Optimierung mit dem Programmmodul OPTIMA

Dr.-Ing. Martin Siffling, Dr.-Ing. Casimir Katz, SOFiSTiK AG; Fabian Gerold M.Eng.,Uni Weimar

Zusammenfassung:
Mit dem neuen Programm OPTIMA stellt die SOFiSTiK AG ein Tool zur Lösung von Optimierungsaufgaben zur Verfügung. Im nachfolgenden Beitrag wird eine kurze Einführung in das Programm gegeben. Dabei werden kurz die Grundlagen der Optimierung skizziert und die in OPTIMA implementierten Verfahren beschrieben. Die praktische Anwendung des Programms wird an einfachen kleinen Beispielen erläutert. Für weiterführende Informationen wird auf die Literatur verwiesen.

Summary:
The latest development of the SOFiSTiK AG is a new program called OPTIMA to solve optimisation problems. The following paper is a short introduction in the usage of this program, describing the basic optimisation theory, the implemented optimisation algorithms and showing the practical use on simple examples. For further information we refer to the technical literature.

1 GRUNDLAGEN UND BEGRIFFE DER OPTIMIERUNG

Ziel einer Optimierung ist es, eine (skalare) **Zielfunktion** z(**x**) zu minimieren oder zu maximieren. Im Zusammenhang mit einer FE-Berechnung können beliebige Ergebniswerte oder Kombinationen davon als Zielfunktion definiert werden, im letzteren Fall spricht man von **Mehrzieloptimierung**. Zielfunktion kann z.B. das Gesamtgewicht, die 1. Eigenfrequenz eines Systems, oder die Kosten eines Gebäudes sein.
Eine Maximierungsaufgabe lässt sich durch den Zusammenhang min{ z(**x**) } = max{ -z(**x**) } in eine Minimierungsaufgabe überführen. Daher wird im Folgenden ohne Beschränkung der Allgemeinheit nur noch von Minimierung gesprochen.
Voraussetzung für eine Optimierung ist, dass **Optimierungsvariablen x** im System enthalten sind, die einen Einfluss auf die Zielfunktion haben. Der Berechnungsaufwand steigt überproportional mit der Anzahl n der Optimierungsvariablen (x_1, x_2, …, x_n) = **x** an. Daher sollte ihre Zahl so gering wie möglich gehalten werden.

Die Optimierungsvariablen besitzen Schranken $x_{min} \leq x \leq x_{max}$. Innerhalb davon darf der Optimierungsalgorithmus die Variablen verändern.
Bei praktischen Anwendungen treten üblicherweise **Nebenbedingungen** auf, d.h. es müssen bestimmte Bedingungen erfüllt sein wie z.B. Spannungs- oder Verformungsbeschränkungen.

Damit stellt sich die Optimierungsaufgabe folgendermaßen dar:
min z(**x**)
so dass
g(**x**) = 0 Gleichungsnebenbedingungen
h(**x**) < 0 Ungleichungsnebenbedingungen

Das Problem mit den Nebenbedingungen ist, dass Ergebnisse wie Spannungen oder Verformungen erst nach einer Berechnung des Systems vorliegen und somit erst hinterher festgestellt werden kann, ob die Berechnung überhaupt zulässig war. Damit die Berechnung nicht völlig verworfen werden muss, gibt es unterschiedliche Ansätze zur Behandlung von (verletzten) Nebenbedingungen. Eine Methode ist die Verwendung einer Straffunktion. Dabei wird ein bestimmter Betrag auf die Zielfunktion addiert, um zu erreichen, dass der Optimierungsalgorithmus diesen unzulässigen Bereich meidet.

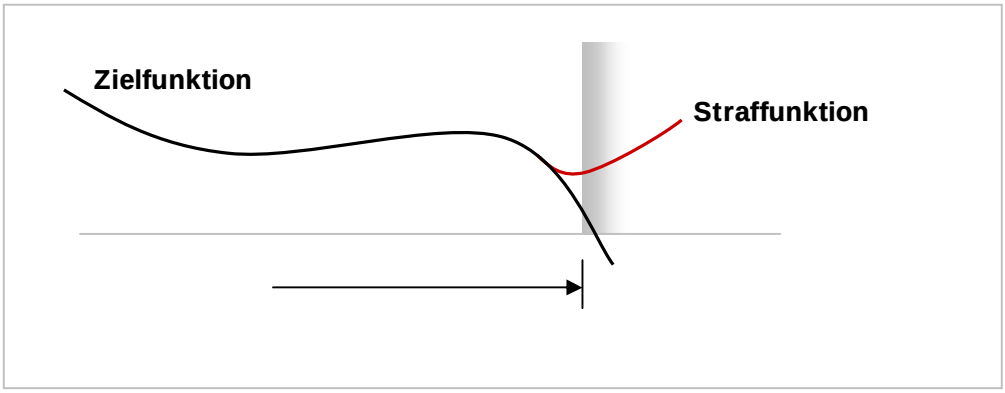


Abbildung 1: Kombination aus innerer und äußerer Straffunktion

Lokale und globale Optimierung

Häufig haben Optimierungsprobleme nicht nur ein einziges Minimum, sondern mehrere **lokale** Minima und ein **globales** Minimum mit dem niedrigsten Zielfunktionswert des gesamten zulässigen Suchraumes. Ein *lokales* Suchverfahren folgt vom Startpunkt aus immer der Abstiegsrichtung, und gelangt so zwangsläufig zu einem Minimum, ohne jedoch ausschließen zu können dass es noch weitere Minima mit niedrigerem Zielfunktionswert gibt. Ein lokales Suchverfahren verhält sich wie

Wasser, dass in einer Hügellandschaft abwärts fließt und in einer Mulde bleibt, ohne den Rand der Mulde überwinden und weiter talwärts fließen zu können.
Optimierungsergebnisse sind aber meistens gut zu verifizieren indem man den Zustand vorher und nachher vergleicht und entscheidet, ob die erreichte Verbesserung ausreicht. Falls dies nicht der Fall ist, kann die Optimierung wiederholt werden, ggf. mit einem anderen Startpunkt.

Formoptimierung

Die Formoptimierung als Optimierungsaufgabe kann

1. über CAD Parameter, oder
2. über FE-Parameter

beschrieben werden.

Bei der Verwendung von FE-Parametern werden beispielsweise Ausnutzungsgrade, Spannungen, Querschnittsabmessung, Verformungen, etc. optimiert. Im Gegensatz dazu werden bei den CAD-Parametern z.B. Knotenkoordinaten, Anzahl von Elementen, etc. optimiert.
Im nachfolgenden Beispiel (Abbildung 2) werden nur die z- Koordinaten der oberen Knoten als Variablen definiert. Die unteren Knoten sind fest.

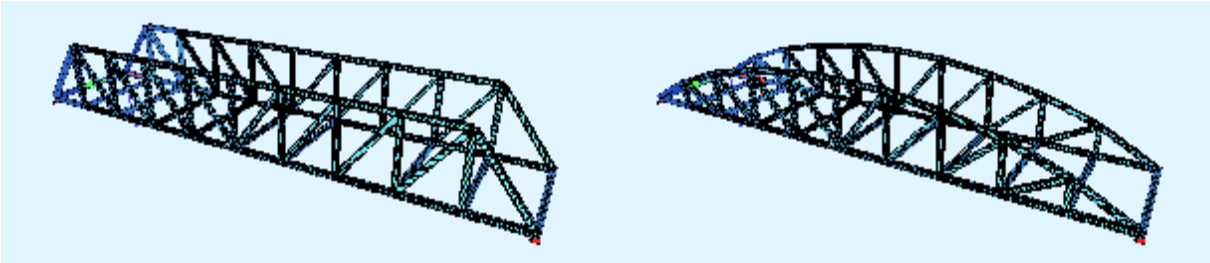


Abbildung 2: Formoptimierung eines Fachwerks

Topologieoptimierung

Hierbei handelt es sich um eine Methode die im frühen Entwicklungsstadium eingesetzt werden kann um den Materialverbrauch zu minimieren und die optimale „Topologie“ eines Bauteils zu finden. Gemeint ist damit, dass von einem vorgegebenen Konstruktionsraum gering beanspruchtes Material entfernt wird, bis nur noch das tatsächlich notwendige Material übrig bleibt.

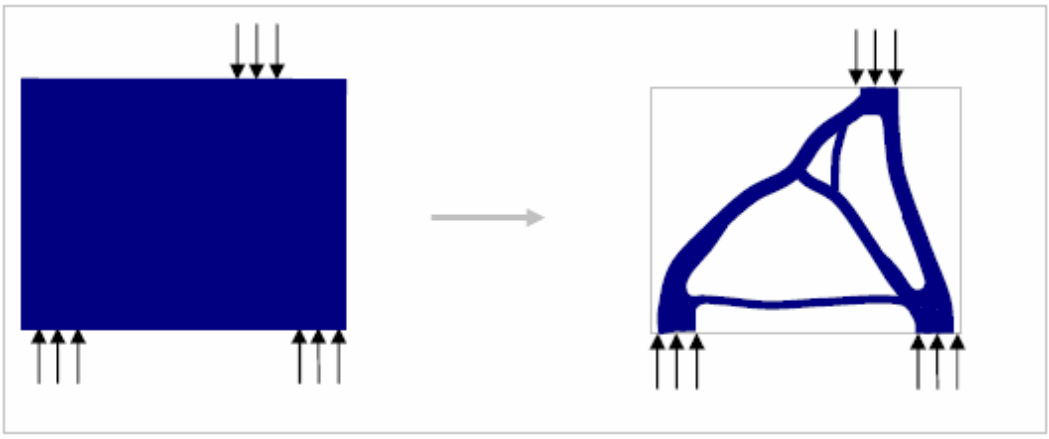


Abbildung 3: Topologieoptimierung

Meistens ist eine manuelle Nachkonstruktion und Glättung der Struktur notwendig. Die Topologieoptimierung kann wie die Formoptimierung am 2D- oder 3D- Modell erfolgen. Dieses Verfahren wird hauptsächlich in der Automobil- und Luftfahrtindustrie zur Querschnitss- und/oder Materialoptimierung angewendet und ist ein fester Bestandteil des Entwicklungsprozesses geworden.

Optimierung mit Excel

In Microsoft Excel ist ein Programm integriert (Menü „Extras“- „Add-Ins“ - „Solver“), mit dem lokale Optimierungen vorgenommen werden können. Es können alle in Excel verfügbaren Funktionen verwendet werden, um eine Zielfunktion zu formulieren. Auch Nebenbedingungen können berücksichtigt werden.

Abbildung 4: Eingabefenster des „Solver"

2 OPTIMIERUNGSVERFAHREN

Es gibt kein Optimierungsverfahren das immer die beste Lösung mit dem geringsten Aufwand erreicht. Es gibt viele Verfahren, die sich stark unterscheiden in der Art der Lösungssuche und der Art der Probleme, für die sie jeweils am besten geeignet sind.
Optima bietet mehrere Verfahren an, deren Funktionsweise und Anwendung im Folgenden näher erläutert werden soll.

2.1 Polytop Algorithmus

Für den Polytop (Vielflach) Algorithmus sind zunächst n+1 Punkte im Suchraum erforderlich. Diese Punkte werden nach ihrem Funktionswert $F_{n+1} \geq F_n \geq ... \geq F_2 \geq F_1$ geordnet und bilden so ein Polytop.

