Optimierung und Robustheitsbewertung mit OptiSLang bei der Robert Bosch GmbH

Markus Könning[1*], Henning Kreschel[1],
Roland Schirrmacher[1], Andreas Plotzitza[2]

[1] Robert Bosch GmbH, FV/PLE4, Schwieberdingen

[2] Robert Bosch GmbH, PT-BEU/ECC1, Leinfelden-Echterdingen

Zusammenfassung

Bohrhämmer mit einer ausgeprägten Bohr-Performance überzeugen den Kunden bei der Kaufentscheidung. Somit steht in der Hammerentwicklung die Auslegung der Schlagkette, bestehend aus Schläger, Döpper und Werkzeug, an oberster Stelle. Mithilfe der hier vorgestellten Optimierungsmethode ist erstmals eine dynamische Auslegung der Schlagkette inklusive eines Betonmodells, das die durch den Stoßprozess erzeugte unterschiedliche Belastungsgeschwindigkeit im Beton realitätsnah abbildet, möglich.

Weiterhin wird gezeigt, dass das gefundene Optimum gegenüber Schwankungen, wie zum Beispiel Fertigungstoleranzen, robust ist. Im Vortrag wird eine Optimierung und Robustheitsanalyse eines Schlagwerkes mit der Software OptiSLang gezeigt.

Keywords: Optimierung, Robustheit, Fertigungstoleranz, Bohrhammer

[*] Kontakt: Markus Könning, Robert Bosch GmbH, FV/PLE4, Postfach 300240, 70442 Stuttgart, E-Mail: Markus.Koenning@de.bosch.com

1 Problembeschreibung

1.1 Ziel der Optimierung

Im Rahmen der Bemühungen, die bestmöglichste Abtragsleistung eines Bohrhammers zu erzielen, liegt es nahe die Schlagkette so zu optimieren, dass die Eindringung des Meißels in den Beton maximiert wird. In **Abbildung 1** ist das Problem als Skizze dargestellt.

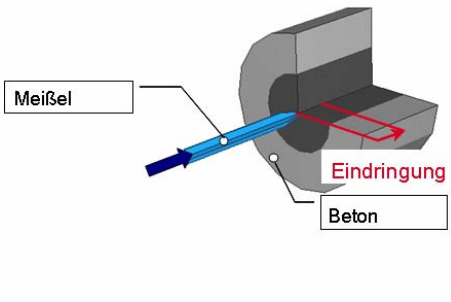

Abbildung 1: Modelldarstellung.

1.2 Bohrhammer im Aufbau

In **Abbildung 2** ist ein detaillierter Aufbau des Bohrhammers dargestellt. Für die Optimierung wurde ein FEM-Modell bestehend aus Schlagkette (Schläger-Döpper-Werkzeug) und Betonmodell aufgebaut.

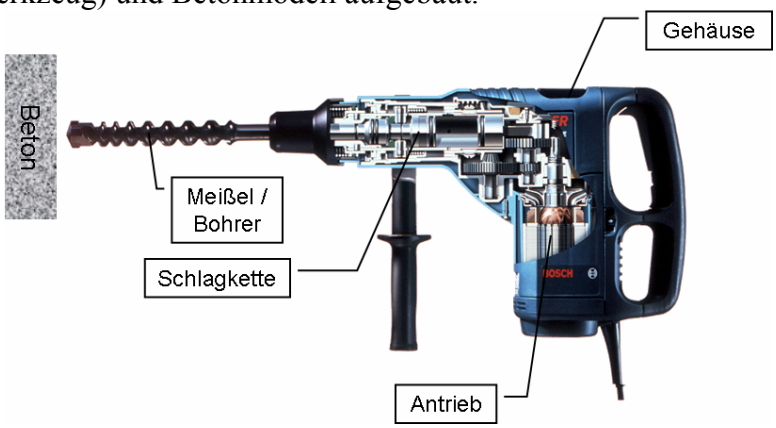

Abbildung 2: Detailaufbau beim Bohrhammer.

Aus der Schlagkette lassen sich direkt die Eingangsparameter für die Optimierung ableiten. Es handelt sich hierbei um 4 kontinuierliche und 2 diskrete Parameter. Insgesamt werden 32 Ergebnisgrößen definiert, wobei 2 Parameter die Restriktionen beschreiben, 1 Parameter die Zielfunktion und die restlichen Parameter zum Systemverständnis mit angezeigt werden.

