RHEINISCHE FACHHOCHSCHULE KÖLN
Fachbereich für Ingenieurwesen

Optimierung der Auslegungsparameter eines permanentmagneterregten Synchronmotors am Beispiel eines FE-Entwicklungs-Tools

Kander Akinci 111142025

Projektarbeit

Prüfer: Prof. Dr. rer. nat. Lampe, Jörg
Studiengang: Elektrotechnik (B.Eng.)

Sommersemester 2020

Kurzfassung

Die folgende Arbeit befasst sich mit der Optimierung der Auslegungsparameter von numerischen Analysen komplexer Systeme. Hierbei ist das Kernthema, die Defizite der klassischen Optimierungsverfahren in der Handhabung rechenaufwendiger Systeme zu umgehen.

Das dafür angewandte Verfahren ist die Response Surface Methode (RSM). Mit Hilfe der RSM wird eine Approximation, auch Metamodell genannt, erstellt. Dieses Metamodell ersetzt die rechenaufwendige Systemanalyse und wird für die Optimierung der Auslegungsparameter verwendet.

Als Simulationsmodell wird ein Finite-Elemente-Programm (“FEMAG DC“) zur Berechnung zweidimensionaler und achsensymmetrischer Magnet- und Wirbelstromfelder verwendet. Die Implementierung der Simulation in das RSM Programm wird mit der Programmiersprache Python realisiert. Es werden die Approximation und die anschließende Optimierung mit den gängigen Bewertungskriterien analysiert.

Diese Arbeit ist ein Bespiel dafür, wie die RSM umgesetzt wird und kann helfen eigene Projekte in diesem Bereich zu realisieren.

Inhaltsverzeichnis

1 Einleitung

In allen Bereichen der menschlichen Gesellschaft werden permanent Entscheidungen getroffen. Ob sie nun persönlicher, staatlicher oder gemeinschaftlicher Natur sind. Die menschliche Entscheidungsfähigkeit kann aus philosophischer Sicht als Ausdruck des freien Willens interpretiert werden. Die Tatsache, dass im technischen Bereich wichtige Entscheidungen von Computern getroffen werden, relativiert diesen Gedanken.

Um eine Entscheidung treffen zu können, ist ein Entscheidungsspielraum notwendig. Dieser ist in der Regel nicht unbegrenzt, weshalb Entscheidungsrestriktionen beachtet werden müssen. Wenn eine Entscheidung getroffen wird, verändert sich die Entscheidungsumgebung, was den Vergleich von verschiedenen Entscheidungen nicht mehr möglich macht. Um Entscheidungen vergleichen zu können, muss eine spezifische Zielsetzung formuliert werden. Die optimale Entscheidungsfindung kann folgender-maßen formuliert werden:

Unter Einbezug der Entscheidungsauswirkungen und Restriktionen ist die Entscheidung zu bestimmen, welche die Zielsetzung am besten erfüllt.

Die Optimierungstheorie bietet eine Reihe an Hilfsmitteln, die eine systematische Entscheidungsfindung ermöglichen. Hierfür ist es notwendig, dass Restriktionen, Auswirkungen und Ziele in mathematischer Form ausgedrückt werden können. [Markos, Marion, & Martin, 2015]

1.1 Problemstellung

Moderne Simulationsprogramme ermöglichen heutzutage sehr genaue Analysen komplexer Systeme. Detaillierten Analysen können in einer Optimierungsumgebung jedoch zu Problemen führen. Durch die Größe der Probleme erfordern einzelne Systemanalysen einen hohen Berechnungsaufwand. Bei multidisziplinären Optimierungen haben Ziel- und Restriktionsfunktionen meistens einen nichtlinearen Verlauf, was ein häufiges Lösen der Systemgleichung erforderlich macht. Die Optimierungsalgorithmen kennen zudem nur das Systemverhalten in einem Designpunkt, was bedeutet, dass sie nicht sagen können, wie sich das System global verhält. Ohne diese globale Sicht, ist es nicht

möglich, die Qualität der Optimierung zu bewerten. Aufgrund der Nichtlinearität sind Optimierungsaufgaben häufig nicht konvex. Es können also mehrere lokale Lösungen existieren, wobei nur eine das globale Optimum repräsentiert. Um neben dem lokalen, auch das globale Optimum zu finden, muss ein erheblicher Aufwand betrieben werden, welcher in der Regel nicht mehr vertretbar ist.

Die Forderungen nach immer effizienteren Erzeugnissen in kürzeren Entwicklungszyklen, steht diesem Umstand entgegen. Das Ziel bei der Anwendung von Verfahren zur multidisziplinären Optimierung muss also sein, die Zahl der Systemanalysen auf ein Minimum zu reduzieren.

Eine Möglichkeit sind globale Approximationen, wie die Response Surface Methode (RSM). Diese erzeugt schnell auswertbare Approximationen, sogenannte Response Surface Approximationen (RSA), die an Stelle der aufwendigen Systemanalysen verwendet werden. Ziel ist dabei nicht, dass exakte Ergebnis zu bestimmen, sondern mit möglichst geringem Berechnungsaufwand, so nah wie möglich an das exakte Ergebnis heranzukommen.

Diese RSA müssen alle für eine Optimierung wichtigen systeminhärenten Eigenschaften wiedergeben und dürfen dabei nur so wenig Systemanalysen wie möglich benötigen. Die vorliegende Arbeit untersucht anhand einer konkreten Anwendung, wie Approximationen und deren Optimierungen ausfallen. [Gleichmar, 2004]

1.2 Zielsetzung und Aufbau der Arbeit

Ziel ist es, das Ergebnis der Response Surface Methode und die anschließende Optimierung an einem komplexen Simulationsmodell zu untersuchen. Das Simulationsmodell ist ein Permanentmagnet erregter Synchronmotor, dieser mit dem Finite Elemente Programm “FEMAG-DC“ erstellt wurde. Folgende Schritte werden unternommen, um das Ergebnis auszuarbeiten:

- Als erstes werden die Grundlagen der Response Surface Methode beschrieben.
- Anschließend wird das Simulationsmodell untersucht, um aus den Auslegungsparametern der Maschine die geeigneten Variablen auszuwählen. Für diese Variablen werden anschließend Grenzwerte festgelegt, um den Designraum einzugrenzen.

- Zunächst muss für die RSM eine Designmatrix erzeugt werden, die nach den Methoden der DOE erzeugt wird.
- Die grobe Parallelisierung der Systemanalysen wird mit Python realisiert und kann die Auslastung der Rechenkapazität verbessern. Ein Großteil der Simulationen wird parallel berechnet. So kann die Zeit, die benötigt wird, um ein komplexes System zu optimieren, weiter verringert werden.
- Die Berechnung der Regressionskoeffizienten mit der Methode kleinster Fehlerquadrate, erzeugt die Approximation der Simulation.
- Eine Zielfunktion wird definiert, welche die Systemantwort bewertet.
- Darauffolgend, kann eine Funktion erzeugt werden, die sich aus der Approximation der Simulation und der Zielfunktion zusammensetzt. Diese wird mit einem klassischen Optimierer nach den minimalen Funktionswerten optimiert.
- Zuletzt werden die Ergebnisse einer kritischen Beurteilung unterzogen und die Erfahrungen mit der Response Surface Methode werden in einem Fazit zusammengefasst.

2 Response Surface Methode und Design of Experiments

“Die Response Surface Methode ist eine Sammlung statistischer und mathematischer Methoden, die bei der Entwicklung, Verbesserung und Optimierung von Prozessen oder Produkten eingesetzt werden kann“ [Myers, Montgomery, & Anderson-Cook, 1995]. Um eine Approximation zu erstellen, werden Datenpunkte benötigt, welche Informationen über das Experiment (Eingabewerte) und das Ergebnis der Systemantwort enthalten. Die RSM beinhaltet unter anderem Methoden zur Planung dieser Experimente (Design of Experiments, DOE). Zudem enthält die RSM Verfahren zur Bestimmung der Koeffizienten der Approximationsmodelle und zur Bewertung der Response Surface Approximation (RSA). Im Folgenden Kapitel werden einige dieser Methoden vorgestellt, um eine Grundlage für die Verfahrensweise in den darauffolgenden Kapiteln zu schaffen.

2.1 Grundlagen

Die Systemantwort ist von den Entwurfsvariablen (Anzahl der Entwurfsvariablen, n_V) der jeweiligen Experimente abhängig:

$$y = y\left(\xi_1, \xi_2, \dots, \xi_{n_V}\right) = y(\xi) \tag{2.1}$$

Da die Response Surface Approximation (RSA) nur eine Annäherung dieser Funktion ist, kann man annehmen, dass ihre Berechnung bis auf einen Fehler der genauen Systemantwort gleicht:

$$y = \hat{y}\left(\xi_1, \xi_2, \dots, \xi_{n_V}\right) + \varepsilon = \hat{y}(\xi) + \varepsilon \tag{2.2}$$

Die Differenz zwischen der approximierten Lösung und der Systemantwort ist der Fehler der RSA:

$$\varepsilon = y(\xi) - \hat{y}(\xi) \tag{2.3}$$

Diese Differenz kann aus verschieden Ursachen resultieren. Wie in etwa durch systeminhärente Fehlerquellen, die numerischer oder modelbedingter Natur sind. Um eine Approximation $\hat{y}$ so zu erzeugen, dass der Fehler gering bleibt, stützt sich die RSM auf einen Datensatz, der vorher erzeugt wurde. Diese hat eine Anzahl von n_P Stützstellen.