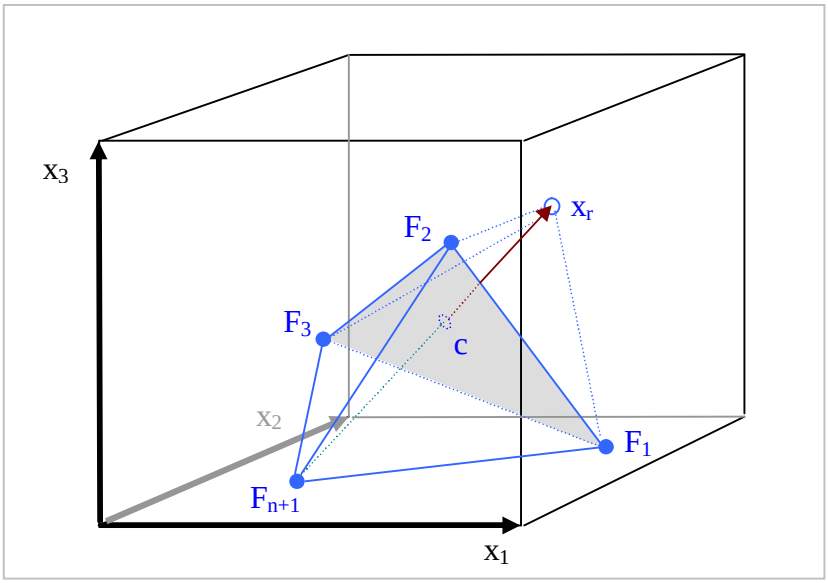


Abbildung 5: Polytop bei n=3

Anschließend wird aus den n besten Punkten der **Schwerpunkt c** berechnet und vom schlechtesten Punkt (F_{n+1}) über den Schwerpunkt der sogenannte **Reflektionspunkt** berechnet $\mathbf{x}_r = \mathbf{c} + \alpha \cdot (\mathbf{c} - \mathbf{x}_{n+1})$. Der Koeffizient α bestimmt die Entfernung des Reflektionspunktes vom Schwerpunkt, bei $\alpha = 1$ ist der Reflektionspunkt genauso weit vom Schwerpunkt entfernt wie der schlechteste Punkt. Je nach Zielfunktionswert des Reflektionspunktes sind 3 Fälle zu unterscheiden:

1. Ist der neu gewonnene Reflektionspunkt weder ein neuer bester Punkt noch ein neuer schlechtester Punkt des Polytops ($F_1 \leq F_r \leq F_n$), so ersetzt der Reflektionspunkt $\mathbf{x}_r$ den schlechtesten Punkt $\mathbf{x}_{n+1}$ und die nächste Iteration wird begonnen.

2. Ist der Reflektionspunkt besser als der beste Punkt F_1, so wird die Richtung der Reflektion mit dem **Expansionspunkt** $\mathbf{x}_e = \mathbf{c} + \beta \cdot (\mathbf{x}_r - \mathbf{c})$ fortgesetzt. Ist die Expansion erfolgreich ($F_e < F_r$), so ersetzt $\mathbf{x}_e$ den schlechtesten Punkt $\mathbf{x}_{n+1}$, andernfalls wird $\mathbf{x}_{n+1}$ durch $\mathbf{x}_r$ ersetzt.
3. Wenn der Reflektionspunkt ein neuer schlechtester Punkt des Polytops ist ($F_r > F_n$), so wird angenommen, dass der Polytop zu groß ist und ein Kontraktionspunkt x_c wird erzeugt:

   $\mathbf{x}_c = \mathbf{x}_1 + \gamma \cdot (\mathbf{x}_{n+1} - \mathbf{x}_1)$ wenn $F_r \geq F_{n+1}$

   $\mathbf{x}_c = \mathbf{x}_1 + \gamma \cdot (\mathbf{x}_r - \mathbf{x}_1)$ wenn $F_r < F_{n+1}$

   Wenn $F_c < \min\{ F_r, F_{n+1} \}$, war die Kontraktion erfolgreich und $\mathbf{x}_c$ ersetzt $\mathbf{x}_{n+1}$. Andernfalls wird eine zweite Kontraktion durchgeführt.

Daraus folgt, dass für eine Iteration 1 bis 3 FE-Berechnungen erforderlich sind (Reflektionspunkt, ggf. Expansionspunkt oder bis zu 2 Kontraktionspunkte).

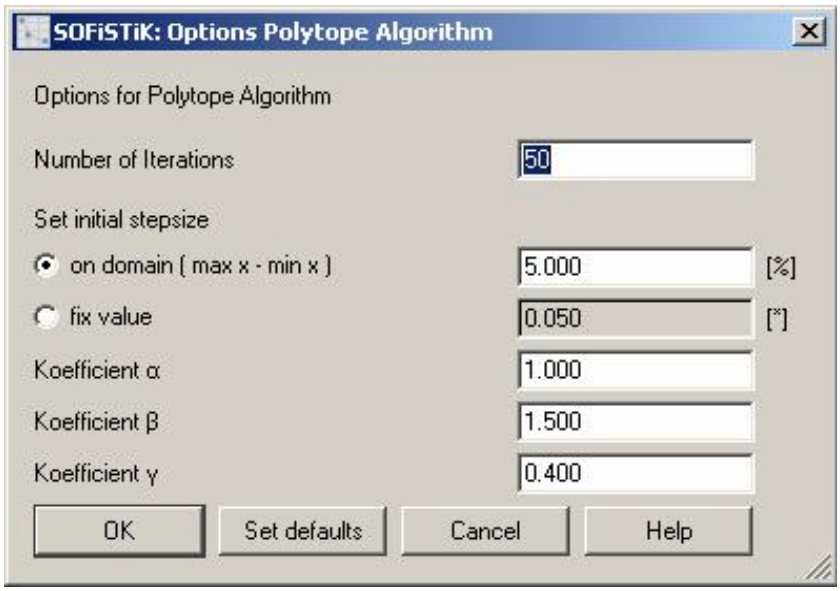


Abbildung 6: Optionen des Polytop Algorithmus

Ein Vorteil des Polytop Verfahrens ist, dass zur Richtungsbestimmung keine Funktionswertberechnungen erforderlich sind, wie beispielsweise beim Gradienten- oder Quasi-Newton Verfahren. Stattdessen wird die Richtung aufgrund der vorhandenen Punkte „vermutet“.

Während der Optimierung zeigt Optima ein Diagramm mit dem Optimierungsfortschritt an. Auf der x-Achse ist die Anzahl an FE-Berechnungen aufgetragen, auf der y-Achse der zugehörige Zielfunktionswert, dividiert durch den Zielfunktionswert des Startpunktes. Zusätzlich wird unter dem Fortschrittsdiagramm angezeigt, welcher Punkt gerade berechnet wird.

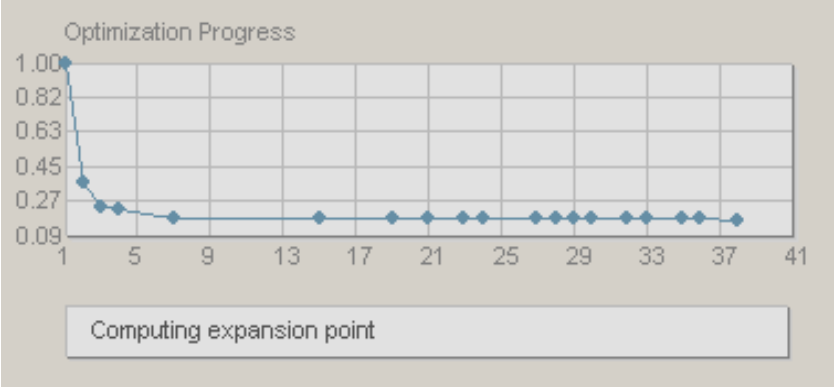


Abbildung 7: Diagramm des Optimierungsfortschritts

Nebenbedingungen werden hier nicht explizit angegeben, sondern müssen direkt in die Zielfunktion mit eingearbeitet werden.

2.2 Evolutionsstrategien

Evolutionsstrategien gehören zum Bereich der Bionik, der wissenschaftlichen und technischen Nutzung von Prinzipien aus der Natur. Das Darwin'sche Prinzip „Survival of the fittest" wird in diesem Verfahren angewendet, daher der Name „Evolutionsstrategie".
Evolution in der Natur basiert auf Rekombination, Mutation und Selektion und diese Prinzipien können auch zur Optimierung technischer Systeme verwendet werden. Ein Vektor von Optimierungsvariablen **x** mit zugehörigem Zielfunktionswert soll im Folgenden als ein Individuum bezeichnet werden. Eine Menge von λ Individuen stellt eine Population dar. Ausgehend von einem Elternindividuum werden nun λ Nachkommen einer Population erzeugt, indem zu den (Eltern-) Variablen **x** Mutationsanteile addiert werden. Es wird von allen Individuen der Population der Zielfunktionswert bestimmt und der beste wird selektiert als Elternindividuum der nächsten Generation. Zur Erzeugung von Individuen ergibt sich zunächst folgende Vorschrift:

$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{r}_k$ mit $\mathbf{r}_k$ als einem Vektor von normalverteilten Zufallszahlen.

Die Standardabweichung der Normalverteilung ist ein Maß für die Schrittweite:

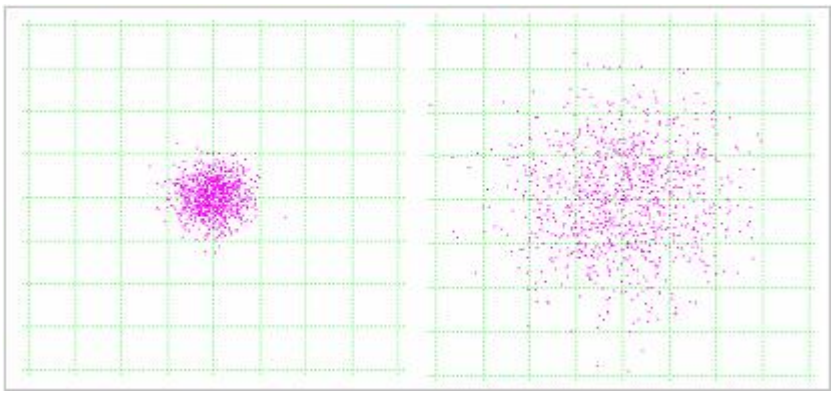


Abbildung 8: Kleine und große Standardabweichung

Das Streuungsmaß ist entscheidend für die Konvergenzgeschwindigkeit; eine zu kleine Schrittweite führt zu einem zwar relativ konstanten, aber langsamen Optimierungsfortschritt. Zu große Schrittweiten können unter Umständen nur zu Verschlechterungen führen. Dieser Zusammenhang zwischen Fortschrittsgeschwindigkeit und der Schrittweite wurde von Rechenberg [3] untersucht mit dem Ergebnis einer adaptiven Schrittweitensteuerung:

> *„Ist während der Optimierung der Quotient q der erfolgreichen Mutationen zur Gesamtzahl der Mutationen kleiner als 1/5, so wird die Schrittweite verkleinert, ist der Quotient dagegen größer als 1/5, so ist die Schrittweite zu vergrößern.“*