2 Durchführung

2.1 Modellaufbau

Für das vorhandene FEM-Modell ist es notwendig, dass alle Eingangsgrößen im Modell parametrisiert sind. Bei dem FEM-Paket ABAQUS bietet sich hierbei die Parametrisierung mittels der Programmiersprache Python an, da Python-Skripte in ABAQUS direkt ablaufen können. In **Abbildung 3** ist die Routine skizziert, die bei einer Berechnung eines erzeugten Designs durchlaufen werden muss. Demnach werden zuerst Eingangsparameter in ASCII-Format in *parametrisierung.py* definiert. Anschließend wird daraus ein Input-File *analyse.inp* erzeugt, was von ABAQUS verarbeitet wird. Nach der erfolgreichen Berechnung gibt ABAQUS ein Ausgabefile(ODB-File) aus, welches mittels einer weiteren Python-Routine *resultate.py* so verarbeitet wird, dass OptiSLang die benötigten Ausgabeparameter im ASCII-Format aus *simulation.erg* erhält.

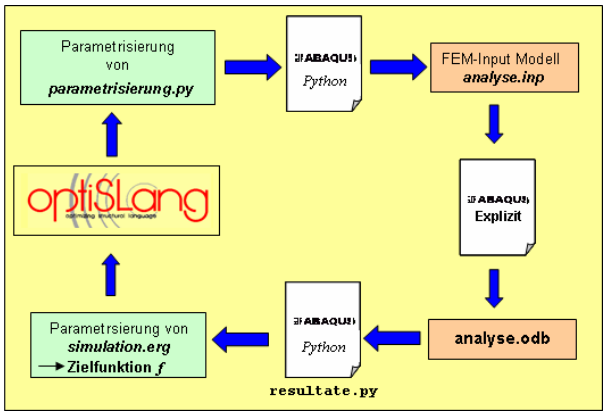

Abbildung 3: Routine zur Jobverarbeitung.

2.2 Auswahl der Optimierungsmethode

In dem Programm OptiSLang stehen für die Optimierung 3 verschiedene Methoden zur Verfügung. Zum einen wird der NLPQL-Algorithmus (Nonlinear Programming Quadratic Linesearch) von Prof. Schittkowski angeboten. Hierbei handelt es sich um einen Algorithmus aus dem Bereich der Gradientenverfahren, die ihre Schwächen bei der Optimierung von Gebieten mit mehreren lokalen Optima haben. Zum anderen wird eine Optimierungsmethode auf Basis von Antwortflächen angeboten, die so genannte Adaptive Response Surface Method (ARSM). Hier wird auf der Antwortfläche einer Approximation der ursprünglichen Zielfunktion optimiert. Des Weiteren wird noch ein Evolutionärer Algorithmus angeboten, der seine Stärken bei der globalen Optimierung von Gebieten mit vielen lokalen Optima hat.
Zur Optimierung des Bohrhammers wurde der Evolutionäre Algorithmus gewählt, da das Lösungsgebiet mit großer Wahrscheinlichkeit mehrere lokale Optima hat. Die Einstellungen bei diesem sind wie folgt:

- Population: 8 (parallel berechnet)
- Generation: 14
- (weitere Einstellungen sind Grundeinstellungen)

3 Ergebnisse der Optimierung

Nach der Rechenzeit von sieben Tagen wurde eine Verbesserung der Eindringtiefe von 25 % zu einem gegebenen Referenzmodell erzielt. Das Ergebnis ist in **Abbildung 4** zu sehen.

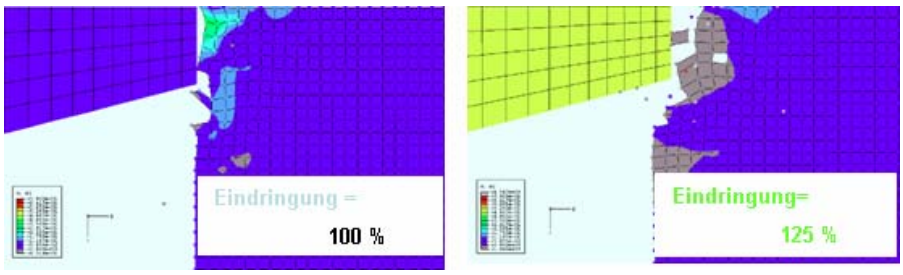

Abbildung 4: Eindringung bei Referenzmodell (links) und dem gefundenen Optimum(rechts)

Die berechneten Eindringtiefen des Meißels in den Beton sind anschließend in einem praxisnahen Versuch bestätigt worden

4 Robustheitsanalyse

Im Anschluss an die Optimierung wurde noch eine Robustheitsanalyse des optimierten Bauteils durchgeführt. Dafür wurden die Toleranzangaben aufbereitet. Für die Robustheitsanalyse wurde das Latin Hypercube Sampling mit 72 Samplingpunkten benutzt. Als Ergebnis erhält man eine Korrelationsmatrix (siehe **Abbildung 5**).

