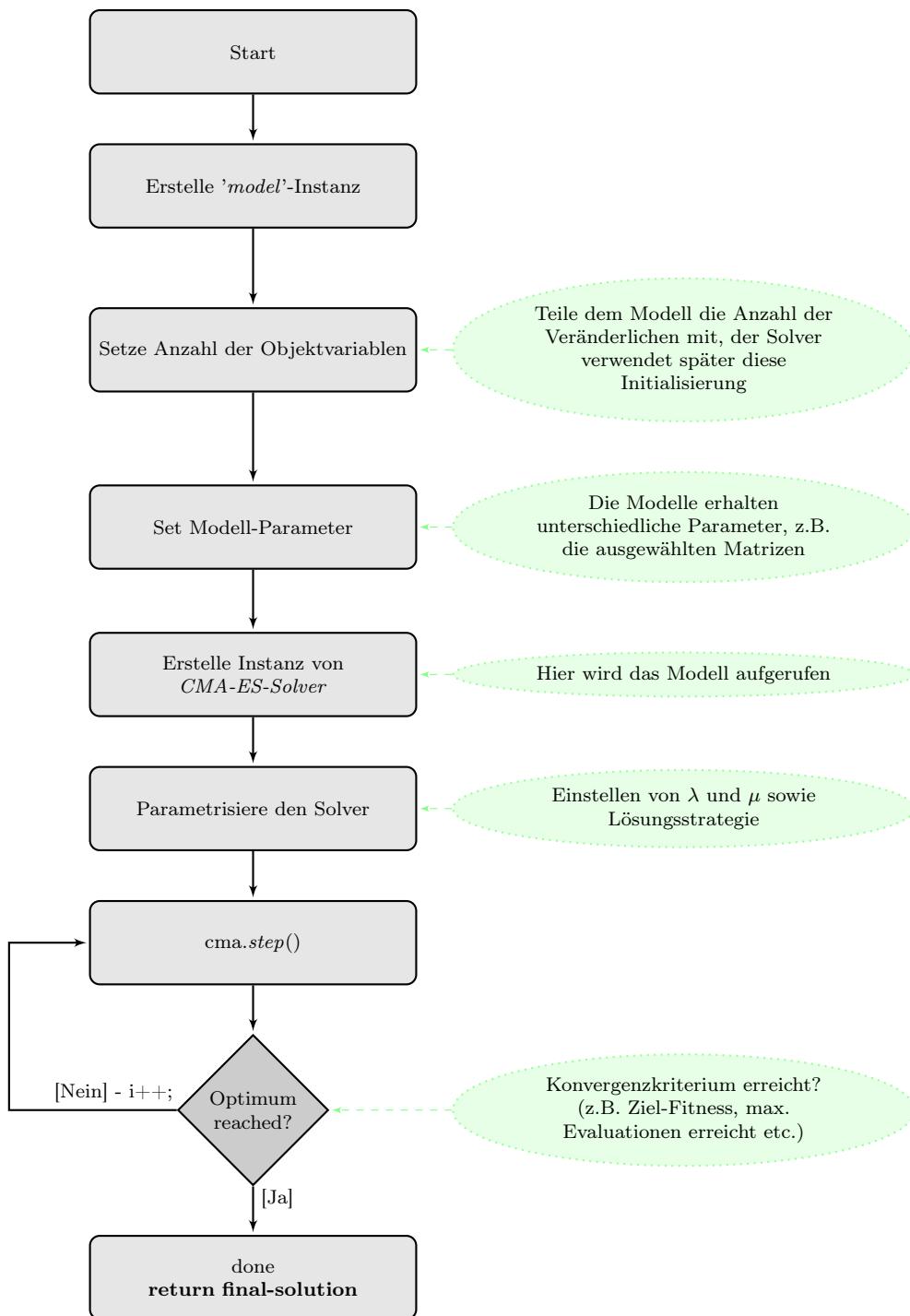


Abbildung 3.24.: Die Grafik zeigt den Ablauf im Modell um eine Lösung zu finden. Es werden so lange Evolutionsschritte durchgeführt bis das Konvergenzkriterium erreicht wird.



# 4. Ergebnisse

In diesem Kapitel werden die Ergebnisse der Arbeit präsentiert. Zuerst werden künstliche Werte für die gemessene Phase als Eingabe für den Algorithmus verwendet. Darauf folgend werden reale Messdaten als Input verwendet und die Resultate verglichen.

## 4.1. Ergebnisse

Die Ergebnisse der Arbeit werden hier vorgestellt. Das Modell aus 3.2 wird in verschiedenen Experimenten untersucht. Es wird analysiert, wie gut die Lösbarkeit registrierter und unregistrierter Probleme ist. Dazu werden zuerst für jede Referenzantenne immer eine mögliche Konfiguration gewählt und das Ergebnis aus  $M$ -Durchläufen untersucht. Im Anschluss

Zunächst werden die Ergebnisse der idealen Messwerte vorgestellt. Dazu wurden eine Reihe von Punkten definiert, von denen die idealen Phasenwerte ermittelt wurden.

### Unregistriertes Problem

Unregistrierte Probleme nennt man im Jargon der EO Probleme ohne Randbedingungen. Im Folgenden werden die Modelle analysiert, ohne Randbedingungen vorzugeben.

### Registriertes Problem

Den Objektfunktionen werden im Folgenden Randbedingungen vorgegeben. Folgende Randbedingungen werden umgesetzt:

Nummer	Bedingung
1	$n_k \geq 0$
2	$x, y, z \geq  5 $
3	$x, y, z \leq  25 $

Tabelle 4.1.: Die implementierten Randbedingungen für das Problem.



## 5. Diskussion

Vergleich der Ergebnisse mit den Zielen der Arbeit.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellen-tesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in

hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetur.

# 6. Schluss

## 6.1. Ausblick

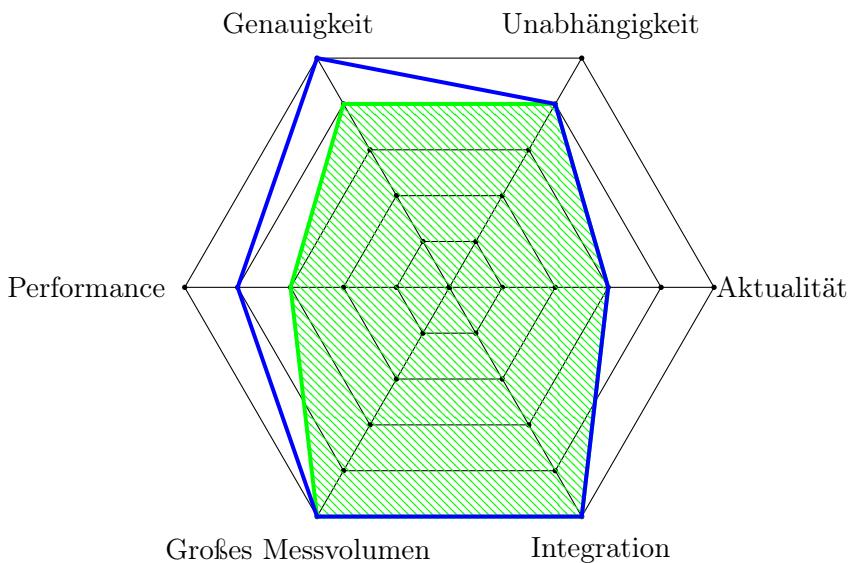
Wie in den Ergebnissen gezeigt, ist die Lösung nicht immer zufriedenstellend gefunden worden. Die Ergebnisse dieser Arbeit weisen relativ große Abweichungen in der gefundenen z-Koordinate auf. Das wird aus den in den Ergebnissen präsentierenden Grafiken deutlich. Wie in der Analyse der Komplexität, Kapitel 3.6, geschildert, ist der Suchraum sehr komplex. Es ist möglich durch vorhandene Informationen die Problemdimension zu reduzieren. Ein darauf basierendes Modell wurde in dieser Arbeit nicht untersucht. Die Umsetzung eines solchen Modells ist mit dem in dieser Arbeit geschaffenen Rahmen jedoch ohne großen Aufwand möglich. Es muss lediglich eine neue Objektfunktion erstellt werden.

Ungünstige Umstände (schlechte Konditionierung, geringe Anzahl an Antennen etc.) können zu den unzureichenden Ergebnissen beitragen. Zur Zeit wird in einem solchen Fall ein statistischer Ansatz durchgeführt. Dazu werden mehrere Lösungsversuche durchgeführt und anschließend der Median-Wert der Variablen für die Abschätzung genutzt. Das verbessert das Ergebnis zu Lasten der Ausführungszeit. Die Zeit bis ein Resultat vorliegt kann sich zum Teil auf mehrere Sekunden vergrößern. Das liegt außerhalb der Anforderungen. Hier muss durch weitere empirische Analysen ein gutes Mittelmaß gefunden werden.

Es ist im Weiteren auch nicht sinnvoll die in dieser Arbeit gefundene Implementation als vollständig richtig anzunehmen. Obwohl auf die Verifikation einzelner Programmteile genau geachtet wurde (siehe z.B. Kapitel 3.7) konnte die Implementation des Modells mit keiner Referenz verglichen werden. Alle Eingaben in das Modell, wie die statischen Matrizen und der Ergebnisvektor, wurden von Hand berechnet und konnten so geprüft werden. Allerdings gibt es schlicht keine andere Umsetzung des Modells. Es ist ratsam, die Implementation einem Review zu unterziehen.