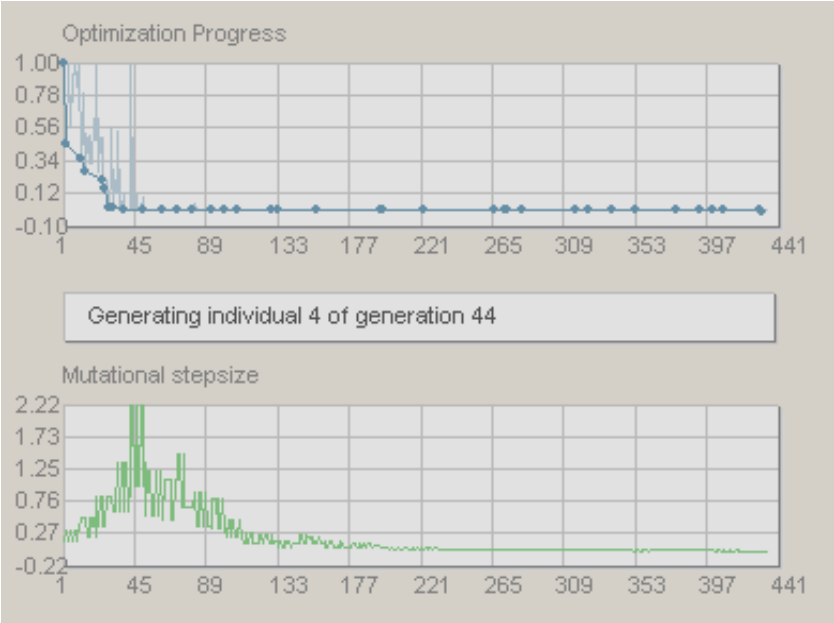


Abbildung 9: Anpassung der Schrittweite durch Optima

Rekombination kann dadurch erreicht werden, dass zwei oder mehrere Individuen einer Population selektiert und ihre Optimierungsvariablen **x** diskret oder intermediär „verschmolzen“ werden:

Diskrete Rekombination

1.3	3.6	⇒	1.3
2.5	2.0		2.0
0.1	4.2		4.2
8.5	7.5		8.5
4.4	5.3		5.3
Elternvektoren			Nachkommenvektor

Intermediäre Rekombination

1.3	3.6	⇒	2.45
2.5	2.0		2.25
0.1	4.2		2.15
8.5	7.5		8.0
4.4	5.3		4.85
Elternvektoren			Nachkommenvektor

Evolutionsstrategien zeichnen sich durch eine hohe Anwendungsbreite vor allem bei Problemen mit ganzzahligen und reellen Variablen.
Ein großer Vorteil von Evolutionsstrategien ist, dass sie lokale Minima überwinden können, was bei ableitungsgestützten Verfahren nicht der Fall ist.
Weiterhin gefährden eine schlecht gewählte Konfiguration und schlechte Startwerte der Optimierungsvariablen normalerweise nicht die gesamte Optimierung, sondern verlängern nur die Rechenzeit. Deshalb werden Evolutionsstrategien oft als „robust“ bezeichnet.

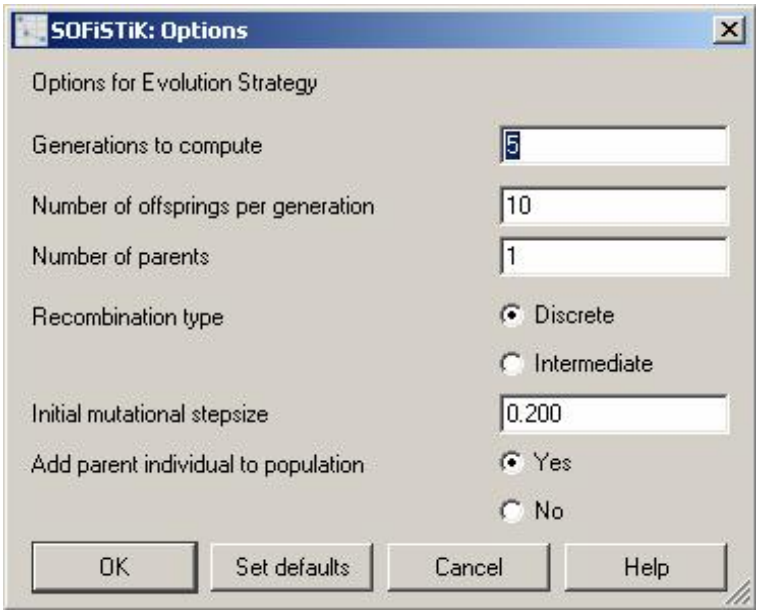


Abbildung 10: Optionen der Evolutionsstrategie

In den Optionen zur Evolutionsstrategie lassen sich die Anzahl der Generationen und die Anzahl an Individuen pro Generation einstellen. Mit der Anzahl der Individuen pro Generation steigt auch der Berechnungsaufwand.
Die Optimierungsberechnung wird über folgende Parameter gesteuert:

- Generations to compute = Anzahl der zu berechnenden Generationen
- Number of offsprings per generation = Anzahl der Individuen pro Generation
- Number of parents = Anzahl aus wie vielen Elterngenerationen ein Nachkomme erzeugt wird
- Recombination type = Art der Rekombination
- Initial mutational stepsize = Startschrittweite für die Mutation
- Add parent individual to population = hier wird das Elternindividuum (das beste Individuum der vorhergehenden Generation) der Nachkommenpopulation hinzugefügt. Dies verhindert einen Evolutionsrückschritt, falls alle Individuen einer Generation keine Verbesserung gebracht haben.

Nebenbedingungen werden über eine Straffunktion und Addition der Strafterme auf die Zielfunktion berücksichtigt.

2.3 Achsenparallele Suche

Die Achsenparallele Suche, auch als „Gauß-Seidel Verfahren“, oder „Koordinatenverfahren“ bezeichnet, löst ein mehrdimensionales Optimierungsproblem durch aufeinander folgende eindimensionale Optimierungen. Eine Iteration umfasst eine (achsenparallele) Liniensuche für jede Optimierungsvariable.

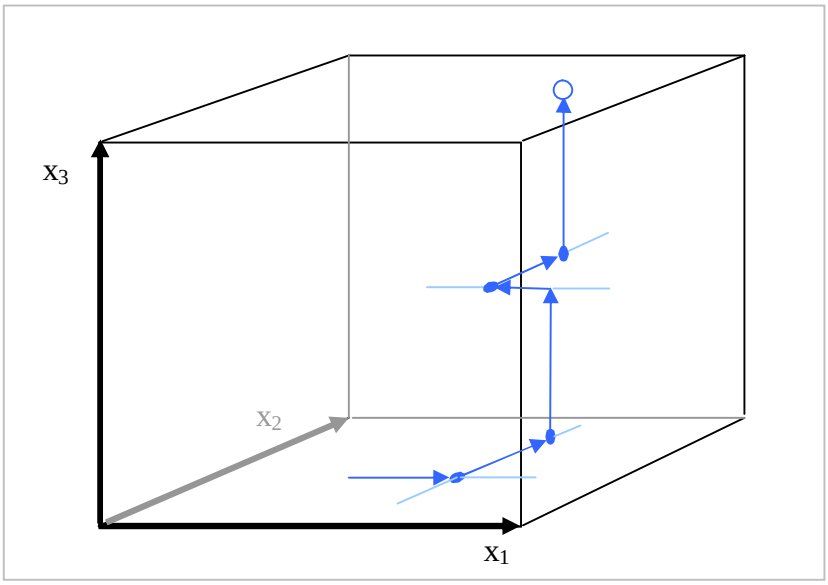


Abbildung 11: Suchablauf bei der Achsenparallelen Suche

Die Reihenfolge der Variablen spielt eine wesentliche Rolle bei diesem Verfahren.

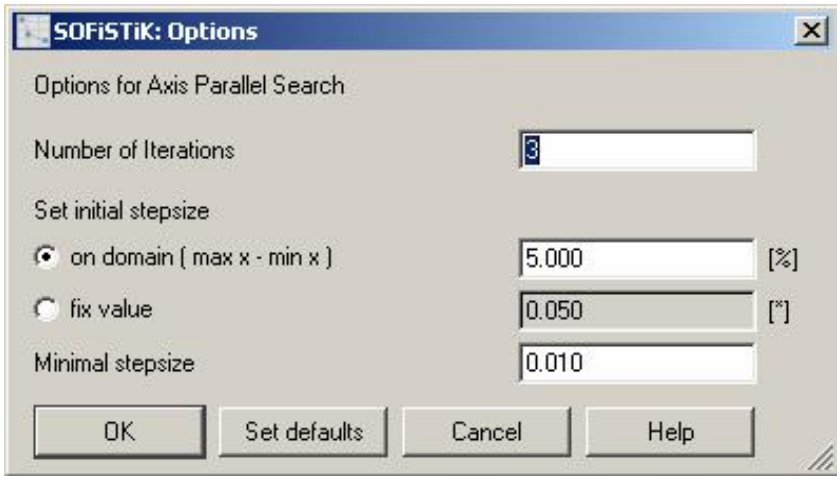
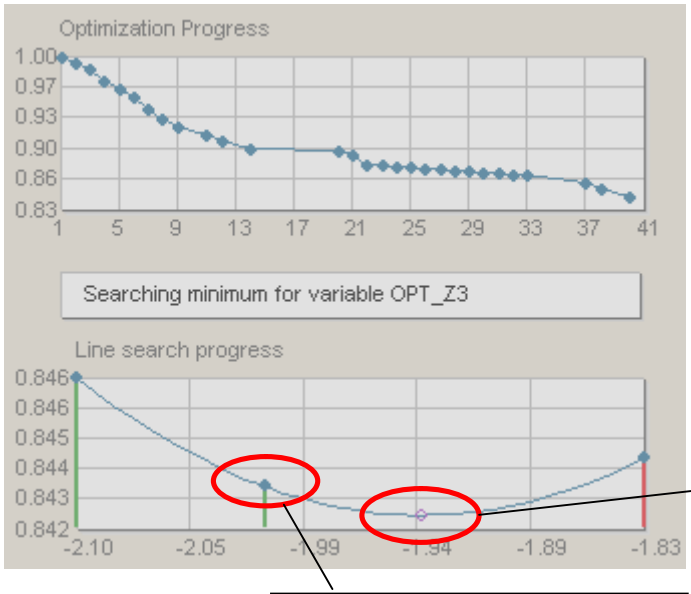


Abbildung 12: Optionen der Achsenparallelen Suche

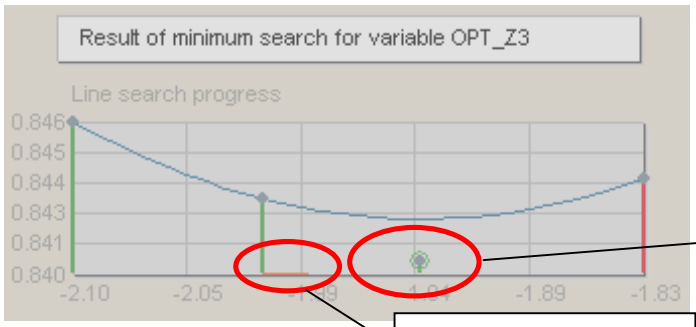
In den Optionen zur Achsenparallelen Suche lassen sich die Anzahl der Iterationen einstellen, pro Iteration werden 3n FE-Berechnungen vorgenommen. Außerdem lässt sich die Startschrittweite für die Punkte der Liniensuche einstellen, entweder am Definitionsbereich der jeweiligen Optimierungsvariablen orientiert ($x_{max} - x_{min}$) oder mit einem festen Wert. Eine minimale

Schrittweite lässt sich ebenfalls vorgeben, diese bewirkt, dass keine FE-Berechnung vorgenommen wird, wenn die berechnete Schrittweite kleiner als dieser Wert ist.
Während der Optimierung wird im oberen Diagramm von Optima der Optimierungsfortschritt angezeigt und im unteren Diagramm der Verlauf der Liniensuche dargestellt. Dies beinhaltet einen Punkt für jede Stützstelle mit einem grünen bzw. roten Balken, falls der Punkt eine oder mehrere Nebenbedingungen verletzt hat. Die x-Achse des Diagramms zeigt den Wert der aktuellen Optimierungsvariablen an, die y-Achse den Zielfunktionswert.