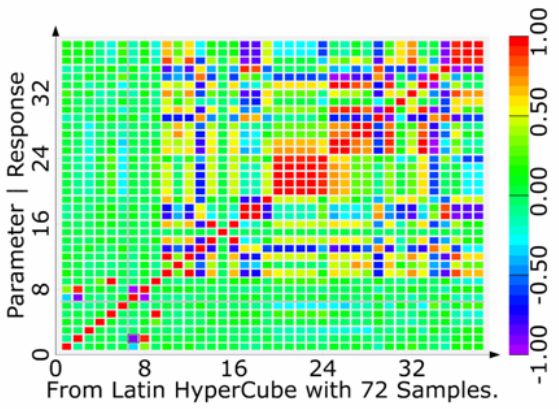

Abbildung 5: resultierende Korrelationsmatrix

Dadurch, dass die Fertigungstoleranzen sehr gering sind, wirken sich diese Toleranzen nicht negativ auf die Ergebnisgröße für die Eindringtiefe aus. Dies ist in **Abbildung 6** dargestellt.

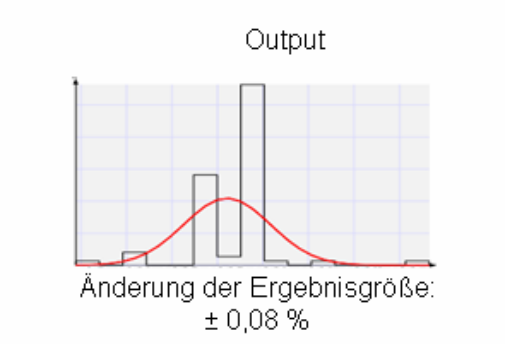

Abbildung 6: Verteilung der Eindringtiefe

5 Zusammenfassung und Ausblick

Im vorliegenden Beispiel konnte die Optimierungsmethoden erfolgreich eingesetzt werden. Die numerischen Ergebnisse wurden durch praxisnahe Versuche bestätigt. Durch die Robustheitsanalyse konnten die Parameter, die einen großen Einfluss auf die Streuung der Zielfunktion haben, identifiziert werden. Die vorgegebene Streuung der Eingangsgrößen bewirkt lediglich eine geringe Streuung des Wertes der Zielfunktion.

Es wäre wünschenswert, wenn in naher Zukunft im Programm OptiSLang neben dem Evolutionären Algorithmus auch die ARSM gemischte Parameter verarbeiten könnte. Dann gäbe es eine Alternative, die in diesem Dimensionsraum auch noch recht erfolgreich sein dürfte. Ebenso ist es sinnvoll in Zukunft die Optimierung mit der Robustheitsanalyse zu koppeln, damit nicht unnützerweise mehrmals eine Robustheit für mehrere „Optima“ durchgeführt werden muss. Eine direkte Kopplung zu ABAQUS, d.h. ein direktes Einlesen des ODB-Files wäre ebenfalls von Vorteil, da der Umweg über die Pythonskripte sehr mühsam ist. Im Monitoring für die Stochastik (Robustheit) wäre es sinnvoll, einen prozentualen Anteil der Beitragsleister an den Ergebnisgrößen anzuzeigen.

Literatur

ABAQUS: User's Manual. Abaqus Inc. Pawtuc USA, 2003.

GOLDBERG, D. E.: Gentic Algortihms in Search, Optimization, and Machine Learning. Reading, 1989.

HIMSTEDT, T.; MÄTZEL, K.: Mit Python programmieren. Heidelberg, 1999.

OPTISLANG: User's Manual. Version 2.0. März 2004

PLOTZITZA, A. ; EIBL, J.: A Technique for a Numerical Simulation of Concrete Slabs for Demolishing by Blasting. Structural Dynamics, Eurodyn 2002, Grundmann & Schueller (eds.), 2002, pp. 1457-1467.

SCHITTKOWSKI K.: Scaling NLPQL: A FORTRAN subroutine solving constrained nonlinear programming problems. Universität Stuttgart, 1985.