Im Rahmen dieser Arbeit konnte eine automatische, auf Messungen basierende Kalibrierung nicht mehr erprobt werden. Das hier vorgestellte Konzept der Entfernungsfindung eignet sich dazu, ein solches System zu implementieren. Dazu könnte ein Phantom mit einer bekannten geometrischen Proportion mit Tags ausgestattet werden. Die Antennen würden die Phasenwerte ausgeben und die Positionen der Antennen würde optimiert werden können. Das bedeutet die Berechnung des umgekehrten Falls zur Entfernungsfindung eines Tags.

Abbildung 6.1.: Grafische Übersicht der Ergebnisse gegen die gestellten Anforderungen (blau) an diese Arbeit. Der grüne Bereich stellt die erreichten Ziele dar. Wie bereits Diskutiert, werden die Anforderungen an die Genauigkeit nicht gut erfüllt. Auch die Performance ist nicht zufriedenstellend erreicht, die Gründe wurden bereits angegeben. In den anderen Bereichen zeigt sich eine gute Deckung mit den Anforderungen.



### Weitere Anwendungsmöglichkeiten

Das Modell, das in dieser Arbeit entwickelt wurde, ermöglicht eine Anwendung Abseits der Entfernungsfindung. Dieser Ansatz kann zur Lösung eines anderen Problems beitragen, das durch die freie Anordnung der Antennen entsteht. Wenn mehrere Tags im Raum identifiziert werden, muss die Steuerung der Antennenumschaltung<sup>1</sup> zur Zeit alle Antennen in einem Round-Robin-Verfahren umschalten. Nach einer Umschaltung kann es dazu kommen, dass keine der Antennen einen der zuvor identifizierten Tags "sieht". Da die Umschaltung zur Zeit zufällig zur nächsten Antenne übergeht, werden auch Antennen genommen, die keine günstige Positions berechnung erlauben. Die mögliche Lösung dafür ist, die Bestimmung der Kondition für jede Konfiguration aus vier Antennen dazu zu verwenden, eine intelligente Steuerung der Umschaltung zu ermöglichen. Dazu würden aus dem gewählten Antennen aufbau die Antennen kombinationen ihrer Konditionszahl nach Aufsteigend<sup>2</sup> sortiert und diese Antennen kombinationen von der Antennen umschaltung bevorzugt. Dadurch werden stets die am besten konditionierte Kombinationen gewählt.

<sup>1</sup>Diese bestimmt von welcher Antenne gerade versucht wird die Tags zu lesen – es können nicht gleichzeitig alle Antennen gelesen werden

<sup>2</sup>zur Erinnerung, gute Kondition = kleine Konditionszahl

# A. Abbildungen

## A.1. Messaufbauten

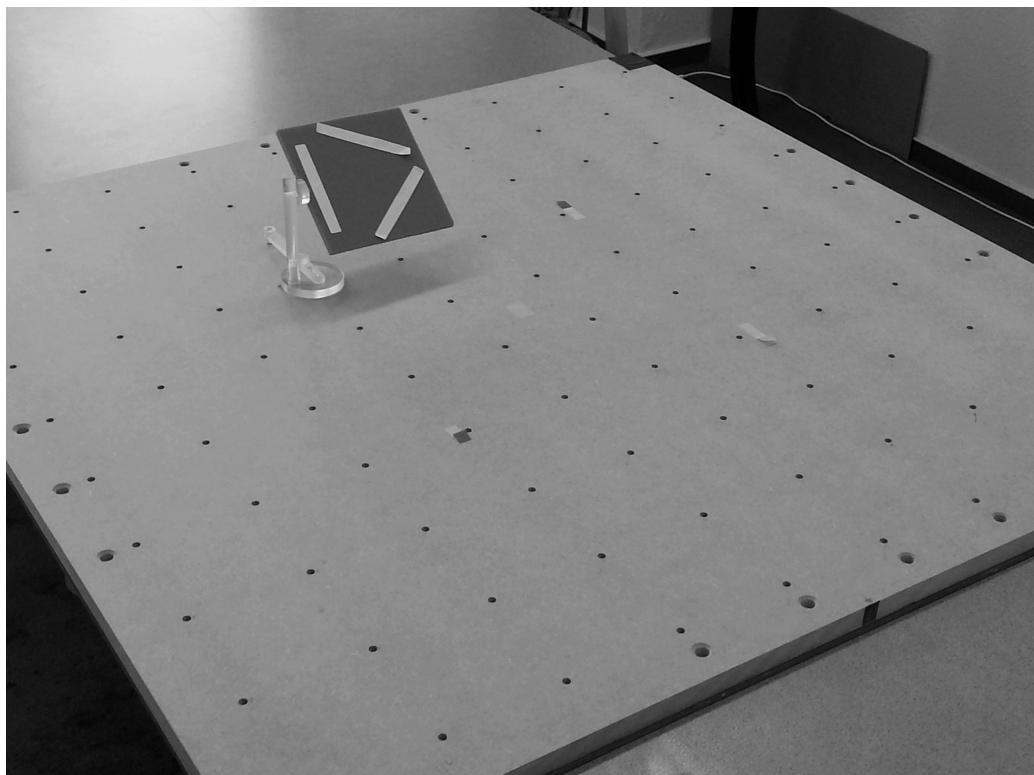


Abbildung A.1.: Reproduzierbare Aufstellung für die Messungen. Im Bild zu erkennen ist ein Taghalter mit drei Tags und eine Grundplatte, die eine Fläche von  $1 \times 1$  Meter abdeckt. Dabei beträgt die Distanz zwischen den einzelnen Positionen 10 cm, sodass sich insgesamt 100 Messpunkte anfahren lassen. Die Positionen sind aufsteigend nach Richtung der Ebenen benannt. Der *Ursprung* befindet sich in der Linken unteren Ecke (vom Bild nicht erfasst). Der Aufsteller mit den Tags befindet sich demnach an Position (3, 6).

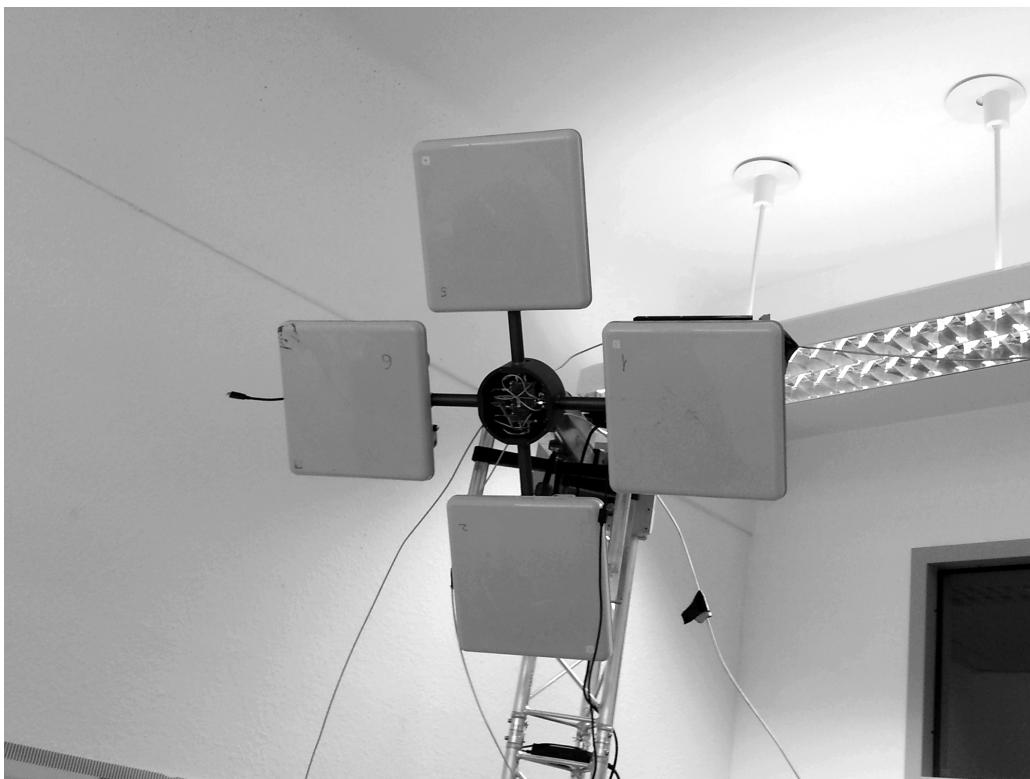


Abbildung A.2.: Messaufbau mit *Blick* auf die reproduzierbare Aufstellung. Antenne 1 ist die oben gelegene Antenne. Im Uhrzeigersinn ordnen sich die restlichen Antennen an. Zur Orientierung: Die reproduzierbare Aufstellung befindet sich in Richtung des Betrachters, von dem Aufbau aus gesehen.