Vermutetes Minimum

Stützstellen der Liniensuche



Tatsächliches Minimum

Neue Schrittweite

Die Schrittweite für die nächste Iteration wird am unteren Rand des Liniensuch-Diagramms angezeigt, ausgehend vom Startpunkt der Liniensuche.
Nebenbedingungen werden über eine Straffunktion und Addition der Strafterme auf die Zielfunktion berücksichtigt.

2.4 Quasi-Newton Verfahren

Das Quasi-Newton Verfahren basiert auf dem Gradientenverfahren, bei dem ausgehend von einem Startpunkt die ***Richtung des steilsten Abstiegs*** bestimmt wird. Diese Richtung gibt der Gradient $\mathbf{g} = -\nabla z(\mathbf{x})$ an, der aus den partiellen Ableitungen für alle Variablen der Funktion besteht. Die partiellen Ableitungen werden mittels des Differenzenquotienten anstatt des Differentialquotienten bestimmt.

Zur Verbesserung der Suchrichtung können Approximationen der Hesseschen Matrix $\mathbf{H} = \nabla^2 z(\mathbf{x})$ herangezogen werden. Zu Beginn der Optimierung kann die Hessesche Matrix einfach durch die Einheitsmatrix angenähert werden, d.h. die Suchrichtung entspricht zunächst dem Gradienten. In jedem Iterationsschritt kann die Approximation mit untenstehender Formel verbessert werden, weshalb auch von *Update- Verfahren* gesprochen wird.

$$\mathbf{H}_{k+1} = g\mathbf{H} + \left(1 + gq\,\frac{\mathbf{q}^T\mathbf{H}\mathbf{q}}{\mathbf{p}^T\mathbf{q}}\right)\cdot\frac{\mathbf{p}\mathbf{p}^T}{\mathbf{p}^T\mathbf{q}} - g\,\frac{(1-q)}{\mathbf{q}^T\mathbf{H}\mathbf{q}}\mathbf{H}\mathbf{q}\cdot\mathbf{q}^T\mathbf{H} - \frac{gq}{\mathbf{p}^T\mathbf{q}}(\mathbf{p}\mathbf{q}^T\mathbf{H} + \mathbf{H}\mathbf{q}\mathbf{p}^T)$$

mit der Vektordifferenz der Optimierungsvariablen der aktuellen und der vorherigen Iteration

$\mathbf{p}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$

und der Vektordifferenz der Gradienten der aktuellen und der vorherigen Iteration.

$\mathbf{q}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$

Die Wahl der Parameter g und q beeinflussen die Güte der Approximation, es können eigene Werte verwendet werden oder eine der bekannten Standardkonfigurationen:

$g \equiv 1,\quad q \equiv 0$ nach Davidon, Fletcher und Powell (DFP Verfahren)

$g \equiv 1,\quad q \equiv 1$ nach Broyden, Fletcher, Goldfarb und Shanno (BFGS Verfahren)

$g \equiv 1,\quad q = \mathbf{p}^T\mathbf{q}/(\mathbf{p}^T\mathbf{q} - \mathbf{q}^T\mathbf{H}\mathbf{q})$ nach Broyden

Aus dem Gradienten und der Hesseschen Matrix wird die Suchrichtung berechnet: $\mathbf{s}_k := \mathbf{H}_k \cdot \mathbf{g}_k$

Anschließend wird in dieser Richtung eine eindimensionale Minimierung, die sogenannte **Liniensuche** durchgeführt:

$\min\ z(\mathbf{x}_{k+1}) := z(\mathbf{x}_k - a \cdot \mathbf{s}_k)$

Ist ein a gefunden, für das die Zielfunktion in der aktuellen Richtung $\mathbf{s}$ minimal wird, so werden die Optimierungsvariablen $\mathbf{x}_x$ ersetzt durch $\mathbf{x}_{k+1} = \mathbf{x}_k - a_{\min} \cdot \mathbf{s}_k$ und die nächst Iteration wird begonnen.

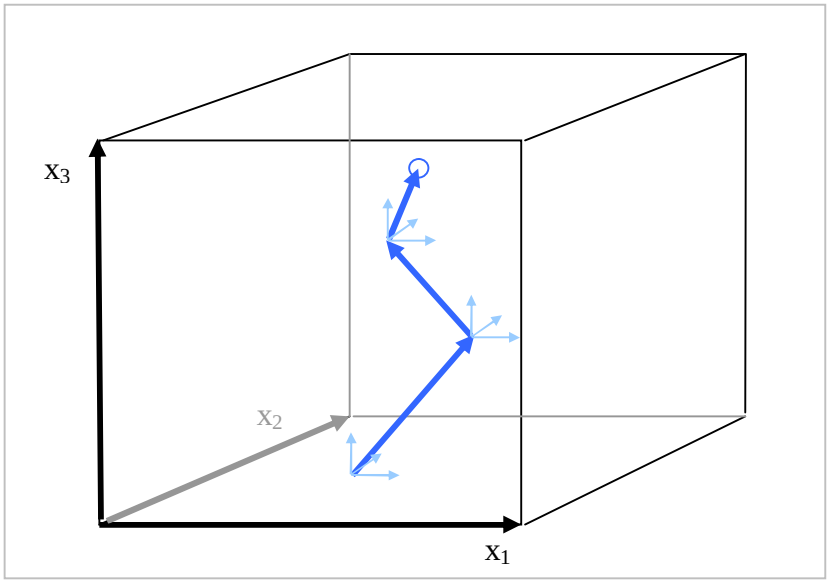


Abbildung 13: Iterationsverlauf beim Gradientenverfahren bzw. Quasi-Newton Verfahren

Auf mathematische Funktionen angewendet, kann der Differenzenquotient bei sehr kleinen

Die Vorgaben für das Optimierungsverfahren werden in nachfolgendem Dialog eingegeben.

Für weiterführende Erläuterungen verweisen wir auf die Fachliteratur.

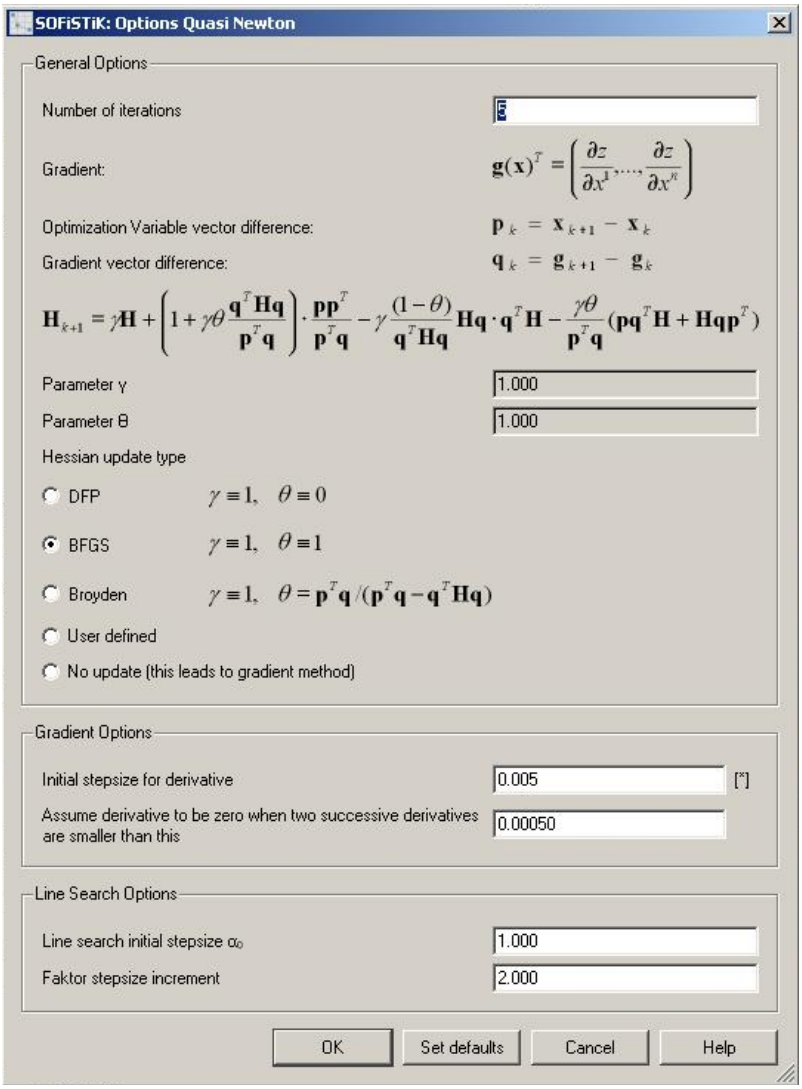


Abbildung 14: Optionen des Quasi-Newton Verfahrens

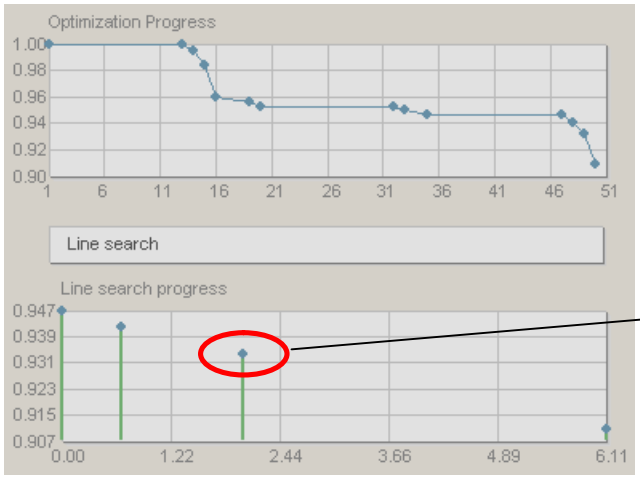
Neben der Anzahl der Iterationen lässt sich die Art des Update Verfahrens einstellen. Ferner können die Schrittweite zur Gradientenberechnung und eine Mindestgröße für partielle Ableitungen eingeben, die nach zweimaliger Unterschreitung dazu führt, dass die Ableitung für die betreffende Variable nicht mehr neu berechnet wird.