Es ist üblich, die verwendeten natürlichen Entwurfsvariablen in dimensionslose Variablen zu transformieren. So können die Stützpunkte bewertet werden, da sie bezüglich ihrer Einheiten und Grenzwerte gleichwertig skaliert wurden. Jede Entwurfsvariable ξ_i der Stützstelle $\boldsymbol{\xi}(p)$ wird auf einen Wert zwischen -1 und 1 normiert:

$$x_{i(p)} = \frac{\xi_{i(p)} - \frac{\max_j(\xi_{i(j)}) + \min_j(\xi_{i(j)})}{2}}{\frac{\max_j(\xi_{i(j)}) - \min_j(\xi_{i(j)})}{2}} \quad mit \begin{cases} i = 1, 2, \dots, n_V \\ j, p = 1, 2, \dots, n_P \end{cases} \tag{2.4}$$

Wurden alle Entwurfsvariablen transformiert, kann der Ansatz von (2.5) folgendermaßen formuliert werden:

$$y = \hat{y}\left(x_1, x_2, \dots, x_{n_V}\right) + \varepsilon = \hat{y}(x) + \varepsilon \tag{2.5}$$

2.1.1 Ansatzfunktion

Die Ansatzfunktion muss die Systemantwort so gut wie möglich approximieren. Die Wahl der Funktion steht dem Anwender frei. Dabei können außer Polynomterme auch andere Zusammenhänge wie $(\ln(x), \frac{1}{x}, \sqrt{x} \dots)$ verwendet werden. Standartmäßig werden in der RSM, Polynomansätze mit maximal quadratischen Termen eingesetzt. Mit dem linearen Modell lassen sich die Haupteffekte der Entwurfsvariablen identifizieren:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_{n_V} x_{n_V} = \beta_0 + \sum_{i=1}^{n_V} \beta_i \, x_i \tag{2.6}$$

Mit dem quadratischen Ansatz können auch Interaktionen zwischen den Entwurfsvariablen erfasst werden. Der vollständige quadratische Ansatz kann viele verschiedene Funktionsverläufe darstellen:

$$\hat{y} = \beta_0 + \sum_{i=1}^{n_V} \beta_i \, x_i + \sum_{i=1}^{n_V} \sum_{j=1}^{n_V} \beta_{ij} \, x_i x_j \tag{2.7}$$

Die Koeffizienten werden mittels einer linearen Regression berechnet.

2.1.2 Kleinsten Quadrate

Eine Methode zur Bestimmung der Koeffizienten β ist die der Gauß'schen Fehlerquadratminimierung. Dabei ist die Zahl der Stützpunkte im Datensatz mindestens der Zahl der Regressionskoeffizienten $(n_P \geq n_V)$. Der Ansatz mit den jeweiligen Stützstellen, wird wie folgt formuliert:

$$\begin{aligned} y_{(p)} &= \beta_0 + \beta_1 x_{1(p)} + \beta_2 x_{2(p)} + \cdots + \beta_{n_V} x_{n_V(p)} + \varepsilon_{(p)} \\ &= \beta_0 + \sum_{i=1}^{n_V} \beta_i x_{i(p)} + \varepsilon_{(p)} \qquad\qquad mit \quad p = 1, 2, \ldots, n_p \end{aligned} \tag{2.8}$$

Die Matrixschreibweise lautet wie folgt:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{2.9}$$

$$\boldsymbol{y} = \begin{bmatrix} y_{(1)} \\ y_{(2)} \\ \vdots \\ y_{(n_P)} \end{bmatrix}, \qquad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_{(1)} \\ \varepsilon_{(2)} \\ \vdots \\ \varepsilon_{(n_P)} \end{bmatrix}, \qquad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{n_V} \end{bmatrix}, \qquad \boldsymbol{X} = \begin{bmatrix} 1 & x_{1(1)} & x_{2(1)} & \ldots & x_{n_V(1)} \\ 1 & x_{1(2)} & x_{2(2)} & \ldots & x_{n_V(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{1(n_P)} & x_{2(n_P)} & \ldots & x_{n_V(n_P)} \end{bmatrix}$$

Die Regressionskoeffizienten werden nach der Gauß'schen Methode so ausgewählt, dass die Summe der Fehler $\varepsilon_{(p)}$ zum Quadrat minimal werden:

$$L = \sum_{p=1}^{n_p} \varepsilon_{(p)}^2 = \sum_{p=1}^{n_p} \left(y_{(p)} - \beta_0 - \sum_{i=1}^{n_v} \beta_i \xi_{i(p)} \right)^2 \tag{2.10}$$

Die Matrixschreibweise lautet dementsprechend:

$$\begin{aligned} L &= \boldsymbol{\varepsilon}^T \boldsymbol{\varepsilon} = (\boldsymbol{y} - \boldsymbol{X\beta})^T (\boldsymbol{y} - \boldsymbol{X\beta}) \\ &= \boldsymbol{y}^T \boldsymbol{y} - \boldsymbol{\beta}^T \boldsymbol{X}^T \boldsymbol{y} - \boldsymbol{y}^T \boldsymbol{X\beta} + \boldsymbol{\beta}^T \boldsymbol{X}^T \boldsymbol{X\beta} \\ &= \boldsymbol{y}^T \boldsymbol{y} - 2\boldsymbol{\beta}^T \boldsymbol{X}^T \boldsymbol{y} + \boldsymbol{\beta}^T \boldsymbol{X}^T \boldsymbol{X\beta} \end{aligned} \tag{2.11}$$

Mit folgender Bedingung wird die Summe der Fehlerquadrate L minimiert:

$$\left. \frac{\partial L}{\partial \boldsymbol{\beta}} \right|_{\boldsymbol{b}=\left[b_0, b_1, \ldots, b_{n_V}\right]^T} = -2\boldsymbol{X}^T \boldsymbol{y} + 2\boldsymbol{X}^T \boldsymbol{X}\boldsymbol{b} = 0 \tag{2.12}$$

Die Regressionskoeffizienten sind somit eindeutig bestimmt:

$$\boldsymbol{b} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y} \tag{2.13}$$

Diese Koeffizienten bilden die approximierte Funktion:

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{b} \tag{2.14}$$

2.2 Beurteilungskriterien

Da die Optimierung an der RSA durchgeführt wird, kann sie nur so gut sein, wie die Approximation selbst. Daher gilt es diese auf ihre Genauigkeit zu überprüfen. Ziel ist es, keine zusätzlichen Systemantworten ermitteln zu müssen. Dabei bleiben zur Gütebestimmung nur die zur Approximation verwendeten Stützpunkte. Da diese fehlerbehaftet sein könnten, wird deren Einfluss auf die Approximation bestimmt.

Die Methode der kleinsten Quadrate, erlaubt eine unverzerrte Schätzung der Regressionskoeffizienten $\boldsymbol{\beta}$. Das beruht auf der Vermutung, dass die Fehler $\boldsymbol{\varepsilon}$ zufällig und unsystematisch auftreten [Myers, Montgomery, & Anderson-Cook, 1995].

$$\begin{aligned} E(\boldsymbol{b}) &= E[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}] \\ &= E[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{\beta}+\boldsymbol{\varepsilon})] \\ &= E[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta}+(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{\varepsilon})] \\ &= \boldsymbol{\beta} \end{aligned} \tag{2.15}$$

Die Kovarianzmatrix von $\boldsymbol{b}$ ist eine $n_K\ x\ n_K$ symmetrische Matrix, dessen Elemente $(j,\ j)$ die Varianz von b_j sind. Die Kovarianzmatrix von $\boldsymbol{b}$ ist:

$$Cov(\boldsymbol{b}) = \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1} \tag{2.16}$$

Die Summe der Fehlerquadrate, ist der erste Anhaltspunkt, aus diesem man Werte der Approximation entnehmen kann.

$$SS_E = \sum_{i=1}^{n_V} e_{(p)}^2 = \boldsymbol{e}^T\boldsymbol{e} \tag{2.17}$$

Das arithmetische Mittel der Fehlerquadrate ergibt sich durch das Einbeziehen der Freiheitsgrade der Regression. Zusätzlich stell es eine Schätzung für die Varianz der Regressionskoeffizienten dar:

$$\hat{\sigma}^2 = MS_E = \frac{SS_E}{n_P - n_K} \tag{2.18}$$

Weitere Werte sind, die Summe der Quadrate der Abweichung der Stützwerte von deren Mittelwert und die Summe der Quadrate der Abweichung der approximierten Werte des Regressionsmodells.

$$SS_T = \boldsymbol{y}^T \boldsymbol{y} - \frac{\left(\sum_{p=1}^{n_p} y_{(p)}\right)^2}{n_p} \tag{2.19}$$

$$SS_R = \boldsymbol{b}^T \boldsymbol{X}^T \boldsymbol{y} - \frac{\left(\sum_{p=1}^{n_p} y_{(p)}\right)^2}{n_p} \tag{2.20}$$

Um einen Bezug von der Summe der Fehlerquadrate SS_E zu der Anzahl und Verteilung der Stützpunkte zu schaffen, werden die statistischen Bewertungsgrößen SS_R und SS_T eingebracht.

$$R^2 = \frac{SS_R}{SS_T} = 1 - \frac{SS_E}{SS_T} \tag{2.21}$$

Das Bestimmtheitsmaß R^2 sagt aus, wie sich die Varianz der Approximation zur Gesamtvarianz gemindert hat. Dieser Wert kann zwischen null und eins liegen. Wohingegen ein Wert gegen eins nicht unbedingt heißt, dass die Approximation sehr gut ist. Es ist möglich diesen Wert zu verbessern, indem man mehr Regressionskoeffizienten verwendet.

Das adjustierte Bestimmtheitsmaß R^2_{adj} berücksichtigt die Zahl der Regressionskoeffizienten und Stützpunkte. Somit ist es ein geeigneterer Wert, um verschiedene Regressionsmodelle zu vergleichen.

$$R^2_{adj} = 1 - \frac{\frac{SS_E}{(n_P - n_K)}}{\frac{SS_T}{(n_P - 1)}} = 1 - \left(\frac{n_P - 1}{n_P - n_K}\right)(1 - R^2) \tag{2.22}$$

Das adjustierte Bestimmtheitsmaß wächst nicht zwangsweise mit der Zahl der Regressionskoeffizienten. Oftmals sinkt der Wert, wenn unnötige Terme hinzugefügt werden [Gleichmar, 2004].