## B. Quellcodeauszüge

### B.1. ObjectiveFunktion - Evolutionary Calibration Header

Vollständiger Quellcode der EvolutionaryCalibration.h

```
1 #ifndef _EVOLUTIONARY_CALIBRATION_H_
2 #define _EVOLUTIONARY_CALIBRATION_H_
3
4 #include <shark/ObjectiveFunctions/AbstractObjectiveFunction.h>
5 #include <shark/Algorithms/DirectSearch/CMA.h>
6 #include <shark/Algorithms/AbstractOptimizer.h>
7 #include <shark/Rng/GlobalRng.h>
8 #include <nr3/nr3.h>
9 #include "../solve.h"
10
11 namespace PRPSEvolution {
12     namespace Models {
13         using namespace shark;
14
15         /**
16         *
17         */
18     struct EvolutionaryCalibration : public SingleObjectiveFunction {
19
20         typedef AbstractOptimizer<shark::VectorSpace<double>, double,
21             SingleObjectiveResultSet<typename shark::VectorSpace<double>::PointType>> base_type;
22
23         typedef typename base_type::ObjectiveFunctionType
24             ObjectiveFunctionType;
25
26         EvolutionaryCalibration() {
27             m_numberOfVariables = Solve::ProblemDimensions::Calibration;
28             m_features |= CAN_PROPOSE_STARTING_POINT;
29         }
30
31         /// \brief From INameable: return the class name.
32         std::string name() const
33         { return "Evolutionary Calibration"; }
34
35         std::size_t numberOfVariables() const{
36             return m_numberOfVariables;
37         }
38
39         bool hasScalableDimensionality() const{
40             return true;
41         }
42
43         void setNumberOfVariables( std::size_t numberOfVariables ){
44             m_numberOfVariables = numberOfVariables;
45         }
46     }
```

```

45  void configure(const PropertyTree &node) {
46      m_numberOfVariables = node.get("numberOfVariables", 51);
47  }
48
49 /**
50 * Generate a starting value
51 * @param[out] x The suggested search point
52 *
53 */
54 void proposeStartingPoint(SearchPointType &x) const {
55     x.resize(numberOfVariables());
56
57     for (unsigned int i = 0; i < 3; i++) {
58         x(i) = Rng::uni(-10, 10);
59     }
60 }
61
62 /**
63 *
64 */
65 double eval(const SearchPointType &p) const;
66
67 /**
68 *
69 */
70 void setParams( const NRmatrix< Doub > &M,
71                 const NRvector< Doub > &v
72                 ) {
73     setMat(M);
74     setVec(v);
75 }
76
77 /**
78 *
79 */
80 /**
81 void setMat( const NRmatrix< Doub > &M ) {
82     A = M;
83     A_isSet = true;
84 }
85
86 /**
87 *
88 */
89 /**
90 void setVec( const NRvector< Doub > &v ) {
91     b = v;
92     b_isSet = true;
93 }
94
95 inline double mkII( const NRmatrix<Doub> &A, const double* x, const
96                     NRvector<Doub> &b ) const;
97
98 private:
99     std::size_t m_numberOfVariables;
100
101    /** The Matrices we need to solve the Problem */
102    NRmatrix< Doub > A;
103    bool A_isSet = false;
104
105    /** The b-vector needed to find a Solution */
106    NRvector< Doub > b;
107    bool b_isSet = false;

```

```
108 | };  
109 |  
110 |     ANNOUNCE_SINGLE_OBJECTIVE_FUNCTION( EvolutionaryCalibration ,  soos::  
|         RealValuedObjectiveFunctionFactory );  
111 |  
112 | }  
113 | } /* end namespace */  
114 |  
115 #endif /* _EVOLUTIONARY_CALIBRATION_H_ */
```

## B.2. ObjectiveFunktion - Evolutionary Calibration Source

Vollständiger Quellcode der EvolutionaryCalibration.cpp

```

1 #include "EvolutionaryCalibration.h"
2
3 namespace PRPSEvolution {
4     namespace Models {
5         using namespace shark;
6
7         double EvolutionaryCalibration::eval( const SearchPointType &p )
8             const
9         {
10             m_evaluationCounter++;
11             std::vector<double> res;
12
13             double x[ m_numberOfVariables ];
14
15             x[ 0 ] = p[ 0 ];
16             x[ 1 ] = p[ 1 ];
17             x[ 2 ] = p[ 2 ];
18             return mkII( A, x, b );
19
20         }
21
22         inline double EvolutionaryCalibration::mkII( const NRmatrix<Double>
23             &A, const double* x, const NRvector<Double> &b ) const
24         {
25             double res;
26             double prod_Ax[3] = {0.,0.,0.};
27             double x_[m_numberOfVariables];
28
29             x_[0]=x[0];
30             x_[1]=x[1];
31             x_[2]=x[2];
32
33             /* multiply the matrix with the vector */
34             for( int i = 0; i < A.nrows(); i++ )
35                 for( int j = 0; j < A.ncols(); j++ )
36                     prod_Ax[i] += A[i][j]*x_[j];
37
38             /* sum up */
39             res = (prod_Ax[0] - b[0]) * (prod_Ax[0] - b[0]);
40             res += (prod_Ax[1] - b[1]) * (prod_Ax[1] - b[1]);
41             res += (prod_Ax[2] - b[2]) * (prod_Ax[2] - b[2]);
42
43             return res;
44         }
45
46     } /* !namespace */
47 } /* !namespace */

```

# C. Gnuplot Skripte

## C.1. Boxplot

Listing C.1: Gnuplot Boxplot-Skript

```
1 set style line 1 linetype 1 linecolor rgbcolor "#2f4f4f" linewidth  
2     1.5 pointtype 7 pointsize .5 pointinterval 1  
3 set style line 2 linetype 1 linecolor rgbcolor "#696969" linewidth  
4     1.5 pointtype 7 pointsize .5 pointinterval 1  
5 set style line 3 linetype 1 linecolor rgbcolor "#708090" linewidth  
6     1.5 pointtype 7 pointsize .5 pointinterval 1  
7 set style line 4 linetype 1 linecolor rgbcolor "#bebebe" linewidth  
8     1.5 pointtype 7 pointsize .5 pointinterval 1  
9  
10 if( i == 0 ) set terminal pngcairo truecolor transparent background "#  
11     ffffff" enhanced font "arial,10" size w, h  
12  
13 set style fill transparent solid 0.3 noborder  
14 #set style boxplot outliers pointtype 19  
15 #set style data boxplot  
16 #set key right bottom vertical Left noreverse enhanced box samplen .2  
17 #set key opaque  
18 set grid  
19  
20 #in this column we find the Data for sigma  
21 lastDataCol = 3+a+2  
22 #int this column we expect the vector  
23 vectorCol = 3+a+3  
24 inputfile = "data/single_.i_.dat"  
25 outMultiplot = "img/boxen/kondensiert/.i_.png"  
26  
27 set output outMultiplot  
28 set multiplot layout 1,3  
29 unset logscale  
30 set autoscale  
31 unset label  
32  
33 #  
34  
35 #setup the 1. plot  
36 set style data boxplot  
37 set xlabel "Optimization objective"  
38 set ylabel "Final Value - Coordinates ( x, y, z )"  
39  
40 set size 1, .6  
41 set origin .0,.4  
42 set autoscale  
43 set xrange [.5:a+.5]  
44 set xtics  
45  
46 set ytics format "%1.1f"  
47 set yrange [-4:4]  
48 if(a>3) set y2tics format "%1.1f"
```

```

43 | if(a>3) set y2label "Final Value - Wavenumbers ( n )"
44 |
45 | print "a=",a
46 | if(a==7) set xtics ("x" 1, "y" 2, "z" 3 ) scale 0.0
47 | if(a==3) plot inputfile u (1):5 ls 1 axes x1y1 notitle , \
48 |   '' u (2):6 ls 2 axes x1y1 notitle , \
49 |   '' u (3):7 ls 3 axes x1y1 notitle
50 | if(a==3) unset y2tics
51 | if(a==3) unset y2label
52 |
53 | if(a==6) plot inputfile u (1):5 ls 1 notitle , \
54 |   '' u (2):6 ls 2 notitle , \
55 |   '' u (3):7 ls 3 notitle , \
56 |   '' u (4):8 ls 4 notitle , \
57 |   '' u (5):9 ls 4 notitle , \
58 |   '' u (6):10 ls 4 notitle
59 |
60 | if(a==7) set xtics ("x" 1, "y" 2, "z" 3, "n0" 4,"n1" 5,"n2" 6,"n3" 7)
61 |   scale 0.0
62 | if(a==7) plot inputfile u ($2 < limit ? (1): 1/0):5 ls 1 axes x1y1
63 |   notitle , \
64 |   '' u ($2 < limit ? (2): 1/0):6 ls 2 axes x1y1 notitle , \
65 |   '' u ($2 < limit ? (3): 1/0):7 ls 3 axes x1y1 notitle , \
66 |   '' u ($2 < limit ? (4): 1/0):8 ls 4 axes x1y2 notitle , \
67 |   '' u ($2 < limit ? (5): 1/0):9 ls 4 axes x1y2 notitle , \
68 |   '' u ($2 < limit ? (6): 1/0):10 ls 4 axes x1y2 notitle , \
69 |   '' u ($2 < limit ? (7): 1/0):11 ls 4 axes x1y2 notitle
70 |
71 | if(a==8) set xtics ("x" 1, "y" 2, "z" 3, "n0" 4,"n1" 5,"n2" 6,"n3" 7,
72 |   "n4" 8) scale 0.0
73 | if(a==8) plot inputfile u ($2 < limit ? (1): 1/0):5 ls 1 axes x1y1
74 |   notitle , \
75 |   '' u ($2 < limit ? (2): 1/0):6 ls 2 axes x1y1 notitle , \
76 |   '' u ($2 < limit ? (3): 1/0):7 ls 3 axes x1y1 notitle , \
77 |   '' u ($2 < limit ? (4): 1/0):8 ls 4 axes x1y2 notitle , \
78 |   '' u ($2 < limit ? (5): 1/0):9 ls 4 axes x1y2 notitle , \
79 |   '' u ($2 < limit ? (6): 1/0):10 ls 4 axes x1y2 notitle , \
80 |   '' u ($2 < limit ? (7): 1/0):11 ls 4 axes x1y2 notitle , \
81 |   '' u ($2 < limit ? (8): 1/0):12 ls 4 axes x1y2 notitle
82 | if(a==9) set xtics ("x" 1, "y" 2, "z" 3, "n0" 4,"n1" 5,"n2" 6,"n3" 7,
83 |   "n4" 8, "N5" 9) scale 0.0
84 | if(a==9) plot inputfile u ($2 < limit ? (1): 1/0):5 ls 1 axes x1y1
85 |   notitle , \
86 |   '' u ($2 < limit ? (2): 1/0):6 ls 2 axes x1y1 notitle , \
87 |   '' u ($2 < limit ? (3): 1/0):7 ls 3 axes x1y1 notitle , \
88 |   '' u ($2 < limit ? (4): 1/0):8 ls 4 axes x1y2 notitle , \
89 |   '' u ($2 < limit ? (5): 1/0):9 ls 4 axes x1y2 notitle , \
90 |   '' u ($2 < limit ? (6): 1/0):10 ls 4 axes x1y2 notitle , \
91 |   '' u ($2 < limit ? (7): 1/0):11 ls 4 axes x1y2 notitle , \
92 |   '' u ($2 < limit ? (8): 1/0):12 ls 4 axes x1y2 notitle , \
93 |   '' u ($2 < limit ? (9): 1/0):12 ls 4 axes x1y2 notitle
94 | if(a==10) set xtics ("x" 1, "y" 2, "z" 3, "n0" 4,"n1" 5,"n2" 6,"n3" 7,
95 |   "n4" 8, "N5" 9, "N6" 10) scale 0.0
96 | if(a==10) plot inputfile u ($2 < limit ? (1): 1/0):5 ls 1 axes x1y1
97 |   notitle , \
98 |   '' u ($2 < limit ? (2): 1/0):6 ls 2 axes x1y1 notitle , \