Schließlich lassen sich noch die Startschrittweite für die Liniensuche a_0 und das Inkrement der Schrittweite b angeben. Solange der Zielfunktionswert in Suchrichtung abfällt, wird a erhöht:

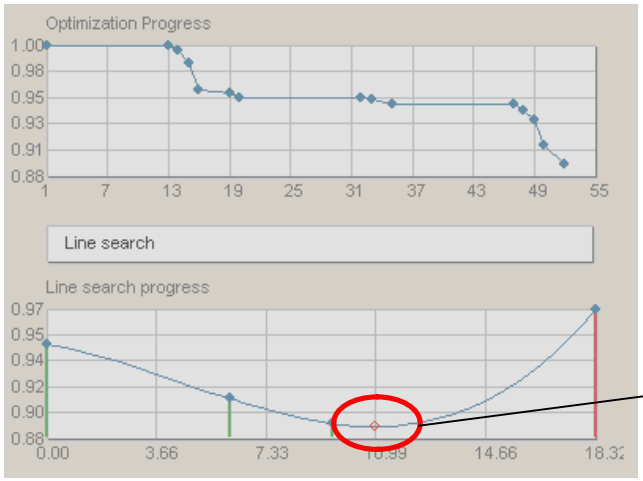
$$a_i = a_{i-1} \cdot b$$

Der Verlauf der Liniensuche wird im unteren Diagramm dargestellt, dabei ist a auf der x-Achse aufgetragen, der Zielfunktionswert auf der y-Achse.

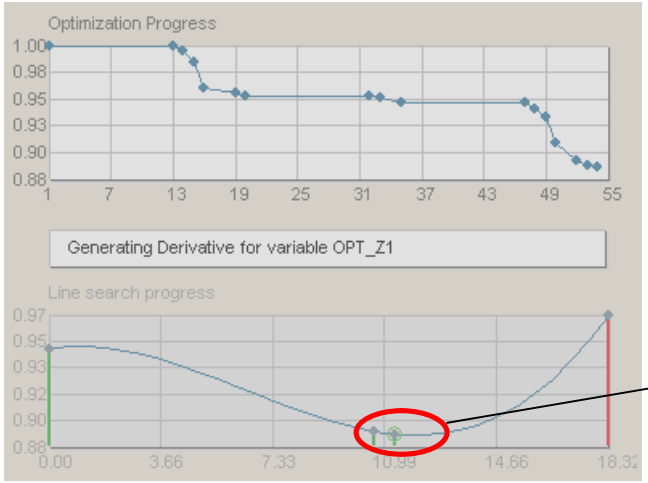
Stützstellen im zulässigen Bereich erhalten einen grünen Balken, wurden jedoch Nebenbedingungen verletzt, ist der Balken rot.



Stützstellen



Vermutetes Minimum



Nebenbedingungen werden über eine Straffunktion und Addition der Strafterme auf die Zielfunktion berücksichtigt.

3 ANWENDUNG

Das Programm OPTIMA wird ausschließlich über einen CADINP Eingabedatensatz und den TEDDY gesteuert.

Für die Durchführung der Optimierung ist ein Datensatz nach folgendem grundsätzlichem Aufbau notwendig

```
+prog template    $ Definition der Optimierungsvariablen
+prog aqua        $ Definition Material und Querschnitte
+prog sofimsha    $ Systemerzeugung
+prog sofiload    $ Lastdefinition
+prog ase/star    $ Berechnung der Lastfälle
+prog maxima      $ Berechnung der Überlagerungen
+prog aqb/bemess  $ Bemessung
+prog template    $ Zielfunktion und Nebenbedingungen
```

Der Programmstart von OPTIMA erfolgt einfach über die Command-Shell mit der Eingabe *optima datfile.[dat]*.

HINWEIS: Ein gleichzeitiger Zugriff auf die Datenbasis mit WPS und OPTIMA ist nicht gestattet.

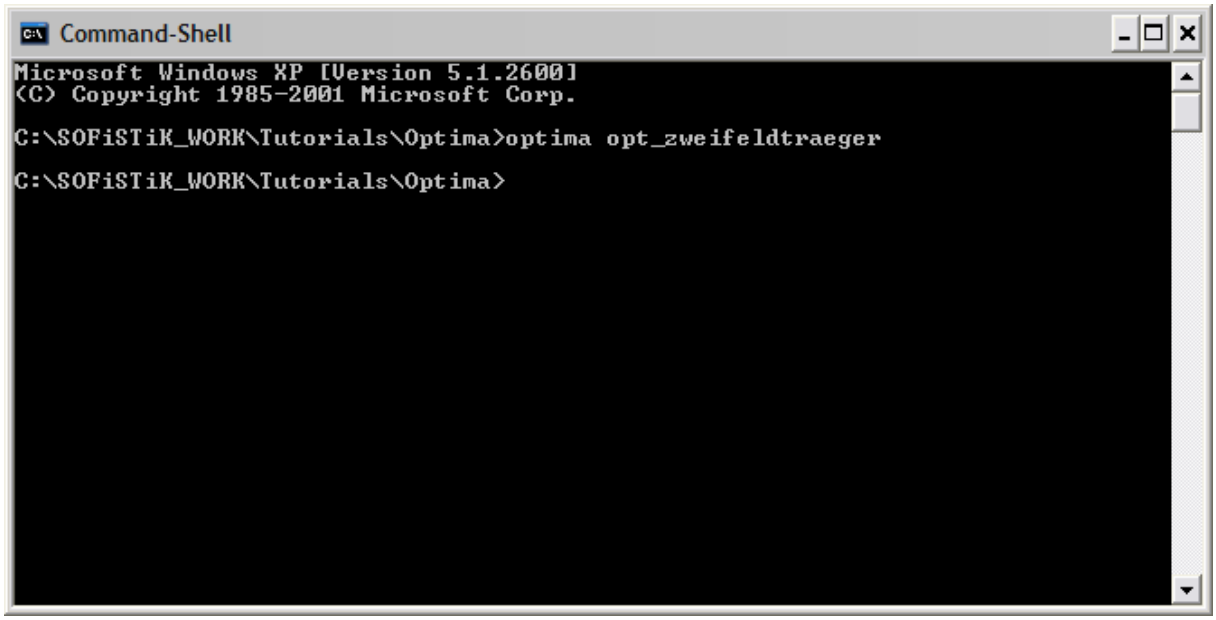


Abbildung 15: Start OPTIMA über die Command-Shell

3.1 Definition der Optimierungsvariablen

Am Beginn der Eingabedatei werden in einem PROG TEMPLATE Block die Optimierungsvariablen eingegeben:

```
+prog template
kopf  Optimization Variables

sto#opt_z1    -2.6,-6,-0.1
sto#opt_z2    -2.6,-6,-0.1
sto#opt_z3    -2.6,-6,-0.1
sto#opt_z4    -2.6,-6,-0.1
sto#opt_z5    -2.6,-6,-0.1
sto#opt_z6    -2.6,-6,-0.1

sto#opt_x1    0.01,0.01,1.5
sto#opt_x2    2,1.6,3
sto#opt_x3    4,3.1,5
sto#opt_x4    6,5.1,7
sto#opt_x5    8,7.1,9
sto#opt_x6    10,9.1,10.9

ende
```

Name der Variable

Startwert, Minimalwert,

3.2 Verwendung der Optimierungsvariablen

In der folgenden Systemeingabe können die Optimierungsvariablen für beliebige Werte verwendet werden. Allerdings sollten die Variablen die gewünschte Zielfunktion wie z.B. das Gesamtgewicht auch tatsächlich beeinflussen.

Verwendung der Optimierungsvariablen für Knotenkoordinaten (Formoptimierung):

```
knot nr   1 x #opt_x1      y  0    z #opt_z1
knot nr   2 x #opt_x2      y  0    z #opt_z2
knot nr   3 x #opt_x3      y  0    z #opt_z3
knot nr   4 x #opt_x4      y  0    z #opt_z4
knot nr   5 x #opt_x5      y  0    z #opt_z5
knot nr   6 x #opt_x6      y  0    z #opt_z6
```

3.3 Eingabe der Zielfunktion und Nebenbedingungen

In einem „prog template“ Block am Ende der Eingabedatei kann die Zielfunktion definiert werden, indem ein beliebiger (skalarer) Wert in die reservierte Variable „opt_target“ geschrieben wird. Dazu müssen die gewünschten Werte mittels „@key“ aus der Datenbasis geholt werden, wie beispielsweise das Gewicht einer Struktur. Dieser Wert muss natürlich von den Optimierungsvariablen abhängen, damit eine echte Zielfunktion z(**x**) gegeben ist.
Mit den Nebenbedingungen wird ebenso verfahren, hierfür sind alle mit „opt_cons“ beginnenden Variablennamen reserviert.
Es gibt genau eine Zielfunktion und beliebig viele Nebenbedingungen.

```
+prog template urs:12
kopf  target function

$ Gesamtgewicht = prop. zur Gesamtlänge der Stäbe
let#cdb_ier 0
@key TRUS
let#tot_length 0
loop  999
  let#length @(DL)  $ Länge des Fachwerkstabes
  let#tot_length #tot_length+#length
endloop #cdb_ier<2

$ Zielfkt aufstellen
sto#opt_target #tot_length

$ max uz in mm
let#lf 1
@key N_DISP #lf
let#uz @(UZ)*1000

$ Knicken Ausnutzung
let#buck #usage(10),1

$ Nebenbedingungen
sto#opt_cons_1   #uz,0,50
sto#opt_cons_2   #buck,0,1
```

Zielfunktionswert

Nebenbedingung

Aktueller Wert, Minimalwert, Maximalwert

Gleichheitsnebenbedingungen können eingegeben werden, indem der Minimalwert dem Maximalwert entspricht. Gleichheitsnebenbedingungen sind immer verletzt, nur die Größe der Überschreitung kann verringert werden.

3.4 Start der Optimierung

Die grafische Oberfläche von Optima beinhaltet ein Modulfenster mit Ausgabe des Berechnungsprotokolls, einen Bereich zur Auswahl des Optimierungsverfahrens über Radiobuttons und jeweils eine Tabelle für die Anzeige der Optimierungsvariablen und Nebenbedingungen. Ein Diagramm informiert über den Optimierungsfortschritt und ein zweites Diagramm gibt je nach Verfahren zusätzliche Informationen.

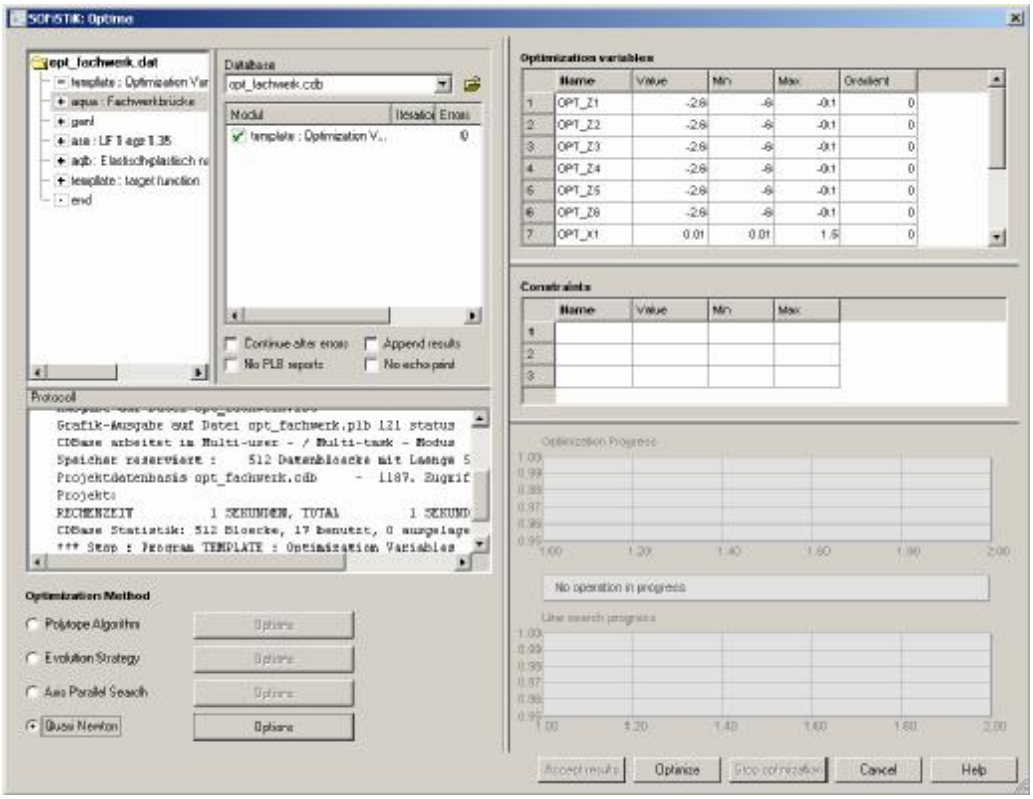


Abbildung 16: Hauptfenster von Optima

Nach Beendigung der Optimierung kann eine erneute Optimierung vorgenommen werden, auch mit einem anderen Verfahren, oder es können die erzielten Ergebnisse übernommen werden. In diesem Fall wird ein „prog template“ Block mit den optimierten Variablen in die Eingabedatei geschrieben und auf „+“ gesetzt. Der alte „prog template“ Block bleibt erhalten, wird jedoch auf „-“ gesetzt.