In dieser Ausarbeitung wird auf Grund der Umfänglichkeit auf eine statistische Bewertung der Ergebnisse verzichtet.

2.3 Design of Experiments

Unter Design of Experiments (DOE) versteht man die Gestaltung der verwendeten Experimente und deren Parameter mit Hilfe von statistischen Methoden. Diese Stützstellen sollen eine möglichst genaue Anpassung des Regressionsmodells an das zu approximierende System ermöglichen. Dennoch müssen bei zeitaufwendigen Systemanalysen die Zahl der Experimente minimal gehalten werden.

Mit dem Full Factorial Design (FFD) kann das Prinzip der Versuchsplanung anschaulich gemacht werden. Dabei wird der Definitionsbereich jeder Entwurfsvariabel nach ihren Grenzwerten in gleich große Bereiche skaliert. In diesem Bereich werden die Entwurfsvariablen l mal gleichmäßig aufgeteilt $(Level)$. Das resultierende Experiment besteht beim FFD aus der Summe der Kombinationen aller möglichen Werte. Die Anzahl der Stützstellen, berechnet sich mit der Anzahl der Entwurfsvariablen n_V und den verwendeten Levels. Werden zwei Entwurfsvariablen und Levels verwendet, ist die Anzahl er Stützstellen $n_P = l^{nv} = 2^2 = 4$. Mit mehr Entwurfsvariablen steigt die Anzahl der Stützstellen exponentiell.

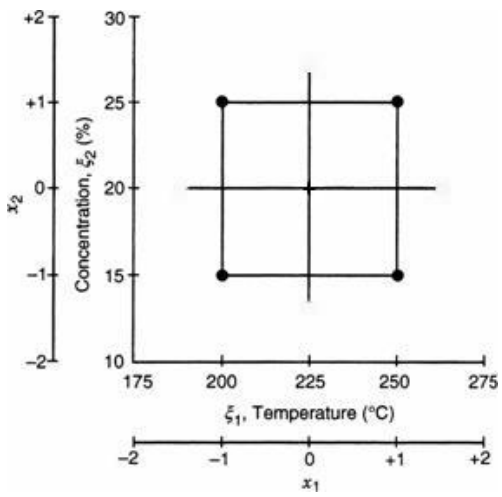

Abbildung 1: Beispiel eines 2^2 Factorial Designs eines chemischen Prozesses.

Bei einer Zahl von $n_V = 4$ Variablen und $l = 3$ Levels, sind schon 81 Stützstellen für ein FFD nötig. Die darin enthaltenen Stützstellen, tragen nicht zu gleichen Teilen zur Bestimmung der Regressionskoeffizienten bei. Einige Variablen und ihre Kombinationen haben einen sehr geringen Einfluss auf die Systemantwort.

Ein weiteres Problem bei statischen Designs ist, das die Grenzwerte, die zur Berechnung des Definitionsbereichs, nicht variieren dürfen. Bei der Auslegung einer elektrischen Maschine ist nahezu jeder Grenzwert vom Wert eines anderen Parameters abhängig. Somit ergibt sich ein unregelmäßig geformter Designraum.

2.3.1 D-Optimales Design

Mit der optimalen Versuchsplanung, auch “Optimal Design of Experiments“ genannt, wird versucht, optimale Designs nach verschiedenen Kriterien für spezifische Anforderungen zu erstellen. Eine dieser Kriterien ist die D-Optimalität. Dabei wird aus einem Pool möglicher Stützstellen, jene zu einem Experiment mit n_p Stützstellen zusammengestellt, welche die Determinante der Informationsmatrix $\boldsymbol{X}^T\boldsymbol{X}$ maximal werden lässt.

$$\max_{\zeta} |\boldsymbol{X}^T\boldsymbol{X}| \tag{2.23}$$

Die Informationsmatrix ist proportional zur Inversen der Kovarianzmatrix $Cov(\boldsymbol{b}) = \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}$, was bei einer Maximierung von $|\boldsymbol{X}^T\boldsymbol{X}|$ bedeutet, dass die Determinante der Kovarianzmatrix von $\boldsymbol{b}$ minimiert wird. Dieser Vorgang bewirkt also, das die ausgewählten Stützstellen, kleine Varianzen des Regressionsvektors $\boldsymbol{b}$ erzeugen [Gleichmar, 2004].

2.3.2 Design-Matrix

Die Designmatrix $\boldsymbol{X}$ ist eine $n_p \; x \; n_k$ Matrix, die abhängig ist, von der Anzahl der Koeffizienten n_K. Wie viele Experimente verwendet werden kann selbst festgelegt werden, wobei es mindestens $n_P \geq n_V$ Stützstellen sein müssen. In dieser Arbeit wurde ein vollständiger quadratischer Ansatz verwendet. Dabei kann es vorkommen, dass die Determinante der Informationsmatrix $\boldsymbol{X}^T\boldsymbol{X}$ **,** durch sehr hoch korrelierende Variablen, singulär wird. Um ein stabiles numerisches Verhalten zu gewährleisten, können Terme aus der Modelfunktion entfernt werden [Triefenbach, 2008].

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{11} x_{11} + \cancel{\beta_{12} x_{12}} + \beta_{22} x_{22} + \varepsilon \qquad (2.24)$$

3 Anwendung der Response Surface Methode mit Python

Die Anwendung der Methoden, die in dem Kapitel zwei beschrieben wurden, werden in diesem Kapitel mit der Programmiersprache Python umgesetzt. Dabei werden die Vorgehensweisen zur Lösung der Probleme, die sich dabei ergeben, beschrieben. Das Simulationsprogramm “FEMAG-DC“ wurde für diesem Zweck als eine Python Funktion eingebunden und wird im Folgenden nur als solche aufgerufen.

3.1 Beschreibung des Simulationsmodells

Das Simulationsmodel ist ein permanentmagneterregter Synchronmotor, dessen Kenngrößen mittels der Finite-Elemente-Methode berechnet werden. Die eingestellte PM-Simulation berechnet dafür in mehrfachen transienten Analysen die Vektorpotenziale der zweidimensionalen Magnetfelder. Der Potenzialverlauf wird in der FE-Methode diskreditiert durch die diskreten Potenziale in den Knotenpunkten. Diese sich zu einem Element ergebenen Knotenpunkte, können eine lineare oder quadratische Approximation des Bereiches bilden. Der Abstand der Knotenpunkte, die das Element bilden und die Art des Models, bestimmen die Genauigkeit der Approximation. Es ist zu empfehlen, für die RSM einen linearen Verlauf zu wählen und das Netz nicht zu engmaschig zu gestalten. Um die Simulationsdauer zu verkürzen, wird nur ein Teil des rotationssymmetrischen Modells berechnet.

Die Auslegungsparameter haben durch die magnetische Sättigung im Material und die Entmagnetisierung in den Permanentmagneten einen nicht linearen Zusammenhang. In Abbildung 2 ist zu sehen, dass das Material ab einem Wert bei etwa 1,5 Tesla keinen linearen Zusammenhang mehr aufweist. Diese Grenzwerte werden regelmäßig überschritten und sollen im Regelfall ausgereizt werden.

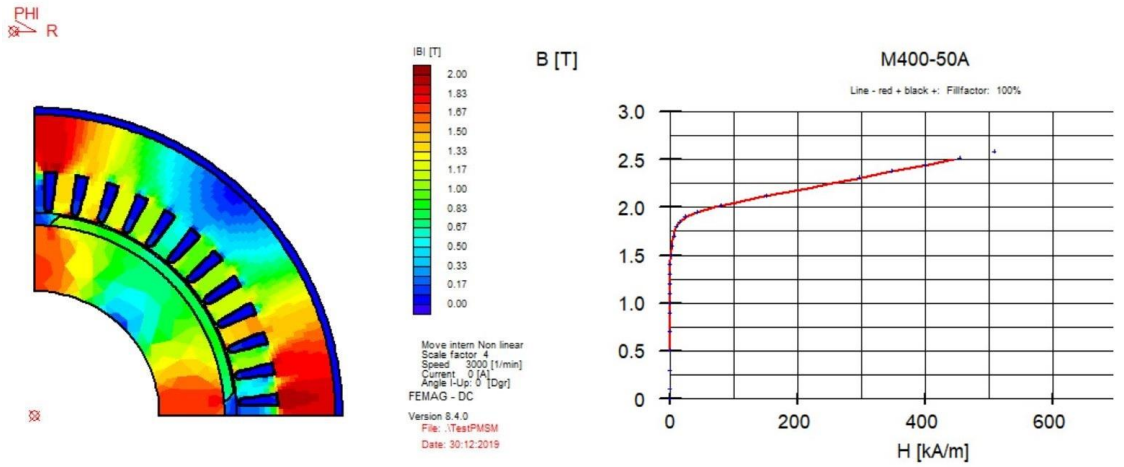

Abbildung 2: Magnetflussdichte im Material und Entmagnetisierungskurve.

Als variable Auslegungsparameter können alle unabhängigen Entwurfsparameter gewählt werden. Dabei sind auch diskrete Werte wie Materialkennlinien eine Option. Die variablen Entwurfsparameter sind in dieser Arbeit kontinuierliche Werte, die den groben Aufbau der geometrischen Auslegung beschreiben.