```

```

99      '' u ($2 < limit ? (4): 1/0):8 ls 4 axes x1y2 notitle , \
100     '' u ($2 < limit ? (5): 1/0):9 ls 4 axes x1y2 notitle , \
101     '' u ($2 < limit ? (6): 1/0):10 ls 4 axes x1y2 notitle , \
102     '' u ($2 < limit ? (7): 1/0):11 ls 4 axes x1y2 notitle , \
103     '' u ($2 < limit ? (8): 1/0):12 ls 4 axes x1y2 notitle , \
104     '' u ($2 < limit ? (9): 1/0):12 ls 4 axes x1y2 notitle , \
105     '' u ($2 < limit ? (10): 1/0):12 ls 4 axes x1y2 notitle
106
107 if(a==11) set xtics ("x" 1, "y" 2, "z" 3, "n0" 4,"n1" 5,"n2" 6,"n3" 7,
108           "n4" 8, "N5" 9, "N6" 10, "N7" 11) scale 0.0
109
110 if(a==11) plot inputfile u ($2 < limit ? (1): 1/0):5 ls 1 axes x1y1
111   notitle , \
112   '' u ($2 < limit ? (2): 1/0):6 ls 2 axes x1y1 notitle , \
113   '' u ($2 < limit ? (3): 1/0):7 ls 3 axes x1y1 notitle , \
114   '' u ($2 < limit ? (4): 1/0):8 ls 4 axes x1y2 notitle , \
115   '' u ($2 < limit ? (5): 1/0):9 ls 4 axes x1y2 notitle , \
116   '' u ($2 < limit ? (6): 1/0):10 ls 4 axes x1y2 notitle , \
117   '' u ($2 < limit ? (7): 1/0):11 ls 4 axes x1y2 notitle , \
118   '' u ($2 < limit ? (8): 1/0):12 ls 4 axes x1y2 notitle , \
119   '' u ($2 < limit ? (9): 1/0):12 ls 4 axes x1y2 notitle , \
120   '' u ($2 < limit ? (10): 1/0):12 ls 4 axes x1y2 notitle
121 unset label
122 unset y2label
123 unset y2range
124 unset y2tics
125 #
126
127 #setup the 2. plot
128 set boxwidth 0.05 relative
129
130 set autoscale
131 set xlabel ""
132 #set logscale y
133 set ylabel "Evaluations"
134 set size .25, .4
135 set origin .0,.0
136 unset xtics
137 #set xrange [-.2:.4]
138 set ytics format "%.1e"
139
140 plot inputfile u ($2 < limit ? (1): 1/0):2 ls 4 notitle
141
142 #
143
144 #setup the 3. plot
145 #set logscale y
146 set ytics
147 set xlabel ""
148 set ylabel "Function Value"
149 set size .25, .4
150 set origin .25,.0
151 unset xtics
152 set ytics format "%.1e"
153
154 plot inputfile u ($2 < limit ? (1): 1/0):3 ls 4 notitle
155

```

```

156 #
157 #setup the 4. plot
158 set xlabel ""
159 set ylabel "Sigma"
160 set ytics format "%.1e"
161
162 set size .25, .4
163 set origin .50,.0
164
165 unset logscale
166 set autoscale
167 unset xtics
168 set ytics
169
170 plot inputfile u ($2 < limit ? (1): 1/0):lastDataCol ls 4 notitle
171
172 #

173 #setup the 5. plot
174 set xlabel ""
175 set ylabel "Distance"
176 set ytics format "%.2e"
177
178 set size .25, .4
179 set origin .75,.0
180
181 set autoscale
182 unset xtics
183 set ytics
184
185 plot inputfile u ($2 < limit ? (1): 1/0):vectorCol ls 4 notitle
186
187 #

188 i=i+1
189
190 unset multiplot
191 unset xtics
192
193 if ( i < m) reread
194 i=0

```

## C.2. Lineplot

Listing C.2: Gnuplot Lineplot-Skript

```

1  #prerequisites set i, n and the number of antennas to proper values
2  at(file , row, col) = system( sprintf("awk -v row=%d -v col=%d 'NR ==
   row {print $col}' %s", row, col, file) )
3
4  set style line 1 linetype 1 linecolor rgbcolor "#882f4f4f" linewidth
   .5
5  set style line 2 linetype 1 linecolor rgbcolor "#88696969" linewidth
   .5
6  set style line 3 linetype 1 linecolor rgbcolor "#88708090" linewidth
   .5
7  set style line 4 linetype 1 linecolor rgbcolor "#ccbebebe" linewidth
   .5
8
9  set style line 5 linetype 1 linecolor rgbcolor "#99696969" linewidth
   .3
10
11 if( i == 0 ) set terminal pngcairo truecolor transparent background "#"
   ffffff" enhanced font 'arial,10" size w, h
12
13 set key right bottom vertical Left noreverse enhanced box samplen .2
14 set key opaque
15 set grid
16
17 lastDataCol = 3+a+2
18 inputfile = "data/.i.".dat"
19 outMultiplot = "img/linien/kondensiert/.i.".png"
20
21 file=inputfile ; row=2 ; col=2
22
23 set output outMultiplot
24 set multiplot layout 1,3
25 unset logscale
26 set autoscale
27
28 #
_____
29 stats inputfile u 1 name "Stat" nooutput
30
31 #print "test ".at(file , Stat_records , 1)
32
33 locallimit=0.5*limit
34 if(a<=3)print "local limit is: ",locallimit
35
36 #setup the first plot
37 set xrange [0:locallimit]
38
39 set ytics format "%0.0f"
40 set yrange [-10:10]
41
42 if(a>3) set autoscale
43
44 set clip one
45 set xlabel "Funtion Evaluations"
46 set ylabel "Objective Values"
47 set xtics
48 set ytics
49 set size 1., .6
50 set origin .0 ,.4
51