4 ERWEITERTE ANWENDUNGSMÖGLICHKEITEN

4.1 Konvergenz statt Minimum

Anstatt den Zielfunktionswert zu minimieren oder zu maximieren, kann man ihn auch gegen einen bestimmten Wert konvergieren lassen, indem opt_target = $(z(\mathbf{x})-a)^2$ minimiert wird. Dies führt dazu, dass $z(\mathbf{x})$ gegen a konvergiert. Dies lässt sich auch auf mehrere Funktionen anwenden: opt_target = $\Sigma(z_i(\mathbf{x})-a_i)^2$.

4.2 Kostenminimierung

Kosten lassen sich über den Einheitspreis der Massen eines Systems minimieren, d.h. $z(\mathbf{x}) = m_1*EP_1 + m_2*EP_2 + ... + m_k*EP_k$.

4.3 Diskrete Optimierung / Querschnittsoptimierung

Alle Optimierungsvariablen die mit „optg_“ statt mit „opt_“ beginnen, werden als ganzzahlig behandelt. Damit lassen sich beispielsweise Stabquerschnitte optimieren, indem die Querschnittsnummer eines Stabes durch die Optimierungsvariable „optg_qnr“ definiert wird, und über eine Tabelle die Zuordnung der Nummer zu einem Profil hergestellt wird:

```
+prog template
kopf  Optimization Variables
sto#optg_qnr   3,1,6
ende
```

Startwert = 3,

Wertebereich 1,2,3,4,5,6

5 BEISPIELE

5.1 Formoptimierung Zweifeldträger

Als Einstiegsbeispiel wird ein einfacher Zweifeldträger betrachtet, der an einer Seite eingespannt ist. Als Belastung wird hier eine einfache Gleichstreckenlast in Höhe von 22 kN/m gewählt. Der Stahlträger besteht aus S 235 und einem HEA 100 Profil.

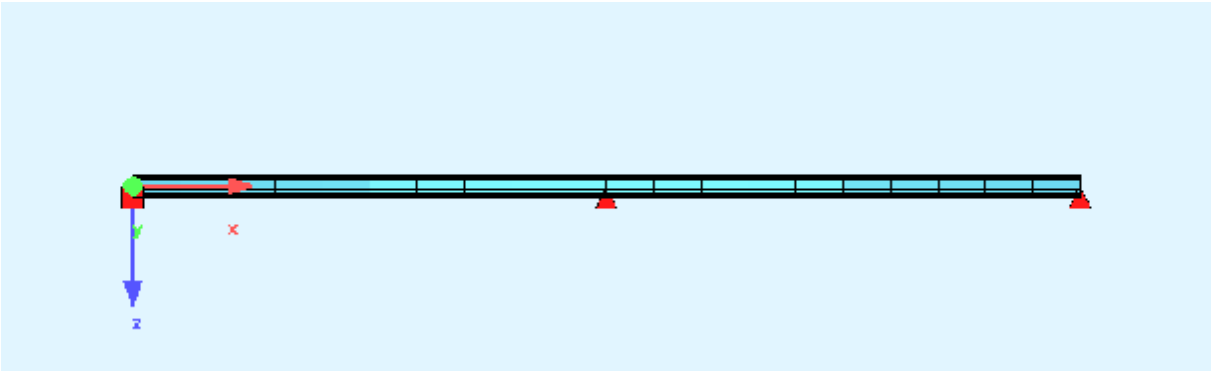


Abbildung 17: Zweifeldträger System

Aufgabe ist es eine „optimale“ Struktur zu finden, wobei als einzige Variable die x-Koordinate des Zwischenauflagers verwendet wird. Vermutlich lässt sich diese Aufgabe damit lösen, eine gleichmäßig minimale Beanspruchung anzustreben. Auf dieser Vermutung aufbauend werden im Nachgang einige Ansätze für Zielfunktionen diskutiert.

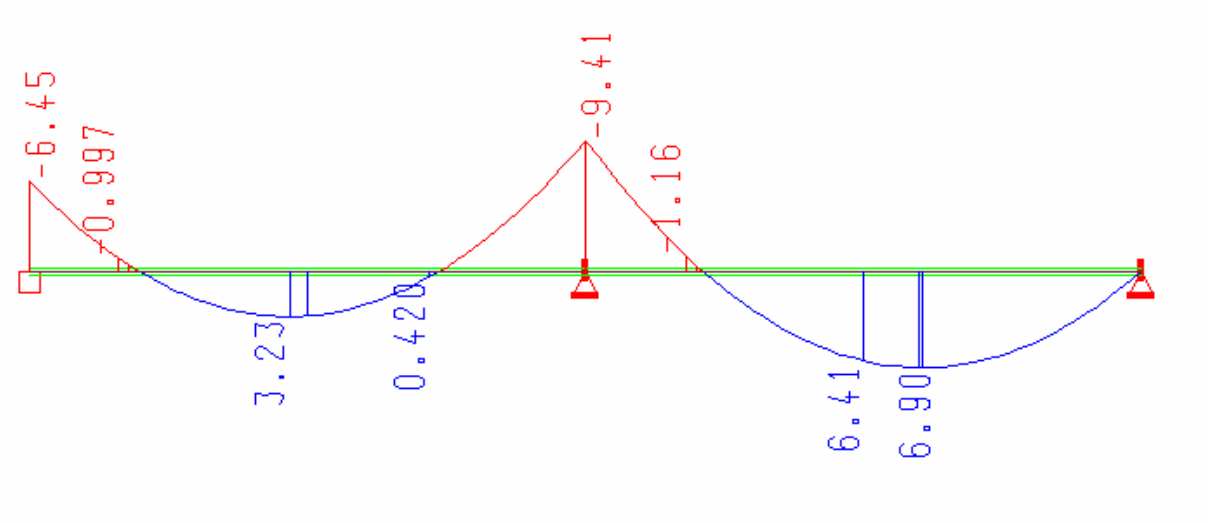


Abbildung 18: Momentenverteilung Ausgangssystem

Das System besteht aus insgesamt 5 Knoten und 4 Stäben. Jedes Feld ist über 2 Stäbe mit einem Mittelknoten idealisiert.

Die Variable opt_x wird in einem ersten Block PROG TEMPLATE definiert.

```
+prog template
kopf  Optimization Variables
sto#opt_x   2,1,3.8
ende
```

Maximalwert = 3.8 m

Minimalwert = 1,0 m

Ausgangswert = 2,0 m

Die Formulierung der Zielfunktion ist maßgebend für das Optimierungsergebnis. Durch ungeschickte Zielfunktionen können die Ergebnisse deutlich von den gewünschten, bzw. erwarteten Ergebnissen abweichen. Auch ist es sehr wichtig, dass die Optimierungsaufgabe und die zu formulierende Zielfunktion auch ein mögliches Ergebnis darstellen. Für die Problemstellung der gleichmäßigen Momentenausnutzung werden verschiedene Ansätze untersucht.

1. Ansatz: Minimierung des Gewichtes:

Da der Querschnitt und damit auch das Gewicht in diesem Beispiel konstant ist, führt dieser Ansatz nicht zum Ziel.

2. Ansatz: Stützmoment = Feldmoment

Da eine Gleichheitsbeziehung nicht optimierbar ist, muss ein anderer Ansatz gefunden werden. Ein Ansatz über den absoluten Wert der Summe ABS(max MY + min MY) ist denkbar. Es ist aber besser das Quadrat der Summe SQR(max MY + min MY) zu verwenden, da dies eine stetig differenzierbare Funktion ist. Die Eingabe in CADINP lautet:

```
+prog template
kopf  Zielfunktion – gleichmäßge Ausnutzung
$ Auslesen von max MY und min MY aus der Datenbasis
let#lf 1
@key BEAM_FOR #lf
Let#maxm @(0,my)
Let#minm @(0,my)

$ Zielfunktion aufstellen
sto#opt_target (#maxm+#minm)**2
```

Betrachtet man das Ergebnis, dann stellt man fest, dass die Spannweite im Feld 1 verkürzt wird. Die Zielfunktion ist nämlich dafür ausgelegt, dass Stütz- und Feldmoment ähnlich groß werden und das ist nur durch eine Vergrößerung der 2. Spannweite möglich. Allerdings handelt es sich hierbei um eine schlechte Zielfunktion, denn die maximalen und minimalen Momente werden größer und damit auch die Ausnutzung unwirtschaftlicher.

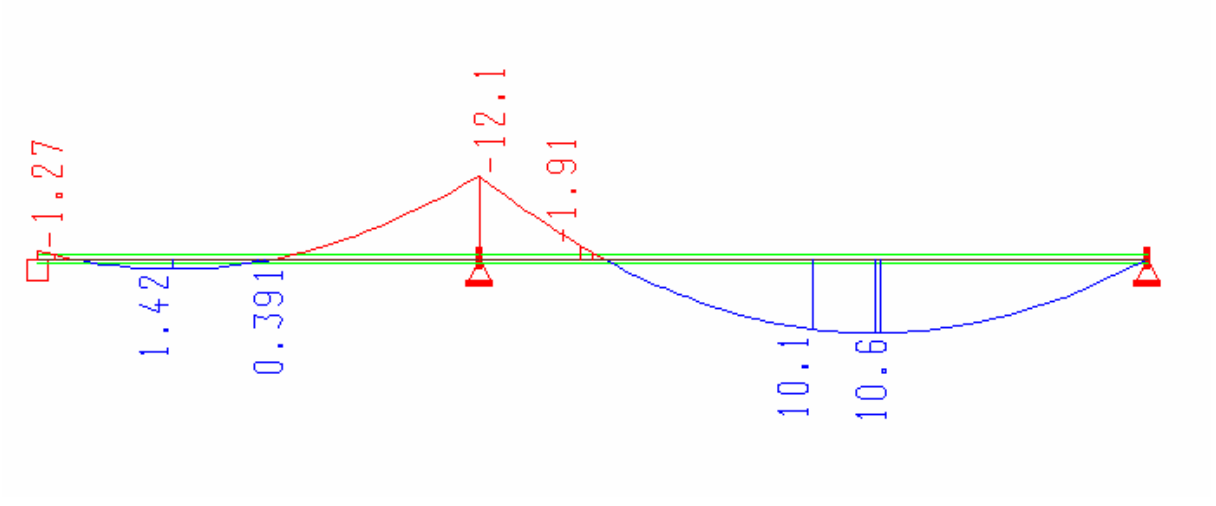


Abbildung 19: Momentenverteilung nach Optimierung Zielfunktion 2

3. Ansatz: Minimierung der Verformungsenergie

Eine weitere Möglichkeit ist die Minimierung der Gesamtenergie. Die Ergebniswerte werden im PROG ASE berechnet und über die Eingabe ECHO STAT angefordert.

```
+prog template
kopf  Zielfunktion – gleichmäßge Ausnutzung
$ Auslesen von max MY und min MY aus der Datenbasis
let#lf 1
@key GRP_LC #lf
Let#En @(0,ETOT)

$ Zielfunktion aufstellen
sto#opt_target #En

ende
```

Als Ergebnis wird eine neue x-Koordinate des mittleren Auflagers von 2.12 m ermittelt.