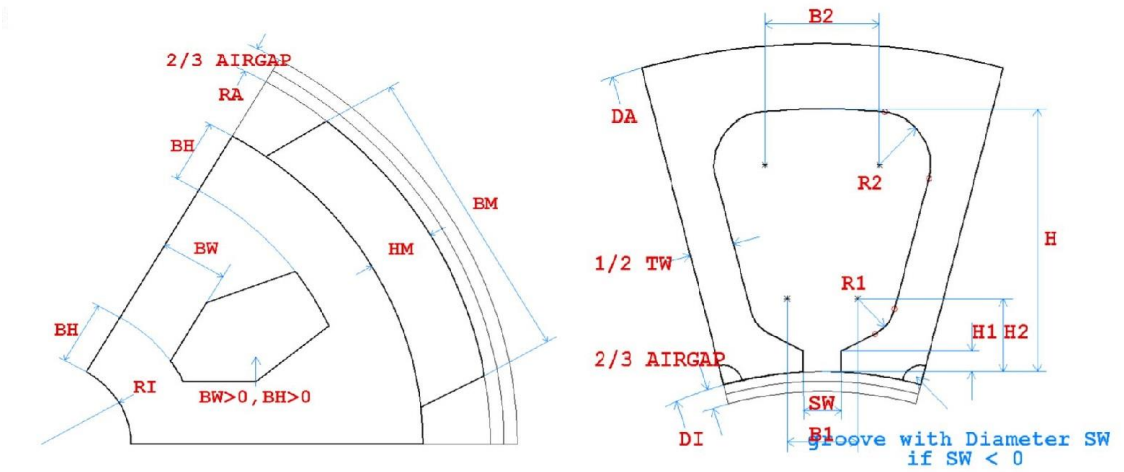

Abbildung 3: Schema der geometrischen Parameter des Rotors (links) und Stators (rechts).

Die Entwurfsvariablen sind folgende:

- DA: Außen Durchmesser (m.yoke_diam)
- H: Nuttiefe (m.slot_height)
- TW: Zahnbreite (m.tooth_width)
- RA: Magnet außen Radius (m.mag_rad)
- BT: Bautiefe (m.arm_lenght)
- HM: Magnethöhe (m.mag_height)

Viele der in Abbildung 3 zu sehenden Parameter, sind von diesen Entwurfsvariablen abhängig. Andere, wie z.B. der Luftspalt (AIRGAP) müssen als eine feste Größe in das Entwurfsmodell eingebracht werden. Es ist wichtig, die Grenzen der Entwurfsvariablen zu definieren, um eine reibungslose Variierung der Entwurfsvariablen zu gewährleisten. Einige der Grenzwerte der Entwurfsvariablen stehen in Abhängigkeit zu Werten anderer. Manche wurden nach eigenem Ermessen ausgewählt und beruhen auf der Gewährleistung mechanischer Stabilität.

Die oberen Grenzen $\max_j(\xi_{i(j)})$ und die unteren Grenzen $\min_j(\xi_{i(j)})$ der einzelnen Variablen $\xi_{i(p)}$ werden in Python als Abfragefunktion berechnet. Dafür wird die Stützstelle $\xi_{(p)}$ mit der zu bestimmenden Variable i in die Funktion eingelesen. Diese gibt den Maximalwert oder Minimalwert der entsprechenden Variablen zurück. Die folgende Tabelle beinhaltet die Grenzwerte der einzelnen Entwurfsvariablen.

$\boldsymbol{\xi_{i(p)}}$	$\boldsymbol{\max_{p}(\xi_{i(p)})}$	$\boldsymbol{\min_{p}(\xi_{i(p)})}$
$\xi_{1(p)} = \mathrm{DA}$	$(RA * 2) + 4mm + H + 200mm$	$2 * (RA + H) + 44mm$
$\xi_{2(p)} = \mathrm{H}$	$\frac{(DA - 390mm)}{2} - 10mm$	$30mm$
$\xi_{3(p)} = \mathrm{TW}$	$\frac{\pi * 390mm}{48} - 10mm$	$10mm$
$\xi_{4(p)} = RA$	$250mm$	$153mm$
$\xi_{5(p)} = BT$	$300mm$	$50mm$
$\xi_{6(p)} = HM$	$RA - 140mm$	$5mm$

Tabelle 1: Maximale und minimale Grenzwerte der einzelnen Entwurfsvariablen.

Hat man Kenntnis über das Verhalten in bestimmten Bereichen, ist zu empfehlen die Grenzwerte nahe der vermuteten Lösung zu platzieren. Das verbessert das Ergebnis der RSA oder vermindert die Berechnungszeit durch Reduktion der benötigten Levels.

Die Simulation bietet als Ergebnis eine sehr große Informationsmenge, die eine sehr detaillierte Darstellung der Verhältnisse ermöglicht. Welche Ergebnisse benötigt werden, hängt von der Zielfunktion ab. Aus welchem Grund die folgenden Werte ausgewählt wurden, wird im Kapitel (3.3.1) deutlich.

M	U_{eff}	P_{VE}	P_{VW}	$cos\varphi$
$Moment$	$Spannung$	$Eisenverluste$	$Wicklungsverluste$	$Leistungsfaktor$

Tabelle 2: Relevante Ergebnisse für die RSA.

3.2 Entwicklung des Metamodells

Die Stützstellen, die für die Berechnung des Response Surface Approximation benötigt werden, sind die D-optimalen Stützstellen, aus einem Pool aus gültigen Stützstellen. Ob eine Stützstelle gültig ist, bestimmt die Tatsache, ob die einzelnen Entwurfsvariablen der zufällig erzeugten Stützstelle seine Grenzwerte einhält. Um ein glaubwürdiges D-optimum zu erhalten, ist es notwendig, dass die gültigen Stützstellen den kompletten Designraum repräsentieren.

Demzufolge muss zunächst ein Raster an Stützstellen erzeugt werden. Der Ausgangspunkt dafür ist die Stützstelle die der Anwender als bestmöglich einstuft. Um den Designraum bestmöglich abzudecken, werden die Start- und Endpunkte des Rasters vom Minimal- und Maximalwert festgelegt. Die Anzahl der erzeugten Kombinationen n_A ist von der Schrittweite s abhängig, die zwischen Anfang- und Endpunkt gewählt wird. Die Schrittweite ist ähnlich der Levels beim FFD. Wobei durch die dynamischen Grenzen einige der Stützpunkte nicht gültig sind, was bedeutet, dass die Schrittweite nicht das Level der Designmatrix wiedergibt.

$$n_A = s^{n_v} \tag{3.1}$$

Die Schrittweite muss bei steigenden Entwurfsvariablen immer kleiner gewählt werden, um die Zahl der Stützstellen im Rahmen zu halten. Hierfür ist es möglich, durch Umstellen der Gleichung (3.2) die Anzahl der Kombinationen vorzugeben. Um in den kommenden Verfahren zu hohe Rechenzeiten zu vermeiden, wurde hier die Anzahl der Kombinationen auf 4096 festgelegt.

Im erzeugten Raster sind nicht alle Stützstellen konstruierbar, da die Grenzwerte, die für die von Anwender eingegebene Stützstelle nicht für alle Stützstellen gültig ist. Um eine Liste von nur gültigen Stützstellen zu erzeugen, werden alle Stützstellen auf ihre Grenzwerte geprüft. Das passiert indem jeder Stützstelle ein Status zugewiesen wird. Der ist davon abhängig, ob die jeweilige Entwurfsvariable $\xi_{i(p)}$ ihre Grenzwerte überschreiten. Wenn das der Fall ist, wird ihr der Status “FALSE“ zugewiesen. Die jeweiligen Grenzwerte aus Tabelle.1 werden über die Funktionen “ub“, “lb“ abgerufen.

```
status  =   var[i] <= ub(var,i) and var[i]  >=  lb(var,i)
```

Code 1: Statusabfrage durch Vergleichsoperatoren (d_optimal_design.py Zeile 132).

Die Zahl der gültigen Stützstellen reduziert sich nach der Sortierung auf 1008 gültige Stützstellen. Diese Stützstellen haben allerdings einen gewissen Abstand zu den Grenzwerten der einzelnen Entwurfsvariablen. Das resultiert daraus, dass die Schrittweite, die relativ grob gewählt wird, den durchschnittlichen Abstand zu den Grenzen bestimmt.

Das D-optimale Design strebt eine möglichst weitläufige und gleichmäße Verteilung der Stützstellen an. Deshalb werden Stützstellen, die sich genau auf den Grenzen befinden zusätzlich in das Set der gültigen Stützstellen hinzugefügt.

Hierfür werden gültige Stützstellen so umgewandelt, das alle Entwurfsvariablen einer Stützstelle dem minimalen oder maximalen Wert entspricht.

	DA	H	TW	RA	BT	HM
1	$\min_{\boldsymbol{p}}(\xi_{1(\boldsymbol{p})})$	$\min_{\boldsymbol{p}}(\xi_{2(\boldsymbol{p})})$	$\min_{\boldsymbol{p}}(\xi_{3(\boldsymbol{p})})$	$\min_{\boldsymbol{p}}(\xi_{4(\boldsymbol{p})})$	$\min_{\boldsymbol{p}}(\xi_{5(\boldsymbol{p})})$	$\min_{\boldsymbol{p}}(\xi_{6(\boldsymbol{p})})$
2	$\min_{\boldsymbol{p}}(\xi_{1(\boldsymbol{p})})$	$\min_{\boldsymbol{p}}(\xi_{2(\boldsymbol{p})})$	$\min_{\boldsymbol{p}}(\xi_{3(\boldsymbol{p})})$	$\min_{\boldsymbol{p}}(\xi_{4(\boldsymbol{p})})$	$\min_{\boldsymbol{p}}(\xi_{5(\boldsymbol{p})})$	$\max_{\boldsymbol{p}}(\xi_{6(\boldsymbol{p})})$
⋮	⋮	⋮	⋮	⋮	⋮	⋮
2^{n_V}	$\max_{\boldsymbol{p}}(\xi_{1(\boldsymbol{p})})$	$\max_{\boldsymbol{p}}(\xi_{2(\boldsymbol{p})})$	$\max_{\boldsymbol{p}}(\xi_{3(\boldsymbol{p})})$	$\max_{\boldsymbol{p}}(\xi_{4(\boldsymbol{p})})$	$\max_{\boldsymbol{p}}(\xi_{5(\boldsymbol{p})})$	$\max_{\boldsymbol{p}}(\xi_{6(\boldsymbol{p})})$

Tabelle 3: Alle möglichen Grenzwertkombinationen.