```

```

52
53 #print "local locallimit ",locallimit
54
55 if( a==3 ) plot inputfile u ($1 < locallimit ? $1 : 1/0):4 w lines
56   title "x" ls 1, \
57   "" u ($1 < locallimit ? $1 : 1/0):5 w lines title "y" ls 2, \
58   "" u ($1 < locallimit ? $1 : 1/0):6 w lines title "z" ls 3
59
60 if( a==7 ) plot inputfile u ($1 < locallimit ? $1 : 1/0):4 w lines
61   title "x" ls 1, \
62   "" u ($1 < locallimit ? $1 : 1/0):5 w lines title "y" ls 2, \
63   "" u ($1 < locallimit ? $1 : 1/0):6 w lines title "z" ls 3, \
64   "" u ($1 < locallimit ? $1 : 1/0):7 w lines title "n0" ls 4, \
65   "" u ($1 < locallimit ? $1 : 1/0):8 w lines title "n1" ls 4, \
66   "" u ($1 < locallimit ? $1 : 1/0):9 w lines title "n2" ls 4, \
67   "" u ($1 < locallimit ? $1 : 1/0):10 w lines title "n3" ls 4, \
68   "" u ($1 < locallimit ? $1 : 1/0):11 w lines title "n4" ls 4, \
69   "" u ($1 < locallimit ? $1 : 1/0):4 w lines title "x" ls 1, \
70   "" u ($1 < locallimit ? $1 : 1/0):5 w lines title "y" ls 2, \
71   "" u ($1 < locallimit ? $1 : 1/0):6 w lines title "z" ls 3
72
73 if( a==8 ) plot inputfile u ($1 < locallimit ? $1 : 1/0):7 w lines
74   title "n0" ls 4, \
75   "" u ($1 < locallimit ? $1 : 1/0):8 w lines title "n1" ls 4, \
76   "" u ($1 < locallimit ? $1 : 1/0):9 w lines title "n2" ls 4, \
77   "" u ($1 < locallimit ? $1 : 1/0):10 w lines title "n3" ls 4, \
78   "" u ($1 < locallimit ? $1 : 1/0):11 w lines title "n4" ls 4, \
79   "" u ($1 < locallimit ? $1 : 1/0):12 w lines title "n5" ls 4, \
80   "" u ($1 < locallimit ? $1 : 1/0):4 w lines title "x" ls 1, \
81   "" u ($1 < locallimit ? $1 : 1/0):5 w lines title "y" ls 2, \
82   "" u ($1 < locallimit ? $1 : 1/0):6 w lines title "z" ls 3
83
84 if( a==9 ) plot inputfile u ($1 < locallimit ? $1 : 1/0):7 w lines
85   title "n0" ls 4, \
86   "" u ($1 < locallimit ? $1 : 1/0):8 w lines title "n1" ls 4, \
87   "" u ($1 < locallimit ? $1 : 1/0):9 w lines title "n2" ls 4, \
88   "" u ($1 < locallimit ? $1 : 1/0):10 w lines title "n3" ls 4, \
89   "" u ($1 < locallimit ? $1 : 1/0):11 w lines title "n4" ls 4, \
90   "" u ($1 < locallimit ? $1 : 1/0):12 w lines title "n5" ls 4, \
91   "" u ($1 < locallimit ? $1 : 1/0):13 w lines title "n6" ls 4, \
92   "" u ($1 < locallimit ? $1 : 1/0):4 w lines title "x" ls 1, \
93   "" u ($1 < locallimit ? $1 : 1/0):5 w lines title "y" ls 2, \
94   "" u ($1 < locallimit ? $1 : 1/0):6 w lines title "z" ls 3
95
96 if( a==10 ) plot inputfile u ($1 < locallimit ? $1 : 1/0):7 w lines
97   title "n0" ls 4, \
98   "" u ($1 < locallimit ? $1 : 1/0):8 w lines title "n1" ls 4, \
99   "" u ($1 < locallimit ? $1 : 1/0):9 w lines title "n2" ls 4, \
100  "" u ($1 < locallimit ? $1 : 1/0):10 w lines title "n3" ls 4, \
101  "" u ($1 < locallimit ? $1 : 1/0):11 w lines title "n4" ls 4, \
102  "" u ($1 < locallimit ? $1 : 1/0):12 w lines title "n5" ls 4, \
103  "" u ($1 < locallimit ? $1 : 1/0):13 w lines title "n6" ls 4, \
104  "" u ($1 < locallimit ? $1 : 1/0):14 w lines title "n7" ls 4, \
105  "" u ($1 < locallimit ? $1 : 1/0):4 w lines title "x" ls 1, \
106  "" u ($1 < locallimit ? $1 : 1/0):5 w lines title "y" ls 2, \
107  "" u ($1 < locallimit ? $1 : 1/0):6 w lines title "z" ls 3
108
109 set autoscale

```

```
110
111 set ytics format "%.1e"
112 set logscale y
113 set xlabel ""
114 set ylabel "Function Value"
115 set size .5, .4
116 set yrange [1e-25:20000]
117 set origin .0,.0
118 plot inputfile u ($1 < locallimit ? $1 : 1/0):2 w lines ls 5 title "
    fitness"
119
120 set size .5, .4
121 set origin .5,.0
122 set autoscale
123 #set yrange [1e-10:2]
124 #set xrange [0:3500]
125 #set clip one
126 set ylabel "Sigma"
127
128 plot inputfile u ($1 < locallimit ? $1 : 1/0):lastDataCol w lines ls
    5 title "{/Symbol s}"
129
130 i=i+1
131
132 unset multiplot
133
134 if ( i < m) reread
135 i=0
```

### C.3. Scatterplot

Listing C.3: Gnuplot Scatterplot-Skript

```

1  # This script is to generate a plot from the final values of the
2  # solutions for one antenna
3  #
4  #prerequisites set i, n and the number of antennas to proper values
5  #
6  at(file, row, col) = system( sprintf("awk -v row=%d -v col=%d 'NR ==
7  row {print $col}' %s", row, col, file) )
8  to(file, min, first, mean, third, max, row) = system( sprintf("echo %d
9  %e %e %e %e >> %s", row, min, first, mean, third, max, file) )
10 header(file) = system( sprintf("echo \"#Idx min first mean third max\""
11 >> %s", file) )
12 toScientific(file, min, first, mean, third, max, row) = system(
13   sprintf("echo %d %e %e %e %e >> %s", row, min, first, mean,
14   third, max, file) )
15 remove(file) = system( sprintf("rm %s", file) )
16 to2(file, value) = system( sprintf("echo 1 2 3 %s %s", value, file) )
17 echoStats(min, first, mean, third, max) = system( sprintf("echo %e %e
18 %e %e %e ", min, first, mean, third, max) )
19 unset style
20 set style line 1 linetype 1 linecolor rgb "#708090" linewidth 1
21   pointtype 7 pointsize .5
22 set style line 2 linetype -1 linecolor rgb "#2f4f4f" linewidth 1.2
23
24 set style line 3 linetype 1 linecolor rgb "#ee708090" linewidth 1.000
25   pointtype 7 pointsize .5 pointinterval 1
26 #set style line 3 linetype 1 linecolor rgb "red" linewidth 1.000
27   pointtype 7 pointsize 1 pointinterval 5
28 #set style line 4 linetype 1 linecolor rgb "gray" linewidth 1
29   pointtype 2 pointsize default pointinterval 0
30
31 set style arrow 1 heads size screen 0.008,90 ls 2
32
33 if( i == 0 ) set terminal pngcairo truecolor transparent background "#
34   ffffff" enhanced font "arial,10" size w, h
35
36 set style fill transparent solid 0.3 noborder
37 set key right bottom vertical Left noreverse enhanced box samplen .2
38 set key opaque
39 set grid
40
41 lastDataCol = 3+a+2
42 inputfile = "data/single_.i..dat"
43 input_all = "data/single_.i..dat"
44 input_one = "data/.i..dat"
45
46 outMultiplot = "img/linien/kondensiert/scatter".i..png"
47
48 #print "Processing: Start"
49
50 set output outMultiplot
51
52 set multiplot layout a,a
53
54 #collect information about the file
55 unset logscale
56 set autoscale
57 unset label
58 unset xlabel

```

```

49 | unset ylabel
50 |
51 | #
52 | #setup the 1. plot
53 | unset ytics
54 | unset xtics
55 |
56 | LABELX = sprintf("x" )
57 | LABELY = sprintf("y" )
58 | LABELZ = sprintf("z" )
59 | LABELN0 = sprintf("n0" )
60 | LABELN1 = sprintf("n1" )
61 | LABELN2 = sprintf("n2" )
62 | LABELN3 = sprintf("n3" )
63 | LABELN4 = sprintf("n4" )
64 | LABELN5 = sprintf("n5" )
65 | LABELN6 = sprintf("n6" )
66 | LABELN7 = sprintf("n7" )
67 |
68 | unset key
69 |
70 | xlabelpos = .1
71 | labelypos = .5
72 |
73 | # generate first row
74 | set label at graph xlabelpos,labelypos center LABELX front left font "
75 |   Arial,24" textcolor rgb "#4f2f2f"
76 | plot inputfile u ($2 < limit ? $5: 1/0):5 ls 3 notitle
77 | unset label
78 |
79 | plot inputfile u ($2 < limit ? $5: 1/0):6 ls 1 notitle
80 | plot inputfile u ($2 < limit ? $5: 1/0):7 ls 1 notitle
81 | if(a>=4)plot inputfile u ($2 < limit ? $5: 1/0):8 ls 1 notitle
82 | if(a>=5)plot inputfile u ($2 < limit ? $5: 1/0):9 ls 1 notitle
83 | if(a>=6)plot inputfile u ($2 < limit ? $5: 1/0):10 ls 1 notitle
84 | if(a>=7)plot inputfile u ($2 < limit ? $5: 1/0):11 ls 1 notitle
85 | if(a>=8)plot inputfile u ($2 < limit ? $5: 1/0):12 ls 1 notitle
86 | if(a>=9)plot inputfile u ($2 < limit ? $5: 1/0):13 ls 1 notitle
87 | if(a>=10)plot inputfile u ($2 < limit ? $5: 1/0):14 ls 1 notitle
88 | if(a>=11)plot inputfile u ($2 < limit ? $5: 1/0):15 ls 1 notitle
89 |
90 | # 2.
91 | plot inputfile u ($2 < limit ? $6: 1/0):5 ls 1 notitle
92 |
93 | set label at graph xlabelpos,labelypos center LABELY front left font "
94 |   Arial,24" textcolor rgb "#4f2f2f"
95 | plot inputfile u ($2 < limit ? $6: 1/0):6 ls 3 notitle
96 | unset label
97 |
98 | plot inputfile u ($2 < limit ? $6: 1/0):7 ls 1 notitle
99 | if(a>=4)plot inputfile u ($2 < limit ? $6: 1/0):8 ls 1 notitle
100 | if(a>=5)plot inputfile u ($2 < limit ? $6: 1/0):9 ls 1 notitle
101 | if(a>=6)plot inputfile u ($2 < limit ? $6: 1/0):10 ls 1 notitle
102 | if(a>=7)plot inputfile u ($2 < limit ? $6: 1/0):11 ls 1 notitle
103 | if(a>=8)plot inputfile u ($2 < limit ? $6: 1/0):12 ls 1 notitle
104 | if(a>=9)plot inputfile u ($2 < limit ? $6: 1/0):13 ls 1 notitle
105 |
106 | # 3.
107 | plot inputfile u ($2 < limit ? $7: 1/0):5 ls 1 notitle
108 | plot inputfile u ($2 < limit ? $7: 1/0):6 ls 1 notitle