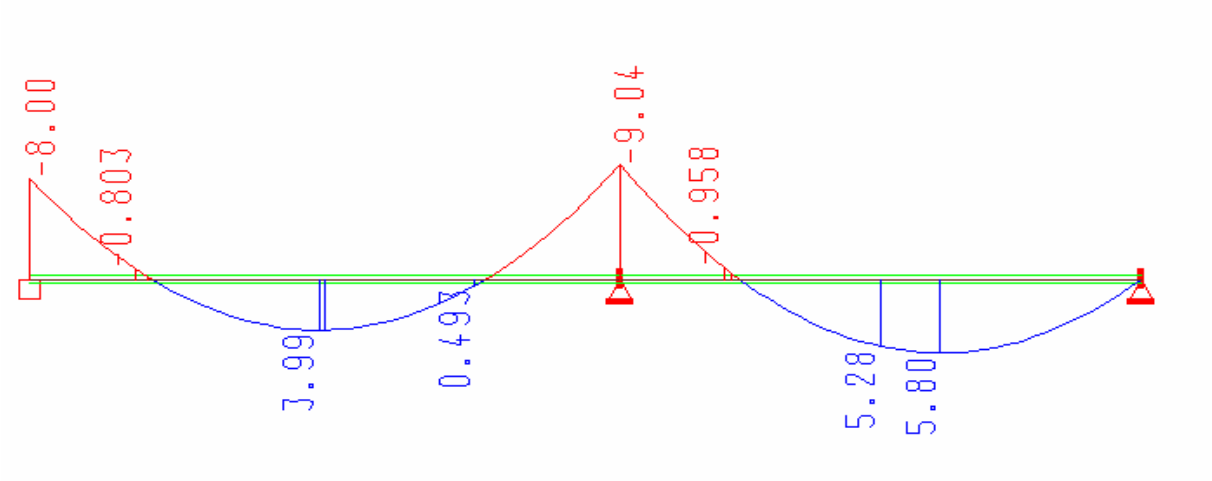


Abbildung 20: Momentenverteilung nach Optimierung Zielfunktion 3

Der Momentverlauf zeigt eine bessere Ausnutzung der Feld- und Stützmomente, aber es sind immer noch Unterschiede vorhanden. Der Verlauf der Optimierung mit dem Quasi Newton Verfahren ist in der nachfolgenden Abbildung dargestellt.

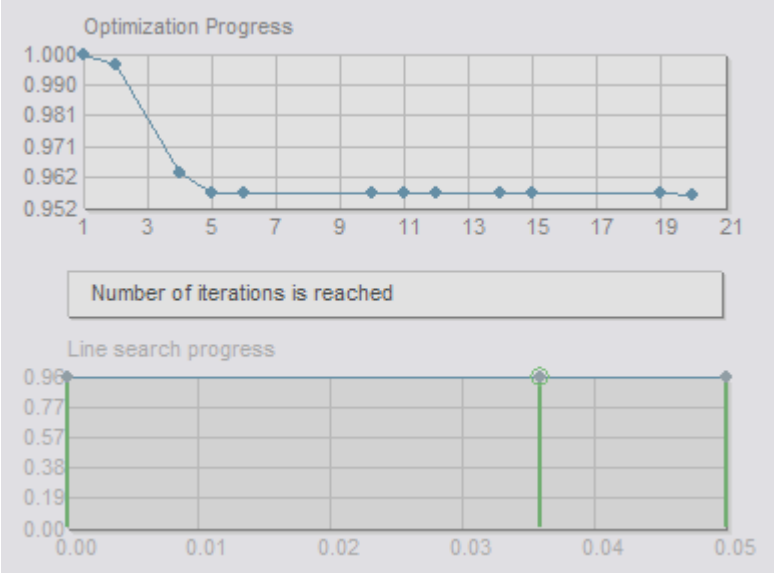


Abbildung 21: Optimierungsverlauf Quasi Newton Verfahren bei Zielfunktion 3

4. Ansatz: Minimierung der Summe der Quadrate der Momente

Als weitere Zielfunktion wird die Summe der Quadrate von max MY und min MY gewählt. Das Auslesen der maximalen und minimalen Momente MY aus der CDB erfolgt wie im Ansatz 2.

```
$ Zielfunktion aufstellen
sto#opt_target (#maxm**2+#minm**2)
```

Mit dieser Zielfunktion wird für die x-Koordinate des mittleren Auflagers ein Wert von 2.23 m ermittelt.

Die Unterschiede der der Feld- und Stützmomente ist mit dieser Zielfunktion weiter verringert worden.

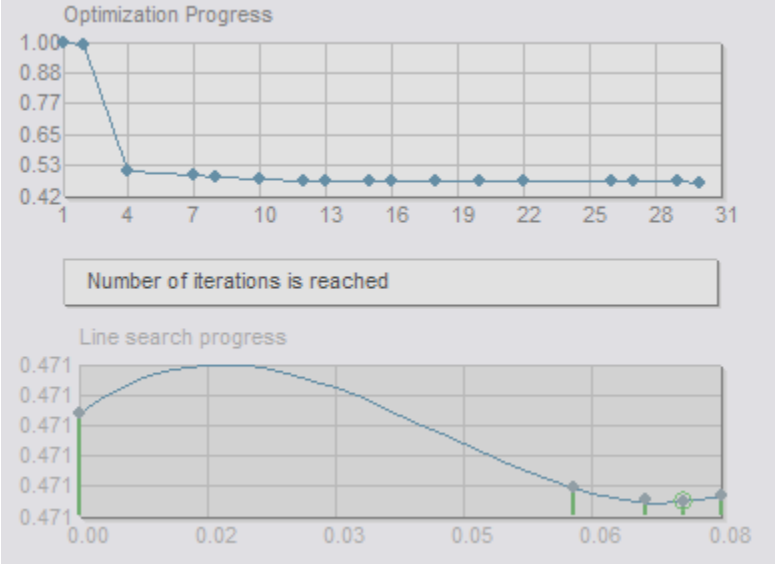


Abbildung 22: Optimierungsverlauf Quasi Newton Verfahren bei Zielfunktion 4

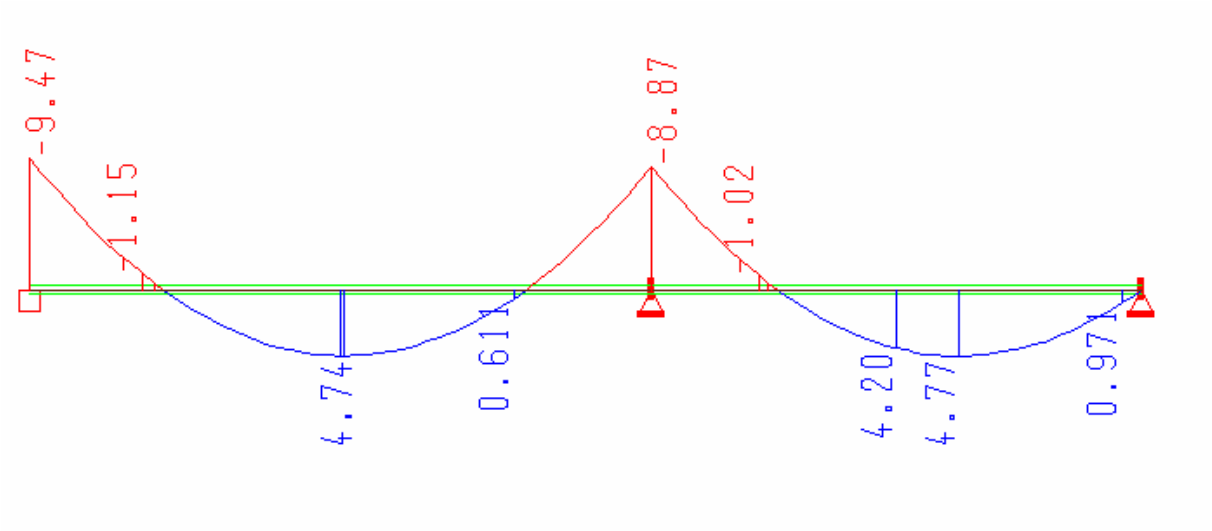


Abbildung 23: Momentenverteilung nach Optimierung Zielfunktion 4

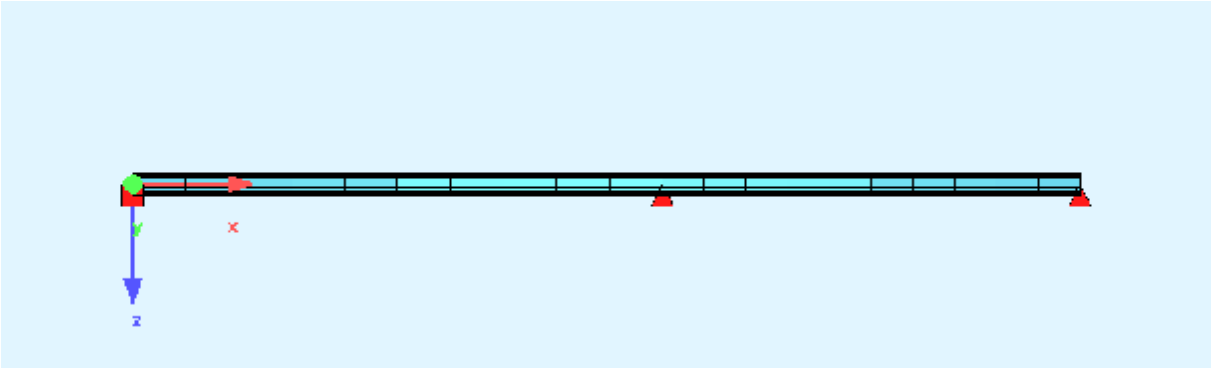


Abbildung 24: System nach Optimierung mit verlängertem 1. Feld

Bereits dieses kleine Beispiel zeigt, dass die Definition des Ausgangssystems und der Zielfunktion einen maßgebenden Einfluss auf die Optimierung hat.

5.2 Formoptimierung Fachwerkbrücke

Als weiteres Beispiel für eine einfache Formoptimierung soll ein Brückenfachwerk dienen. Durch die Veränderung der Knotenkoordinaten des oberen Gurtes in z- Richtung soll das Gesamtgewicht der Brückenkonstruktion minimiert werden. Für dieses einfache Beispiel wird nur ein Querschnitt HEA 220 für alle Fachwerkstäbe verwendet. Als Material wird Baustahl S 235 nach DIN 18800 verwendet.