Werden alle Duplikate entfernt, verringert sich die Anzahl der Randstützpunkte enorm. Das liegt daran, dass acht der zwölf Grenzwerte für jeden Stützpunkt gleich sind. Die Summe dieser Randstützpunkte und die des Rasters, bilden den Stützstellenpool. Es entsteht eine 1248 x 6 Matrix.

	DA	H	TW	RA	BT	HM
1	$510mm$	$30mm$	$10mm$	$153mm$	$50mm$	$5mm$
2	$510mm$	$30mm$	$10mm$	$153mm$	$133.33mm$	$5mm$
⋮	⋮	⋮	⋮	⋮	⋮	⋮
1248	$734mm$	$30mm$	$15.52mm$	$250mm$	$300mm$	$110mm$

Tabelle 4: Stützstellenpool aus Randstützpunkten und gültigen Stützpunkten.

3.2.1 D-Optimale Designmatrix

Die Kriterien des D-Optimalen Designs werden auf eine Designmatrix angewendet. Diese Designmatrix ist eine $n_p \; x \; n_k$ Matrix, bei der n_p die Anzahl der Stützstellen repräsentiert. Wie viel Stützstellen eingesetzt werden, steht dem Anwender zum Teil frei. In dieser Arbeit wurde n_p nach der Vorgehensweise in Abbildung 4 berechnet.

$$\begin{aligned} Lineare\ Haupteffekte \quad & n_V + 1 = 7 \\ 2 - Faktor - Wechselwirkung \quad & n_V * \frac{n_V - 1}{2} = 15 \\ Quadratische\ Effekte \quad & n_V = 6 \\ Versuchsvarianz \quad & (\geq 1) = 7 \\ & n_P = 35 \end{aligned}$$

Abbildung 4: Anzahl der Versuche im erstellten Design [Dr. Waschatz, 2003].

Die endgültige Anzahl der Regressionskoeffizienten n_k entscheidet sich nachdem die korrelierenden Terme aus dem vollständigen quadratischen Ansatz entfernt wurden. Um diese Korrelation zu erkennen, wird der in Kapitel 2.3.2 thematisierte Zusammenhang zur Determinante der Informationsmatrix untersucht. Dabei wird zunächst das Set ξ,

dass aus zufälligen Stützpunkten des Stützstellenpools besteht, nach der Gleichung (2.4) normalisiert.

$$\boldsymbol{\xi} = \begin{bmatrix} \xi_{1(1)} & \cdots & \xi_{n_V(1)} \\ \vdots & \ddots & \vdots \\ \xi_{1(n_P)} & \cdots & \xi_{n_V(n_P)} \end{bmatrix} \; ; \quad \boldsymbol{X} = \begin{bmatrix} x_{1(1)} & \cdots & x_{n_V(1)} \\ \vdots & \ddots & \vdots \\ x_{1(n_P)} & \cdots & x_{n_V(n_p)} \end{bmatrix} \tag{3.3}$$

Die Matrix $\boldsymbol{X}$ wird anschließend mit den fehlenden Termen des vollen quadratischen Ansatzes erweitert. Es wird nun die Determinante dieser Matrix berechnet. Es wird der Reihe nach immer eine Spalte entfernt und dann die Determinante dieser Matrix berechnet. Nachdem alle Spalten einmal entfernt wurden, wird jene Spalte bestimmt, dessen Entfernen die Determinante am meisten steigen lässt. Dieser Vorgang wird wiederholt, bis die Determinante einen Wert größer gleich 1 hat. Die verbleibenden Terme bilden die Ansatzfunktion der Approximation. Falls, wie in diesem Fall, die Determinante größer als 1 ist, werden keine Terme entfernt.

−−−−−−−Koeffizienten großer Korrelation werden entfernt−−−−−−−
Bis Det(X′X) > 1
Det(Xn′Xn) = 110254798.53305544

Abbildung 5: Konsolen Ausgabe der Determinante des vollen quadratischen Ansatzes.

Um die Determinante der Informationsmatrix zu maximieren, wird das Fedorov-Austauschverfahren angewendet. Bei diesem Verfahren wird die Größe der Designmatrix nicht verändert. Es wird aus einer zufällig zusammengestellten Designmatrix ein Stützpunkt x_i ausgewählt. Dieser wird mit einem Stützpunkt x_j, aus dem Stützstellenpool ausgetauscht. Das Entfernen und Hinzufügen von Stützpunkten kann folgendermaßen formuliert werden:

$$(\boldsymbol{X}_{neu}^T \boldsymbol{X}_{neu}) = (\boldsymbol{X}_{alt}^T \boldsymbol{X}_{alt}) - (x_i * x_i^T) + \left(x_j * x_j^T\right) \tag{3.4}$$

Der Unterschied vom Fedorov-Algorithmus zum Standard-Austauschverfahren ist, dass eine „delta“-Funktion berechnet wird, welche die Veränderung der Determinante beschreibt:

$$|\boldsymbol{X}_{neu}^{T}\boldsymbol{X}_{neu}| = |\boldsymbol{X}_{alt}^{T}\boldsymbol{X}_{alt}| * \left(1 + \Delta(x_i, x_j)\right) \tag{3.5}$$

Die „delta“-Funktion nutzt die Varianz des Schätzers einer Stützstelle x_z:

$$d(x_z) = x_z^T * \left(\boldsymbol{X}_n^T\boldsymbol{X}_n\right)^{-1} * x_z \tag{3.6}$$

$$\Delta(x_i, x_j) = d(x_j) - \left[d(x_i)\, d(x_j) - \left(d(x_i, x_j)\right)^2\right] - d(x_i) \tag{3.7}$$

$$d(x_i, x_j) = x_i^T * \left(\boldsymbol{X}_n^T\boldsymbol{X}_n\right)^{-1} * x_j = x_j^T * \left(\boldsymbol{X}_n^T\boldsymbol{X}_n\right)^{-1} * x_i \tag{3.8}$$

Es werden für jede Stützstelle in der Designmatrix $(i = 1, 2, \dots, n_P)$ jeweils alle Stützstellen aus dem Stützstellenpool eingesetzt. Die „delta“-Werte werden alle in einer Liste gespeichert. Die Stützstelle x_j, dessen einsetzen in die Stützstelle x_i, den höchsten Wert für $\Delta(x_i, x_j)$ hat, wird für x_i eingesetzt. Dieser Vorgang wird solange wiederholt, bis keine Verbesserung mehr eintritt.

```
        while status    ==  True:
            deltamaxlist    =   []
            print(' ')
            for i in range(len(X)):

                deltalist         =   []

                for j in range(len(Xall)):
                    delta   =   self.deltamaker(X[i],Xall[j],X)
                    deltalist.append([delta,i,j])

                deltalist   =   sorted(deltalist, key=itemgetter(0), re
                                                                verse=True)
                deltamax    =   deltalist[0]
```

```
            if deltalist[0][0] > 1:
                X[deltamax[1]]          =   Xall[deltamax[2]]
                random[deltamax[1]]     =   C[deltamax[2]]

                deltamaxlist.append(deltamax)
            else:
                pass

        if deltamaxlist[durchlauf][0]   >   1+1e-2:
            status              =   True
        else:
            status              =   False
        durchlauf   +=  1
```

Code 2: Fedorov-Algorithmus (d_optimal_design.py Zeile 280).

Durch Sortieren der Werte von Delta, ist bekannt, welcher Austausch den größten „delta“-Wert hat. Ist “deltamax“ des jeweiligen Durchlaufes von i größer als 1, wird $\boldsymbol{X}_{imax}$ mit $\boldsymbol{X}all_{jmax}$ ausgetauscht. Ist der größte Wert der jeweiligen “deltamax“ eines Durchlaufes kleiner als $1 + 10^{-2}$, wird die Schleife abgebrochen.

$$---------Zeilenaustauschverfahren-------$$

$$\begin{array}{lll}
\mathrm{Det(X'X):}\ 110254798.53305544 & \text{Durchlauf: } 1 & \text{IndexX: } 1 \\
\mathrm{Det(X'X):}\ 21797519177.02385 & \text{Durchlauf: } 1 & \text{IndexX: } 2 \\
\vdots & \vdots & \vdots \\
\mathrm{Det(X'X):}\ 1.3968964239692487\mathrm{e}+29 & \text{Durchlauf: } 1 & \text{IndexX: } 35
\end{array}$$

$$\begin{array}{lll}
\mathrm{Det(X'X):}\ 1.6262447445277156\mathrm{e}+29 & \text{Durchlauf: } 2 & \text{IndexX: } 1 \\
\mathrm{Det(X'X):}\ 1.6262447445277156\mathrm{e}+29 & \text{Durchlauf: } 2 & \text{IndexX: } 2 \\
\vdots & \vdots & \vdots \\
\mathrm{Det(X'X):}\ 3.7713956927657996\mathrm{e}+29 & \text{Durchlauf: } 2 & \text{IndexX: } 35
\end{array}$$

$$\begin{array}{lll}
\mathrm{Det(X'X):}\ 4.1445943004332035\mathrm{e}+29 & \text{Durchlauf: } 3 & \text{IndexX: } 1 \\
\mathrm{Det(X'X):}\ 4.1445943004332035\mathrm{e}+29 & \text{Durchlauf: } 3 & \text{IndexX: } 2 \\
\vdots & \vdots & \vdots \\
\mathrm{Det(X'X):}\ 4.3163753667999846\mathrm{e}+29 & \text{Durchlauf: } 3 & \text{IndexX: } 35
\end{array}$$

Abbildung 6: Werte der Determinanten nach dem Austauschverfahren in den Durchläufen der “while“-Schleife.