```

```

109
110 set label at graph labelxpos ,labelypos center LABELZ front left font "
111   Arial ,24" textcolor rgb "#4f2f2f"
112 plot inputfile u ($2 < limit ? $7: 1/0):7 ls 3 notitle
113 unset label
114
115 if(a>=4)plot inputfile u ($2 < limit ? $7: 1/0):8 ls 1 notitle
116 if(a>=5)plot inputfile u ($2 < limit ? $7: 1/0):9 ls 1 notitle
117 if(a>=6)plot inputfile u ($2 < limit ? $7: 1/0):10 ls 1 notitle
118 if(a>=7)plot inputfile u ($2 < limit ? $7: 1/0):11 ls 1 notitle
119 if(a>=8)plot inputfile u ($2 < limit ? $7: 1/0):12 ls 1 notitle
120 if(a>=9)plot inputfile u ($2 < limit ? $7: 1/0):13 ls 1 notitle
121 if(a>=10)plot inputfile u ($2 < limit ? $7: 1/0):14 ls 1 notitle
122 if(a>=11)plot inputfile u ($2 < limit ? $7: 1/0):15 ls 1 notitle
123
124 # 4.
125 if(a>3)plot inputfile u ($2 < limit ? $8: 1/0):5 ls 1 notitle
126 if(a>3)plot inputfile u ($2 < limit ? $8: 1/0):6 ls 1 notitle
127 if(a>3)plot inputfile u ($2 < limit ? $8: 1/0):7 ls 1 notitle
128
129 set label at graph labelxpos ,labelypos center LABELN0 front left font
130   "Arial ,24" textcolor rgb "#4f2f2f"
131 if(a>=4)plot inputfile u ($2 < limit ? $8: 1/0):8 ls 3 notitle
132 unset label
133
134 if(a>=5)plot inputfile u ($2 < limit ? $8: 1/0):9 ls 1 notitle
135 if(a>=6)plot inputfile u ($2 < limit ? $8: 1/0):10 ls 1 notitle
136 if(a>=7)plot inputfile u ($2 < limit ? $8: 1/0):11 ls 1 notitle
137 if(a>=8)plot inputfile u ($2 < limit ? $8: 1/0):12 ls 1 notitle
138 if(a>=9)plot inputfile u ($2 < limit ? $8: 1/0):13 ls 1 notitle
139 if(a>=10)plot inputfile u ($2 < limit ? $8: 1/0):14 ls 1 notitle
140 if(a>=11)plot inputfile u ($2 < limit ? $8: 1/0):15 ls 1 notitle
141
142 # 5.
143 if(a>3)plot inputfile u ($2 < limit ? $9: 1/0):5 ls 1 notitle
144 if(a>3)plot inputfile u ($2 < limit ? $9: 1/0):6 ls 1 notitle
145 if(a>3)plot inputfile u ($2 < limit ? $9: 1/0):7 ls 1 notitle
146 if(a>4)plot inputfile u ($2 < limit ? $9: 1/0):8 ls 1 notitle
147
148 set label at graph labelxpos ,labelypos center LABELN1 front left font
149   "Arial ,24" textcolor rgb "#4f2f2f"
150 if(a>=5)plot inputfile u ($2 < limit ? $9: 1/0):9 ls 3 notitle
151 unset label
152
153 if(a>=6)plot inputfile u ($2 < limit ? $9: 1/0):10 ls 1 notitle
154 if(a>=7)plot inputfile u ($2 < limit ? $9: 1/0):11 ls 1 notitle
155 if(a>=8)plot inputfile u ($2 < limit ? $9: 1/0):12 ls 1 notitle
156 if(a>=9)plot inputfile u ($2 < limit ? $9: 1/0):13 ls 1 notitle
157 if(a>=10)plot inputfile u ($2 < limit ? $9: 1/0):14 ls 1 notitle
158 if(a>=11)plot inputfile u ($2 < limit ? $9: 1/0):15 ls 1 notitle
159
160 # 6.
161 if(a>3)plot inputfile u ($2 < limit ? $10: 1/0):5 ls 1 notitle
162 if(a>3)plot inputfile u ($2 < limit ? $10: 1/0):6 ls 1 notitle
163 if(a>3)plot inputfile u ($2 < limit ? $10: 1/0):7 ls 1 notitle
164 if(a>4)plot inputfile u ($2 < limit ? $10: 1/0):8 ls 1 notitle
165 if(a>5)plot inputfile u ($2 < limit ? $10: 1/0):9 ls 1 notitle
166
167 set label at graph labelxpos ,labelypos center LABELN2 front left font
168   "Arial ,24" textcolor rgb "#4f2f2f"
169 if(a>=6)plot inputfile u ($2 < limit ? $10: 1/0):10 ls 3 notitle
170 unset label
171
172 if(a>=7)plot inputfile u ($2 < limit ? $10: 1/0):11 ls 1 notitle

```

```

169 | if(a>=8)plot inputfile u ($2 < limit ? $10: 1/0):12 ls 1 notitle
170 | if(a>=9)plot inputfile u ($2 < limit ? $10: 1/0):13 ls 1 notitle
171 | if(a>=10)plot inputfile u ($2 < limit ? $10: 1/0):14 ls 1 notitle
172 | if(a>=11)plot inputfile u ($2 < limit ? $10: 1/0):15 ls 1 notitle
173 |
174 | # 7.
175 | if(a>3)plot inputfile u ($2 < limit ? $11: 1/0):5 ls 1 notitle
176 | if(a>3)plot inputfile u ($2 < limit ? $11: 1/0):6 ls 1 notitle
177 | if(a>3)plot inputfile u ($2 < limit ? $11: 1/0):7 ls 1 notitle
178 | if(a>=4)plot inputfile u ($2 < limit ? $11: 1/0):8 ls 1 notitle
179 | if(a>=5)plot inputfile u ($2 < limit ? $11: 1/0):9 ls 1 notitle
180 | if(a>=6)plot inputfile u ($2 < limit ? $11: 1/0):10 ls 1 notitle
181 | set label at graph labelxpos ,labelypos center LABELN3 front left font
|   "Arial,24" textcolor rgb "#4f2f2f"
182 | if(a>=7)plot inputfile u ($2 < limit ? $11: 1/0):11 ls 3 notitle
183 | unset label
184 |
185 | if(a>=8)plot inputfile u ($2 < limit ? $11: 1/0):12 ls 1 notitle
186 | if(a>=9)plot inputfile u ($2 < limit ? $11: 1/0):13 ls 1 notitle
187 | if(a>=10)plot inputfile u ($2 < limit ? $11: 1/0):14 ls 1 notitle
188 | if(a>=11)plot inputfile u ($2 < limit ? $11: 1/0):15 ls 1 notitle
189 |
190 | # 8.
191 | if(a>=8)      plot inputfile u ($2 < limit ? $12: 1/0):5 ls 1 notitle
192 | if(a>=8)      plot inputfile u ($2 < limit ? $12: 1/0):6 ls 1 notitle
193 | if(a>=8)      plot inputfile u ($2 < limit ? $12: 1/0):7 ls 1 notitle
194 | if(a>=8)      plot inputfile u ($2 < limit ? $12: 1/0):8 ls 1 notitle
195 | if(a>=8)      plot inputfile u ($2 < limit ? $12: 1/0):9 ls 1 notitle
196 | if(a>=8)      plot inputfile u ($2 < limit ? $12: 1/0):10 ls 1 notitle
197 | if(a>=8)      plot inputfile u ($2 < limit ? $12: 1/0):11 ls 1 notitle
198 |
199 | if(a>=8)      set label at graph labelxpos ,labelypos center LABELN4
|   front left font "Arial,24" textcolor rgb "#4f2f2f"
200 | if(a>=8)      plot inputfile u ($2 < limit ? $12: 1/0):12 ls 3 notitle
201 | if(a>=8)      unset label
202 |
203 | if(a>=9)      plot inputfile u ($2 < limit ? $12: 1/0):13 ls 1 notitle
204 | if(a>=10)     plot inputfile u ($2 < limit ? $12: 1/0):14 ls 1 notitle
205 | if(a>=11)     plot inputfile u ($2 < limit ? $12: 1/0):15 ls 1 notitle
206 |
207 | # 9.
208 | if(a>=9)      plot inputfile u ($2 < limit ? $13: 1/0):5 ls 1 notitle
209 | if(a>=9)      plot inputfile u ($2 < limit ? $13: 1/0):6 ls 1 notitle
210 | if(a>=9)      plot inputfile u ($2 < limit ? $13: 1/0):7 ls 1 notitle
211 | if(a>=9)      plot inputfile u ($2 < limit ? $13: 1/0):8 ls 1 notitle
212 | if(a>=9)      plot inputfile u ($2 < limit ? $13: 1/0):9 ls 1 notitle
213 | if(a>=9)      plot inputfile u ($2 < limit ? $13: 1/0):10 ls 1 notitle
214 | if(a>=9)      plot inputfile u ($2 < limit ? $13: 1/0):11 ls 1 notitle
215 | if(a>=9)      plot inputfile u ($2 < limit ? $13: 1/0):12 ls 1 notitle
216 |
217 | if(a>=9)      set label at graph labelxpos ,labelypos center LABELN5
|   front left font "Arial,24" textcolor rgb "#4f2f2f"
218 | if(a>=9)      plot inputfile u ($2 < limit ? $13: 1/0):13 ls 3 notitle
219 | if(a>=9)      unset label
220 |
221 | if(a>=10)     plot inputfile u ($2 < limit ? $13: 1/0):14 ls 1 notitle
222 | if(a>=11)     plot inputfile u ($2 < limit ? $13: 1/0):15 ls 1 notitle
223 |
224 | # 10.
225 | if(a>=10)     plot inputfile u ($2 < limit ? $14: 1/0):5 ls 1 notitle
226 | if(a>=10)     plot inputfile u ($2 < limit ? $14: 1/0):6 ls 1 notitle
227 | if(a>=10)     plot inputfile u ($2 < limit ? $14: 1/0):7 ls 1 notitle
228 | if(a>=10)     plot inputfile u ($2 < limit ? $14: 1/0):8 ls 1 notitle
229 | if(a>=10)     plot inputfile u ($2 < limit ? $14: 1/0):9 ls 1 notitle

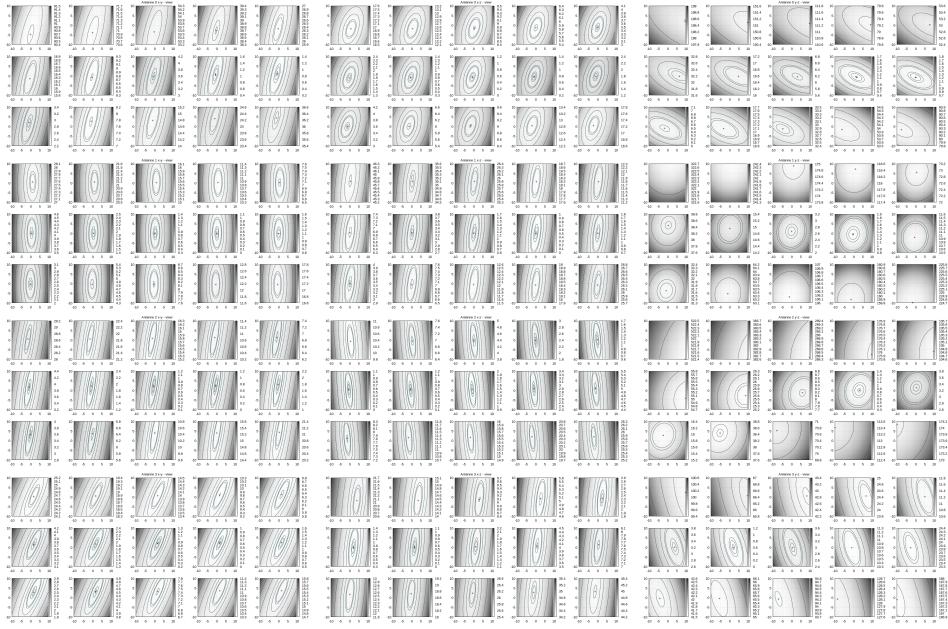
```