Nachdem die Designmatrix $\boldsymbol{X}$ nach den Kriterien des D-Optimalen Designs angepasst wurde, wird die Designmatrix von nun an mit dem Formelzeichen $\boldsymbol{D}$ beschrieben.

3.2.2 Parallele Berechnung der Systemantwort

In die Funktion „FEMAG-DC“ können nur die sechs Entwurfsvariablen in ihrer ursprünglichen Form eingelesen werden. Aus diesem Grund wurde der Zeilenaustausch des Fedorov-Algorithmus auf eine unskalierte Form des zufälligen Sets $\boldsymbol{\xi}$ angewendet. Dieses Set erfüllt somit die Kriterien des D-Optimalen-Designs und wird mit $\boldsymbol{\xi}_D$ beschrieben.

Eine Möglichkeit die Rechenzeit bei aufwendigen Systemanalysen zu verkürzen, ist die Parallelisierung. Mit Hilfe der Funktionen “Multiprocessing“ und “Threading“ wird die Berechnung der 35 Systemanalysen auf die Prozessoren des Rechners aufgeteilt. Das verkürzt die Rechenzeit zwar nicht proportional zur Anzahl der Prozessoren, dennoch kann eine erhebliche Zeitersparnis verzeichnet werden. Die zu berechnenden Stützstellen werden gleichmäßig auf die Prozessoren verteilt. Diese Stützstellenpakete werden in den Prozessoren in eine Queue geladen, die wie ein Briefkasten fungiert. Aus diesem Briefkasten entnehmen zwei Threads Aufgaben und bearbeiten sie. Das geschieht so lange, bis alle Aufgaben im Briefkasten bearbeitet wurden. Die Anzahl der Threads kann erhöht werden, dass bringt jedoch nicht immer eine Zeitersparnis, da sich die Threads die Rechenleistung des Prozessors teilen.

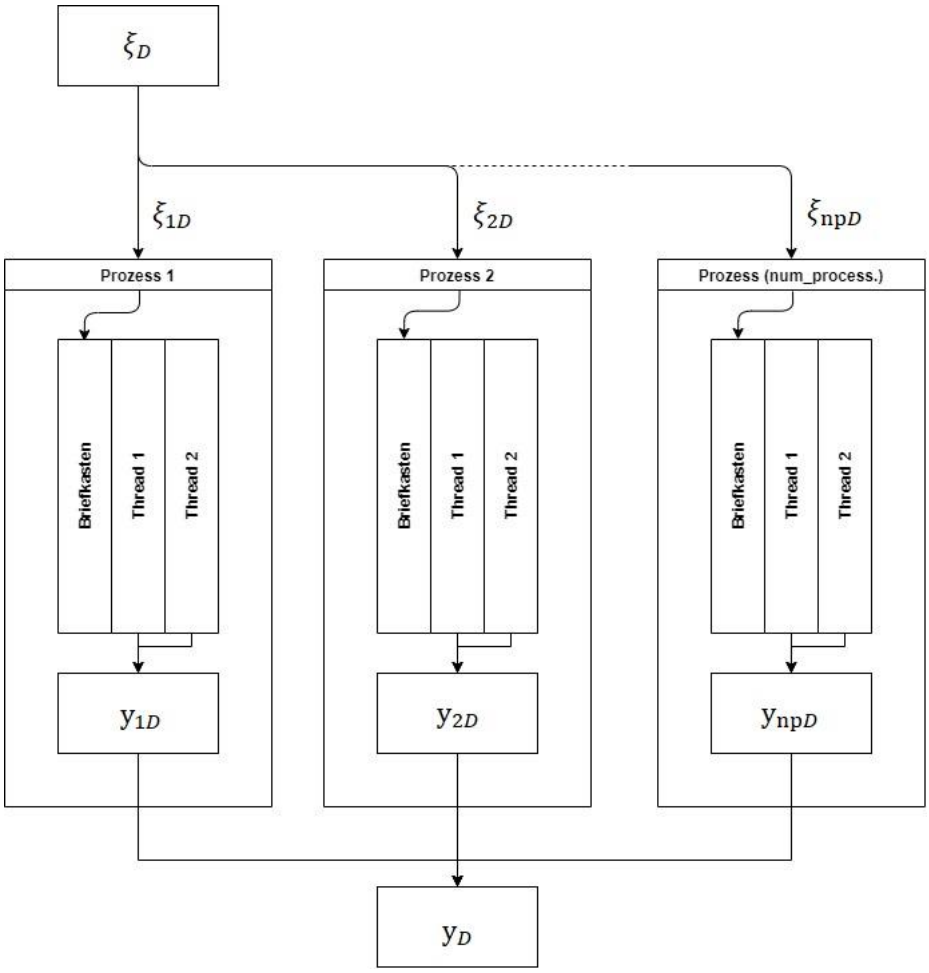

Abbildung 7: Flussdiagramm zum Ablauf des Multithreading.

Nachdem die Systemantworten berechnet wurden, werden die Ergebnisse mit ihren Entwurfsvariablen in ein “Log-file“ gespeichert. In diesem Ordner werden alle Ergebnisse dieses Systems gespeichert. Somit können frühere Berechnungen hinzugezogen werden, um eine genauere RSA zu erzeugen.

3.2.3 Regressionskoeffizienten

Die Regressionskoeffizienten werden anhand der Gleichung (2.1.3) berechnet. Werden die Systemantworten früherer Berechnungen und die des aktuellen Experiments für die RSA genutzt, müssen die Entwurfsparameter der hinzugefügten Stützstellen der aktuellen Designmatrix angepasst werden. Aus diesem Grund wird bei der Löschung der

Terme des vollen quadratischen Ansatzes, die Information darüber, welche Terme gelöscht würden, mitgegeben. In dieser Arbeit wird die RSA nur mit den Stützstellen des optimalen Designs berechnet. So kann das D-Optimale-Design einer besseren Bewertung unterzogen werden. Damit $\boldsymbol{X}$ aus Gleichung (2.1.3) bei der späteren Optimierung nicht immer skaliert werden muss, werden die Regressionskoeffizienten im Bezug auf die unskalierten Entwurfsvariablen berechnet. Dafür wird für die unskalierte Entwurfsmatrix $\boldsymbol{\xi}_D$ der volle quadratische Ansatz gebildet und mit der Information über die entfallenden Terme angepasst. Diese Matrix ist somit die unskalierte $\boldsymbol{D}$ Matrix und wird als $\boldsymbol{\xi}_{DV}$ bezeichnet.

Die Koeffizienten werden für jedes Ergebnis separat angepasst, um ein besseres Ergebnis zu erzeugen. Folglich wird die Vorgehensweise für die Berechnung der Koeffizienten für das Moment dargestellt:

$$b_M = \left(\boldsymbol{\xi}_{\boldsymbol{DV}}^{\boldsymbol{T}}\boldsymbol{\xi}\right)^{-1}\boldsymbol{\xi}^T y_M \tag{3.9}$$

3.3 Optimierung des Metamodells

Die kurze Berechnungszeit der Approximation ermöglicht es einen klassischen Optimierer mit vielen verschiedenen Startparametern zu verwenden. Das beste Ergebnis der lokalen Optima wird dann als Lösung ausgewählt. Hierbei ist zu beachten, dass es beim mathematischen Modell Kombinationen der Entwurfsvariablen möglich sind. Diese im Simulationsprogramm nicht umgesetzt werden können. Solche Ergebnisse sind nicht zugelassen und müssen durch Restriktionen vermieden werden.

3.3.1 Zielfunktion

Die Zielfunktion soll die Werte der fünf Ergebnisse zusammenfassend bewerten und mit einem Wert beschreiben. Hierfür muss man wissen, welcher Wert für die einzelnen Ergebnisse optimal ist. Ergebnisse werden nicht immer direkt in die Zielfunktion eingesetzt, da sie in die gesuchten Parameter umgerechnet werden müssen. Ziel ist es, nicht nur die Auslegungsparameter wie Spannung, Leistung und Wirkungsgrad optimal zu erreichen, sondern es werden in der Regel noch andere Bedingungen gestellt. Diese sind

meist wirtschaftlicher Art und sollen helfen die Kosten für die Maschine bei gleichbleibender Funktionalität zu minimieren. Bei permanenterregten Maschinen stellen die Magnete, die aus seltenen Erden gewonnen werden, eine wirtschaftliche und ökologische Belastung dar. Aus diesem Grund wird versucht, das Volumen der Magneten in der Maschine minimal zu halten. Dabei ist ein Teil der Zielfunktion an die Werte der Entwurfsvariablen geknüpft:

$$V_M = \ BT * HM * 2\pi * (RA - HM) * 0{,}85 \qquad (3.10)$$

Die einzuhaltenden Auslegungsparameter sind:

- Abgegebene Leistung $P_{ab} = 36{,}5\text{KW}$
- Betriebsspannung $U_{eff} = 400\text{V}$
- Leistungsfaktor $cos\varphi = 1$
- Wirkungsgrad $\eta = 1$

Einige dieser Parameter sind nur Richtwerte, die technisch nicht eingehalten werden können. Um zu verhindern, dass der steilste Abstieg der Zielfunktion in eine ungünstige Richtung verläuft, müssen die Gewichtungen der einzelnen Ziele angepasst werden. Dies stellt eine neue Herausforderung dar, denn es ist vor allem bei komplexeren Aufgaben nicht direkt ersichtlich, wie die einzelnen Parameter im Verhältnis zum Ziel stehen. Eine Möglichkeit wäre es, verschiedene gültige Auslegungen einer fachlichen Bewertung zu unterziehen. Diese Daten könnten genutzt werden, um die Gewichtungen der Zielfunktion anzupassen. In dieser Arbeit wurden die Gewichtungen der Zielfunktion so angepasst, dass der prozentuale Anstieg vom Optimum der Parameter in etwa gleich ist. Das vermeidet große Ausreißer, ist jedoch keine spezifische Zielfunktion. Der Wert des Faktors a_4 entscheidet, wie wichtig die Minimierung des Magnetvolumen im Vergleich zu den anderen Parametern ist.

$$\eta = \frac{M * 2\pi * 50s^{-1}}{M * 2\pi * 50s^{-1} + P_{VE} + P_{VW}} \tag{3.11}$$

$$z = \left(1 - \frac{M * 2\pi * 50s^{-1}}{36500W}\right)^{2} * a_1 + (1 - cos\varphi)^2 * a_2 + \left(1 - \frac{U_{eff}}{400V}\right)^{2} * a_3 + (1 - \eta)^2 * a_4 + V_M * a_4 \tag{3.12}$$

Die Zielfunktion wird über die Regressionskoeffizienten mit den Entwurfsvariablen verknüpft.

3.3.2 Optimierung

Um die RSA in Kombination mit der Zielfunktion zu optimieren, wird eine Funktion geschrieben, die bei der Eingabe der Entwurfsparameter die quadratischen Terme hinzufügt. Das geschieht wieder durch die Mitgabe der Information über die gelöschten Terme. Anschließend wird $\hat{y}$ berechnet und in die Zielfunktion eingesetzt. Das Ergebnis diese Funktion wird mit einem Optimierer der „Scipy“-Bibliothek minimiert. Dabei ist es wichtig, dass die Entwurfsparameter in den festgelegten Grenzen bleiben. Die festen Grenzen aus Tabelle 1 werden in die Variable „bnd“ geschrieben. Alle anderen Grenzen werden mit „np.inf“ als unendlich und somit als nicht definiert festgelegt. Die Grenzen, die in Abhängigkeit zu einer anderen Entwurfsvariablen stehen, werden als Einschränkung oder „constraint“ definiert. Eine Liste von Einschränkungen wird erstellt. Diese trägt Informationen, welche Funktion die Einschränkungsinformationen enthält und wie diese zu interpretieren sind. Es gibt den Typ „ineq“ für ungleich und „eq“ für gleich.

Um mehrere Lösungen zu erhalten, werden alle Stützpunkte des D-optimalen Design verwendet. Diese werden in einer Schleife als Startwert „x0“ eingesetzt. Die Ergebnisse und Entwurfsvariablen, der jeweiligen Optimierungsläufe, werden in eine Liste eingelesen.

```
cons = ({'type':'ineq','fun':constraint00},
        {'type':'ineq','fun':constraint01},
```

```
        {'type':'ineq','fun':constraint02},
        {'type':'ineq','fun':constraint03})


bnd     =     ((np.inf,np.inf),(30,np.inf),
               (10,15.525),(153,250),(50,300),(5,np.inf))

for i in range(len(X)):


    x0  =   X[i]

    op  =   optimize.minimize(optimierungsfunktion, x0, args=(B,ent_sp)
                              , method='SLSQP', bounds=bnd,
                               constraints=cons,options={'disp':True})

    optimalx    =   op.x
    for i in range(len(optimalx)):
        optimalx[i]=round(optimalx[i],2)

    E.append([round(op.fun,2),list(op.x)])


E       =   sorted(E, key=itemgetter(0), reverse=False)

Eopt    =   E[1]
```

Code 3: Optimierung mit mehreren Startwerten und Grenzwerten für die Entwurfsvariablen.

An den einzelnen Durchläufen ist zu sehen, dass manchmal verschiedene Startpunkte im selben Minimum konvergieren. Andere wiederum können stark abweichende Lösungen liefen. Das hebt nochmal hervor, welchen Vorteil die Response Surface Methode hat. Denn bei den 35 Durchläufen wurde die RSA und die Zielfunktion 1407 mal berechnet.

```
00:[197 820.07, [526.27, 46.44, 15.52, 153.0, 50.0, 5.0]]
01:[197 820.11, [526.63, 46.49, 15.52, 153.0, 50.0, 5.0]]
```

```
02:[197 820.12, [525.71, 46.34, 15.52, 153.0, 50.0, 5.0]]
03:[197 820.67, [527.56, 46.44, 15.52, 153.0, 50.0, 5.0]]
04:[197 820.91,   [527.95, 47.2, 15.52, 153.0, 50.0, 5.0]]
05:[197 821.1,   [523.86, 45.44, 15.52, 153.0, 50.0, 5.0]]
06:[197 821.43, [526.05, 46.54, 15.52, 153.0, 50.0, 5.0]]
07:[197 823.99,    [526.8, 46.62, 15.52, 153.0, 50.0, 5.0]]
                                ...
33:[1 708 274.64, [550.0, 30.0, 10.0, 217.67, 300.0, 5.0]]
34:[1 970 154.68, [734.0, 30.0, 15.52, 250.0, 300.0, 5.0]]
```

Abbildung 8: Liste mit den Ergebnissen der 35 Optimierungsdurchläufe.

4 Bewertung der Optimierung

Um die Optimierung zu bewerten, müssen zwei Aspekte in Betracht gezogen werden. Einmal stellt sich die Frage, wie gut die Approximation das Verhalten des Systems wiedergibt und ob die gewünschten Eigenschaften durch die Optimierung erreicht wurden. Durch genaue Systemantworten kann die Approximation gut analysiert werden.

4.1 Analyse der Approximation

Das naheliegendste ist es sich die Residuen anzuschauen, um die Abweichungen von genauen Systemantworten zu untersuchen. In Abbildung 9 ist zu sehen. wie die berechneten Ergebnisse von den genauen Systemantworten abweichen. Hier ist zu erkennen, dass bei niedrigen Werten die Abweichungen prozentual größer werden. Die Optimierung einer Funktion, die wie bei den Verlusten eine Abweichung von 2000% aufweist, ist nicht zielführend. Dennoch ist zu sehen, dass bei größer werdenden Werten die Ergebnisse zuverlässiger werden. Durch Anpassen der Grenzwerte der Entwurfsvariablen können diese Bereiche ausgelassen werden. Ein Beispiel ist die Bautiefe (BT) die mit einem minimalen Wert von 50mm viel zu niedrig angesetzt ist für einen Motor mit 365kW Leistung. Da die Werte des Moments, der Spannung und der Verluste proportional zu der Bautiefe sind, würden diese mit ihr steigen.

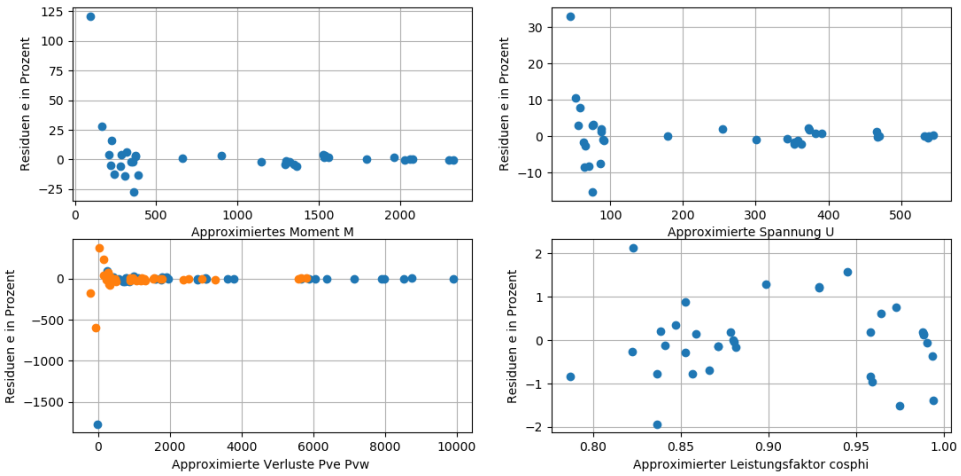

Abbildung 9: Residuen der einzelnen Ergebnisse in Prozent.

Um nicht die Grenzwerte anpassen zu müssen, können auch Terme deren Gewichtung sich auf einen irrelevanten Ergebnisbereich beziehen entfernt werden. Diese Terme verzerren die Approximation in einen ungewünschten Bereich und verschlechtern somit das Ergebnis. Die statistischen Methoden, die dafür verwendet werden, können in [Myers, Montgomery, & Anderson-Cook, 1995] gefunden werden.

Das adjustierte Bestimmtheitsmaß aus der Tabelle 5 zeigt, dass die Approximationen brauchbar erscheinen. Das ist der Tatsache geschuldet, das für jedes Ergebnis ein eignes Metamodell erzeugt wurde, um gegenseitige Verzerrungen zu verhindern. Dennoch kann es zu Problemen führen, wenn die Zielfunktion sensibel auf solche Ausreißer reagiert. Dann können die Optimierer in Minima Konvergieren, die nicht existieren.

R^2_{adj} von M	0.984
R^2_{adj} von Ueff	0.996
R^2_{adj} von Pvw	0.973
R^2_{adj} von Pve	0.916
R^2_{adj} von cosphi	0.924

Tabelle 5: Adjustiertes Bestimmtheitsmaß der jeweiligen RSA.

4.2 Beurteilung der Optimierung

Inwieweit die Optimierungen den Erwartungen gerecht wird, muss der Anwender für sich entscheiden. Die Funktionswerte der Optimierungsfunktion geben keine Auskunft darüber, wie erfolgreich die Optimierung war. Es ist möglich wie in Abbildung 8 verschiedene Ergebnisse zu vergleichen. Dennoch sind diese nur zielführend, wenn die Zielfunktion richtig formuliert wurde. Aus diesem Grund ist die direkte fachliche Beurteilung und eine eventuelle Anpassung der Zielfunktion nicht zu verhindern.