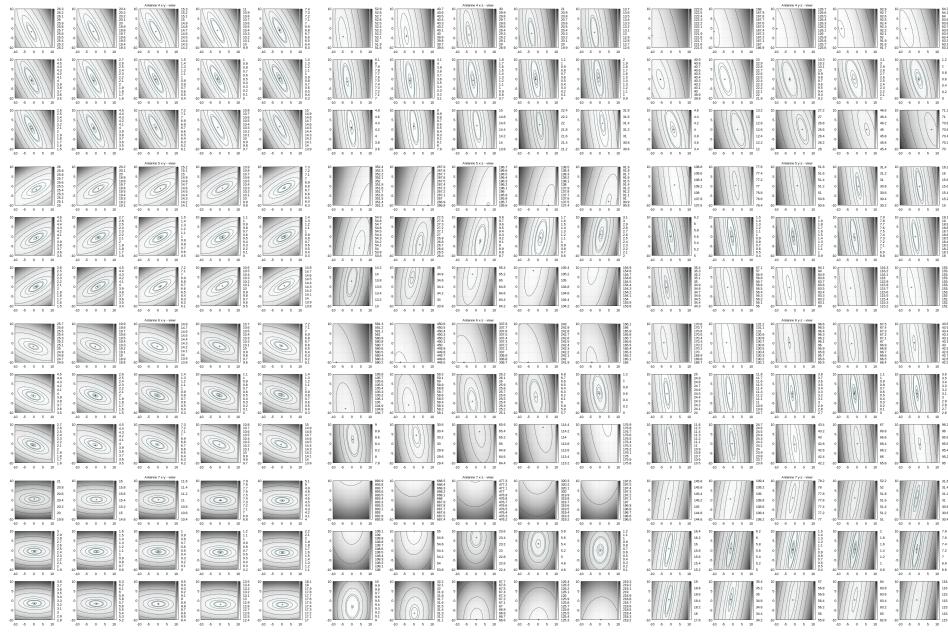
```

230 | if(a>=10)      plot inputfile u ($2 < limit ? $14: 1/0):10 ls 1 notitle
231 | if(a>=10)      plot inputfile u ($2 < limit ? $14: 1/0):11 ls 1 notitle
232 | if(a>=10)      plot inputfile u ($2 < limit ? $14: 1/0):12 ls 1 notitle
233 | if(a>=10)      plot inputfile u ($2 < limit ? $14: 1/0):13 ls 1 notitle
234 |
235 | if(a>=10)      set label at graph labelxpos ,label ypos center LABELN6
236 |         front left font "Arial,24" textColor rgb "#4f2f2f"
237 | if(a>=10)      plot inputfile u ($2 < limit ? $14: 1/0):14 ls 3 notitle
238 | if(a>=10)      unset label
239 |
240 | if(a>=11)      plot inputfile u ($2 < limit ? $14: 1/0):15 ls 1 notitle
241 |
242 | # Generate last row
243 | if(a>=11)      plot inputfile u ($2 < limit ? $15: 1/0):5 ls 1 notitle
244 | if(a>=11)      plot inputfile u ($2 < limit ? $15: 1/0):6 ls 1 notitle
245 | if(a>=11)      plot inputfile u ($2 < limit ? $15: 1/0):7 ls 1 notitle
246 | if(a>=11)      plot inputfile u ($2 < limit ? $15: 1/0):8 ls 1 notitle
247 | if(a>=11)      plot inputfile u ($2 < limit ? $15: 1/0):9 ls 1 notitle
248 | if(a>=11)      plot inputfile u ($2 < limit ? $15: 1/0):10 ls 1 notitle
249 | if(a>=11)      plot inputfile u ($2 < limit ? $15: 1/0):11 ls 1 notitle
250 | if(a>=11)      plot inputfile u ($2 < limit ? $15: 1/0):12 ls 1 notitle
251 | if(a>=10)      plot inputfile u ($2 < limit ? $15: 1/0):13 ls 1 notitle
252 | if(a>=10)      plot inputfile u ($2 < limit ? $15: 1/0):14 ls 1 notitle
253 |
254 | if(a>=11)      set label at graph labelxpos ,label ypos center LABELN7
255 |         front left font "Arial,24" textColor rgb "#4f2f2f"
256 | if(a>=11)      plot inputfile u ($2 < limit ? $15: 1/0):15 ls 3 notitle
257 |
258 | i=i+1
259 |
260 | unset multiplot
261 |
262 | if ( i < m) reread
263 | i=0
264 |#print "rm ".ObjectiveOut.remove( ObjectiveOut )
265 |#print "rm ".SigmaOut.remove( SigmaOut )
266 |#print "rm ".FitnessOut.remove( FitnessOut )
267 |#print "rm ".EvalOut.remove( EvalOut )

```

## D. Fitness Plots







# Literaturverzeichnis

- [1] Evolution strategies – A comprehensive introduction. In: *Natural Computing* 1 (2002), Nr. 1, 3-52. <http://dx.doi.org/10.1023/A:1015059928466>. – DOI 10.1023/A:1015059928466. – ISSN 1567–7818
- [2] BOMZE, I. ; GROSSMANN, W.: *Optimierung - Theorie und Algorihtmen*. BI Wissensch.Vlg, 1993
- [3] BORGWERTH, Bernd ; GNIP, Christoph: Abschätzung der Wellenzahl durch Korrelation mit Kalibierpunkten. (2012)
- [4] BRONŠTEJN, I.N. ; SEMENDJAJEV, K.A. ; MUSIOL, G. ; MÜHLIG, H.: *Taschenbuch der Mathematik*. Deutsch Harri GmbH, 2012 <http://books.google.de/books?id=uPKPMAEACAAJ>. – ISBN 9783817120185
- [5] DESERNO, Thomas M. (Hrsg.) ; HANDELS, Heinz (Hrsg.) ; MEINZER, Hans-Peter (Hrsg.) ; TOLXDORFF, Thomas (Hrsg.): *Bildverarbeitung für die Medizin 2010 - Algorithmen - Systeme - Anwendungen, Aachen, Germany, March 14-16, 2010*. Bd. 574. CEUR-WS.org, 2010 (CEUR Workshop Proceedings)
- [6] *Electromagnetic compatibility and Radio spectrum Matters (ERM); Radio Frequency Identification Equipment operating in the band 865 MHz to 868 MHz with power levels up to 2 W*. 2010
- [7] FINKENZELLER, K.: *RFID-Handbuch: Grundlagen und praktische Anwendungen von Transpondern, kontaktlosen Chipkarten und NFC*. Hanser, 2008 <http://books.google.de/books?id=49HTBDrfqFUC>. – ISBN 9783446412002
- [8] GITHUB.COM: *GitHub — Repository Host*. <https://github.com>, 2013. – [Online, zuletzt geprüft am 30.4.2013]
- [9] HANSEN, N.: Benchmarking a BI-Population CMA-ES on the BBOB-2009 Function Testbed. In: *Workshop Proceedings of the GECCO Genetic and Evolutionary Computation Conference*, ACM, July 2009, S. 2389–2395
- [10] HANSEN, N. ; KERN, S.: Evaluating the CMA Evolution Strategy on Multimodal Test Functions. In: YAO, X. (Hrsg.) u. a.: *Parallel Problem Solving from Nature PPSN VIII* Bd. 3242, Springer, 2004 (LNCS), S. 282–291
- [11] HANSEN, Nikolaus: *Evolution Strategies and CMA-ES (Covariance Matrix Adaptation)*. <https://www.lri.fr/~hansen/gecco2013-CMA-ES-tutorial.pdf>. <https://www.lri.fr/~hansen/gecco2013-CMA-ES-tutorial.pdf>. – [Online, zuletzt geprüft am 30.7.2013]