Das oberste Ergebnis aus Abbildung 8 wird im Folgenden auf erhoffte Merkmale untersucht. Um einen besseren Überblick über die hier vorliegende Geometrie zu erhalten, wird das Modell mit den Entwurfsparametern erzeugt.

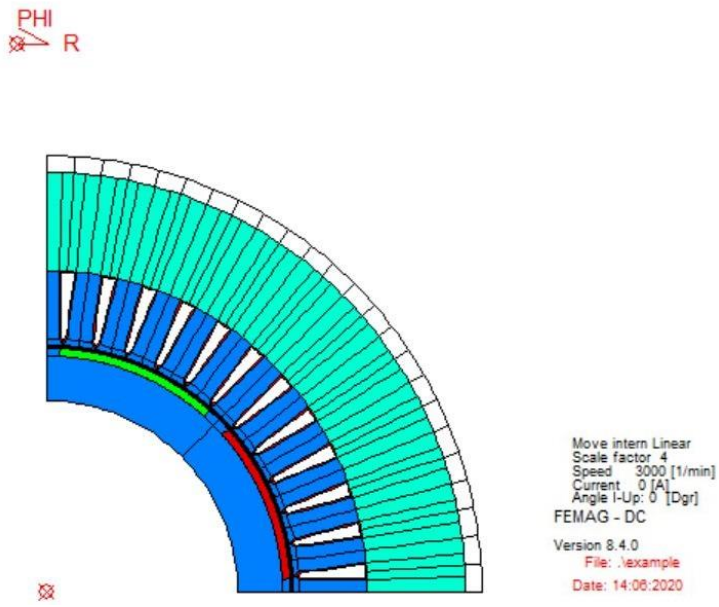

Abbildung 10: Modell mit optimalen Entwurfsvariablen (BT=50mm).

Auf den ersten Blick sind die Proportionen der Auslegung in einem guten Bereich. Wobei die Geometrie der Permanentmagneten ins Auge fällt. Deren Volumen wurde mit einem minimalen BT, RA und HM auf ein Minimum reduziert. Das bedeutet, das das Teilziel, das Volumen zu minimieren, erreicht wurde. Es ist jedoch zu sehen, das diesem Ziel eine höhere Priorität zugeordnet wurde als es gewünscht war. Denn die Funktionalität des Motors ist in dieser Bauform nicht mehr gewährleistet. Die Gewichtungsfaktoren der anderen Summanden der Gleichung (4.12) sind folglich zu klein gewählt worden. Um zu sehen, was das für Auswirkungen hat, sieht man, wenn man sich die Approximierten Ergebnisse dieser Entwurfsvariablen anschaut.

$\hat{y}_M$	$\hat{y}_{Ueff}$	$\hat{y}_{P_{VE}}$	$\hat{y}_{P_{VW}}$	$\hat{y}_{cosphi}$
$32.10Nm$	$38.95V$	$33.82W$	$598.51W$	0.77

Tabelle 6: Approximierte Ergebnisse der optimalen Auslegung.

Bei einem festen Strom I_{eff} von $455A$ je Phase, und einer Drehzahl n von $50s^{-1}$ werden für den Motor folgende Werte errechnet:

$$S = U_{eff} * I_{eff} * 3 \tag{4.1}$$

$$S = 38{,}95V * 455A * 3 = 53166{,}75VA$$

$$P_{auf} = cos\varphi * S \tag{4.2}$$

$$P_{auf} = 0{,}77 * 53166{,}75VA = 40938{,}39W$$

$$P_{ab1} = M * 2\pi * n \tag{4.3}$$

$$P_{ab1} = 32{,}10Nm * 2\pi * 50s^{-1} = 10084{,}51W$$

$$P_{ab2} = P_{auf} - P_{VE} - P_{VW} \tag{4.4}$$

$$P_{ab2} = 40938{,}39W - 33{,}82W - 598{,}51W = 40306{,}06W$$

Die nach der Zielfunktion gewünschte abgegebene Leistung P_{ab} liegt bei $365kW$, was nach Gleichung (4.5) und (4.6) nicht annähernd erreicht wird. Zusätzlich ergeben sich zwei verschiedene Ergebnisse für die abgegebene Leistung, wenn diese mit unterschiedlichen Ansätzen berechnet werden. Verursacht wird das durch die Residuen-Werte, die in den unteren Bereichen der Ergebnisse verstärkt vorkommen. Um einen genaueren Blick auf die Fehlerquelle zu erhalten, ist es notwendig, die Ergebnisse mit

der genauen Systemantwort zu vergleichen. In Tabelle 8 sind die genauen Systemantworten und die Approximierten gegenübergestellt und deren Residuen prozentual aufgetragen worden.

	y	$\hat{y}$	e
M	$214{,}26Nm$	$32.10Nm$	85%
U_{eff}	$61{,}02V$	$38.95V$	36%
P_{VE}	$478{,}79W$	$33.82W$	93%
P_{VW}	$505{,}54W$	$598.51W$	-18%
$cos\varphi$	$0{,}82$	0.77	6%

Tabelle 7: Genaue Systemantwort und approximierte Systemantwort im Vergleich.

Es wird deutlich, dass die in der Analyse der Approximation erkannten Probleme, hier wieder auftreten. Durch die großen Abweichungen kann kein sinnvolles Ergebnis erzeugt werden. Somit werden in die Optimierungsfunktionswerte eingelesen, die nicht existieren. Was dazu führt, dass die Optimierung in ein falsches Minimum konvergiert.

5 Fazit

Die Optimierung mit der Response Surface Methode ermöglicht bei rechenaufwendigen Systemen einen Überblick über den gesamten Designraum. Die Qualität dieser Betrachtung kann jedoch stark variieren. Es ist vorteilhaft den untersuchten Designraum so groß wie nötig zu definieren, damit das Metamodell Rückschlüsse auf das System zulässt. Mit steigender Rechenleistung können in derselben Zeitspanne mehr Datenpunkte berechnet werden. Das bedeutet, dass die RSM erweitert werden kann. Es ist möglich die quadratische Ansatzfunktion mit komplexeren Funktionen zu erweitern. Stehen genügend Datenpunkte zu Verfügung, können die Ansatzfunktionen durch neuronale Netze ersetzt werden, welche eine spezifischere Anpassung ermöglichen.

Die Optimierung multidisziplinärer Probleme mit mehreren Zielen erweist sich bei der Definition der Zielfunktion als problematisch. Die mathematische Formulierung der einzelnen Gewichtungen sollten schrittweise erfolgen. So kann die Übersicht über das Verhalten der Optimierung besser bewahrt werden. Entwurfsvariablen die durch eine komplexe Zielfunktion bewertet wurden, können nicht der gesuchten Entwurfsvariablen entsprechen. Das ist der Tatsache geschuldet, dass die definierte Zielfunktion nur eine Annäherung des Ziels repräsentiert. Bessere Ergebnisse können eindeutigere Ziele liefern, wie das Maximieren oder Minimieren eines bestimmten Wertes, wobei das Ziel unumstritten ist.

Durch leistungsstärkere Computer und größere Datenmengen, wird die rechnergestützte Entscheidungsfindung weiter Einzug in das Leben der Menschen finden. Diese Entscheidungen werden immer mehr von einem empfehlenden in einen ausführenden Charakter übergehen. Wie weit das geht, hängt davon ab, wie lange das der optimale Weg für die Menschheit bleibt.

Anhang: Code

Der Quellcode ist in der Datei „PA7_Kander_Akinci_Code“ zu finden.

Abbildungsverzeichnis

Tabellenverzeichnis

Codeverzeichnis

Literaturverzeichnis

Dr. Waschatz, U. (2003). *Statische Versuchsplanung- zuverlässiger und schneller zu Ergebnissen.*

Fedorov, V. V. (1972). *Theory of Optimal Experiments (Review).* Translated and edited by W. J. Studden and E. M. Klimko.

Gleichmar, R. (2004). *Approximationen und paralleles Rechnen bei der multidisziplin¨aren Strukturoptimierung.* München: Fakultät für Maschinenwesen der Technischen Universität München.

Markos, P., Marion, L., & Martin, B. (2015). *Optimierung: Statische, dynamische, stochastische Verfahren für die Anwendung .* Berlin Heidelberg: Springer-Verlag GmbH.

Myers, R. H., Montgomery, D. C., & Anderson-Cook, C. M. (1995). *Response Surface Methodology: Process and Product Optimization Using Desinged Experiments.* New York: John Wiley & Sons, Inc., Hoboken, New Jersey.

Reichert, P. D.-I. (2008). *FEMAG.* CH-8092 Zuerich: Institute of Electrical Machines ETH Zentrum.

Triefenbach, F. (2008). *Design of Experiments: The D-Optimal Approach and Its Implementation As a Computer Algorithm.* SE-901 87 UME˚ A SWEDEN: Umea University Department of Computing Science.

Eigenständigkeitserklärung

Hiermit bestätige ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen Publikationen, Vorlagen und Hilfsmitteln als die angegebenen benutzt habe. Alle Teile meiner Arbeit, die wortwörtlich oder dem Sinn nach anderen Werken entnommen sind, wurden unter Angabe der Quelle kenntlich gemacht. Gleiches gilt für von mir verwendete Internetquellen. Ich versichere, dass ich diese Arbeit oder nicht zitierte Teile daraus vorher nicht in einem anderen Prüfungsverfahren eingereicht habe. Mir ist bekannt, dass meine Arbeit zum Zwecke eines Plagiatsabgleichs mittels einer Plagiatserkennungssoftware auf eine ungekennzeichnete Übernahme von fremdem geistigen Eigentum überprüft werden kann. Ich versichere, dass die elektronische Form meiner Arbeit mit der gedruckten Version identisch ist.

Köln, 16.06.2020 

Ort, Datum Unterschrift