- [12] HANSEN, Nikolaus: *Performance Evaluation of Anytime Blackbox Optimizers.* <https://www.lri.fr/~hansen/summer-school-performance-slides-final.pdf>. <https://www.lri.fr/~hansen/summer-school-performance-slides-final.pdf>. – [Online, zuletzt geprüft am 27.7.2013]
- [13] HANSEN, Nikolaus: *CMA-ES Anwendungen.* <https://www.lri.fr/~hansen/cmaapplications.pdf>, 2009. – [Online; zuletzt geprüft am 7-Sep-2013]
- [14] In: HANSEN, Nikolaus: *The CMA Evolution Strategy.* 2011
- [15] HANSEN, Nikolaus: *CMA-ES Source Code .* [https://www.lri.fr/~hansen/cmaes\\_inmatlab.html](https://www.lri.fr/~hansen/cmaes_inmatlab.html), 2013. – [Online; zuletzt geprüft am 6-Sep-2013]
- [16] HANSEN, Nikolaus ; ROTH, Stefan:
- [17] HEESCH, Dimitri van: *Doxxygen — Sourcecode documentation System.* <http://www.stack.nl/~dimitri/doxygen/>. <http://www.stack.nl/~dimitri/doxygen/>. – [Online, zuletzt geprüft am 7.5.2013]
- [18] HERMANN, M.: *Numerische Mathematik.* Oldenbourg Wissensch.Vlg, 2001 <http://books.google.de/books?id=6BvvAAAAMAAJ>. – ISBN 9783486255584
- [19] HERMANN, M.: *Numerische Mathematik.* Oldenbourg Wissensch.Vlg, 2001 <http://books.google.de/books?id=l45jSrRdL7AC>. – ISBN 9783486579352
- [20] HTTP://WWW.FOEBUD.ORG/RFIDM: *Digitalcourage.* <http://www.foebud.org/rfid>, 2013. – [Online, zuletzt geprüft am 22.8.2013]
- [21] IGEL, Christian ; HEIDRICH-MEISNER, Verena ; GLASMACHERS, Tobias: *Shark.* In: *Journal of Machine Learning Research* 9 (2008), 993–996. [http://image.diku.dk/shark/sphinx\\_pages/build/html/index.html](http://image.diku.dk/shark/sphinx_pages/build/html/index.html)
- [22] IGEL, Christian ; HEIDRICH-MEISNER, Verena ; GLASMACHERS, Tobias: *Shark.* In: *Journal of Machine Learning Research* 9 (2008), S. 993–996
- [23] KNIPSCHEER, Marius: *Planung eines Entwicklungstools für ein drahtloses 3D-Positionsmesssystem zur Lokalisierung von minimal invasiven chirurgischen Instrumenten.* FH-Gelsenkirchen. 2008
- [24] KOST, Bernd: *Optimierung mit Evolutionsstrategien.* Deutsch Harri GmbH, 2003 <http://books.google.de/books?id=FcgNJIg4lcAC>. – ISBN 9783817116993
- [25] KREUTZ, Martin ; SENDHOFF, Bernhard ; IGEL, Christian: *EALib: A C++ class library for evolutionary algorithms.* 2008. – Erstellt aus Quellcode
- [26] MUZALEWSKI, Mathäus: *Einsatz von Lernverfahren zur Interpolation von Positionsdaten eines RFID-basierten Navigationssystems.* 2011

- [27] OTTO GMBH & Co KG (Hrsg.): *Grundsätze*. [http://www.otto.com/umwelt/umwelt\\_grundindex.html](http://www.otto.com/umwelt/umwelt_grundindex.html), Abruf: 5. Okt. 2004. – Einstiegsseite zum Unternehmensleitbild
- [28] PRESS, W.H.: *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, 2007 <http://books.google.de/books?id=1aA0dzK3FegC>. – ISBN 9780521880688
- [29] RECHENBERG, I.: *Cybernetic Solution Path of an Experimental Problem*. Royal Aircraft Establishment, Library Translation 1122, 1965
- [30] RECHENBERG, I.: *Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, Stuttgart, 1973
- [31] RECHENBERG, I. Werkstatt Bionik und E.: *Evolutionsstrategie '94*. Frommann-Holzboog, Stuttgart, 1994
- [32] RFIDJOURNAL.COM: *RFID-Journal*. <http://www.rfidjournal.com>, 2013. – [Online, zuletzt geprüft am 22.8.2013]
- [33] RITSCHEL, Kai ; DEKOMIEN, Claudia ; WINTER, Susanne: Modellfunktion zur Approximation von Ultraschallkontrastmittelkonzentration zur semi-quantitativen Gewebeperfusionsbestimmung. In: *Bildverarbeitung für die Medizin*, 2012, S. 159–164
- [34] SCHWEFEL, H.-P.: *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*. Birkhäuser .Vlg, 1977
- [35] SCHWEFEL, H.-P.: *Evolution and Optimum seeking*. John Wiley and Son, New York, 1995
- [36] SIMO SÄRKÄÄ ; JAAKKOLA, Kaarle ; HUUSKO, Ville V. Viikari M.: Phase-Based UHF RFID Tracking With Nonlinear Kalman Filtering and Smoothing. (2012), February
- [37] SPELLUCCI, M.: *Numerische Verfahren der nichtlinearen Optimierung*. Birkhäuser .Vlg, 1993
- [38] WIKIPEDIA: *CMA-ES Concept of direct optimization — Wikipedia, The Free Encyclopedia*. [http://en.wikipedia.org/w/index.php?title=File:Concept\\_of\\_directional\\_optimization\\_in\\_CMA-ES\\_algorithm.png&oldid=532567533](http://en.wikipedia.org/w/index.php?title=File:Concept_of_directional_optimization_in_CMA-ES_algorithm.png&oldid=532567533). [http://en.wikipedia.org/w/index.php?title=File:Concept\\_of\\_directional\\_optimization\\_in\\_CMA-ES\\_algorithm.png&oldid=532567533](http://en.wikipedia.org/w/index.php?title=File:Concept_of_directional_optimization_in_CMA-ES_algorithm.png&oldid=532567533). Version: 2013. – [Online; zuletzt geprüft am 6-Sep-2013]
- [39] WIKIPEDIA: *Kalman-Filter — Wikipedia, The Free Encyclopedia*. <http://de.wikipedia.org/w/index.php?title=Kalman-Filter&oldid=116893284>. <http://de.wikipedia.org/w/index.php?title=Kalman-Filter&oldid=116893284>. Version: 2013. – [Online; zuletzt editiert am 4-April-2013]

- [40] WIKIPEDIA: *Konzept des CMA-ES* — Wikipedia, The Free Encyclopedia. [http://en.wikipedia.org/w/index.php?title=File:Concept\\_of\\_directional\\_optimization\\_in\\_CMA-ES\\_algorithm.png&oldid=532567533](http://en.wikipedia.org/w/index.php?title=File:Concept_of_directional_optimization_in_CMA-ES_algorithm.png&oldid=532567533). Version: 2013. – [Online; zuletzt geprüft am 2-Sep-2013]
- [41] WILLE, Andreas ; WINTER, Susanne: Medical Navigation Based on RFID Tag Signals: Model and Simulation. 55 (2010). <http://dx.doi.org/10.1515>. – DOI 10.1515
- [42] WINTER, Susanne: Ansätze zur Kalibrierung der Wellenzahl im RFID-Trackingsystem der Firma amedo. (2013)
- [43] WINTER, Susanne ; RITSCHEL, Kai ; BROLL, Magdalena ; DEKOMIEN, Claudia: Kalibrierung eines 3D-Ultraschallsystems mit evolutionärer Optimierung. In: [5], S. 187–190
- [44] ZURMÜHL, R. ; FALK, S.: *Matrizen und ihre Anwendungen für angewandte Mathematiker, Physiker und Ingenieure: Teil 2: Numerische Methoden*. Springer, 1986 (Matrizen und ihre Anwendungen / Rudolf Zurmühl, Sigurd Falk). <http://books.google.de/books?id=jN75e772xIQC>. – ISBN 9783540154747