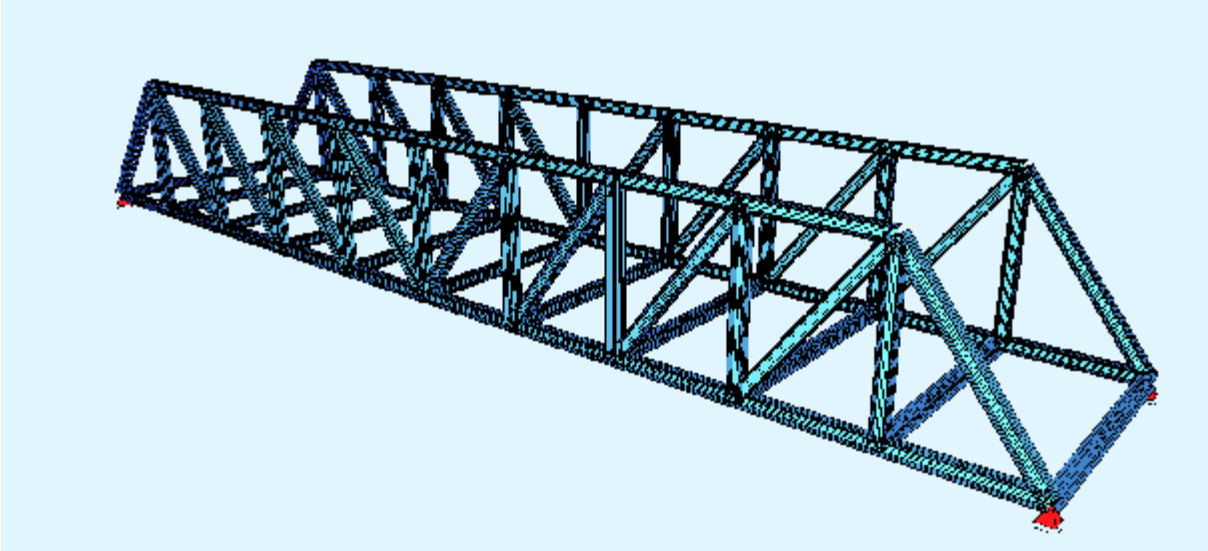


Abbildung 25: Isometrie des Brückenfachwerks

Die Knotenkoordinaten des oberen Gurtes sind variabel in z- Richtung. Unter Ausnutzung der Symmetrie in zwei Richtungen werden insgesamt 5 Variablen für die z-Koordinaten erforderlich.

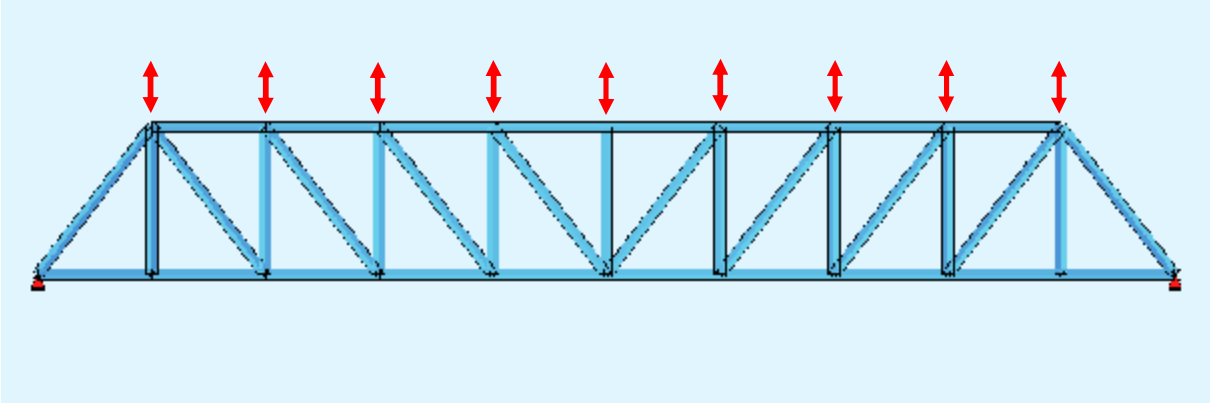


Abbildung 26: Optimierungsvariablen

```
+prog template
kopf  Optimization Variables
sto#opt_z2   -2.6,-6,-0.1
sto#opt_z3   -2.6,-6,-0.1
sto#opt_z4   -2.6,-6,-0.1
sto#opt_z5   -2.6,-6,-0.1
sto#opt_z6   -2.6,-6,-0.1
ende
```

Als Belastung werden Einzellasten im LF 1 auf die Untergurtknoten in Höhe von 1.35·G + 1.5·Q angesetzt, wobei Q einer Flächenlast von 16.7 kN/m² entspricht (SLW 30).
Die Formulierung der Zielfunktion ist entscheidend für das Optimierungsergebnis. In diesem Beispiel soll das Gesamtgewicht der Konstruktion minimiert werden. Da dieses proportional zur Gesamtlänge der Fachwerkstäbe ist, wird die Gesamtlänge als Zielfunktion verwendet.
Neben der Zielfunktion werden zwei Nebenbedingungen formuliert. Die 1. Nebenbedingung ist die Begrenzung der maximalen Verschiebung auf 50 mm. Die 2. Nebenbedingung ist die Stabilität der Fachwerkstäbe. Für den Ausnutzungsgrad wird gefordert: $\kappa < 1$.

```
+prog template
kopf target function
$ Gesamtgewicht = prop. zur Gesamtlänge der Stäbe
let#cdb_ier 0
@key TRUS
let#tot_length 0
loop  999
  let#length @(DL)  $ Länge des Fachwerkstabes
  let#tot_length #tot_length+#length
endloop #cdb_ier<2
$ Zielfunktion aufstellen
sto#opt_target #tot_length

$ max uz in mm
let#lf 1
@key N_DISP #lf
let#uz @(UZ)*1000

$ Knicken Ausnutzung
let#buck #usage(10),1

$ Nebenbedingungen
sto#opt_cons_1   #uz,0,50
sto#opt_cons_2   #buck,0,1
ende
```

Die Aufgabenstellung wurde mit den verschiedenen Optimierungsalgorithmen bearbeitet. In den nachfolgenden Bildern sind die Ergebnisse zusammengestellt.

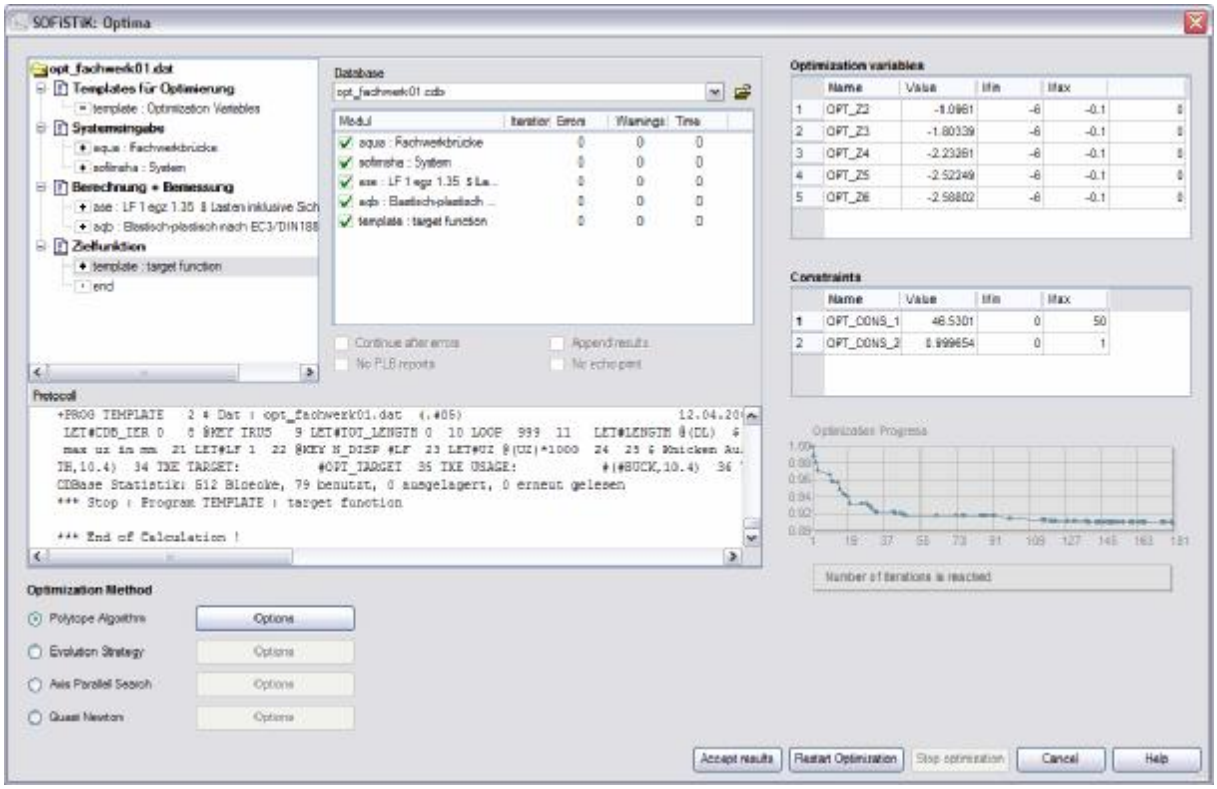


Abbildung 27: Ergebnis Polytop Algorithmus nach 2 Optimierungsläufen

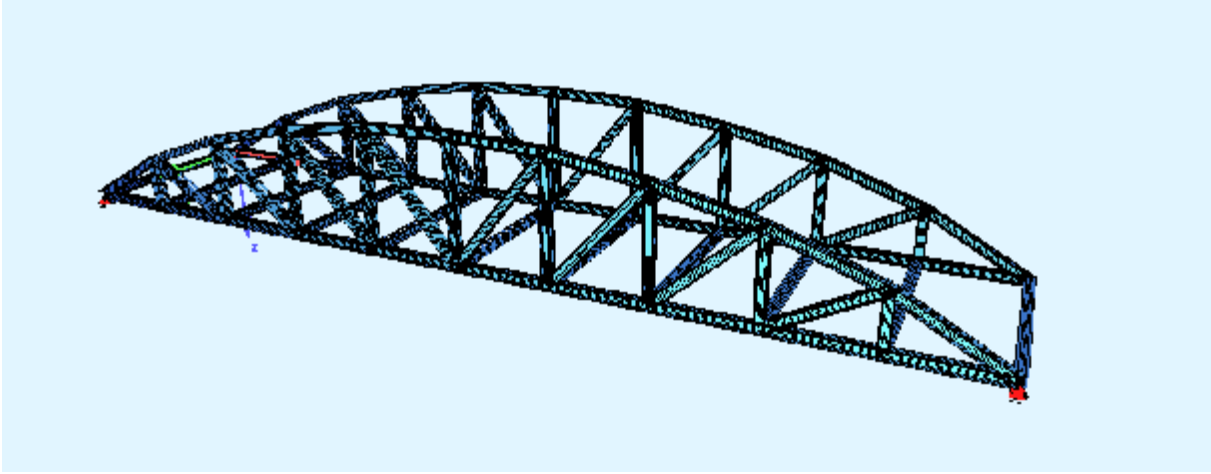


Abbildung 28: Fachwerkbrücke nach Optimierung mit Polytop Algorithmus

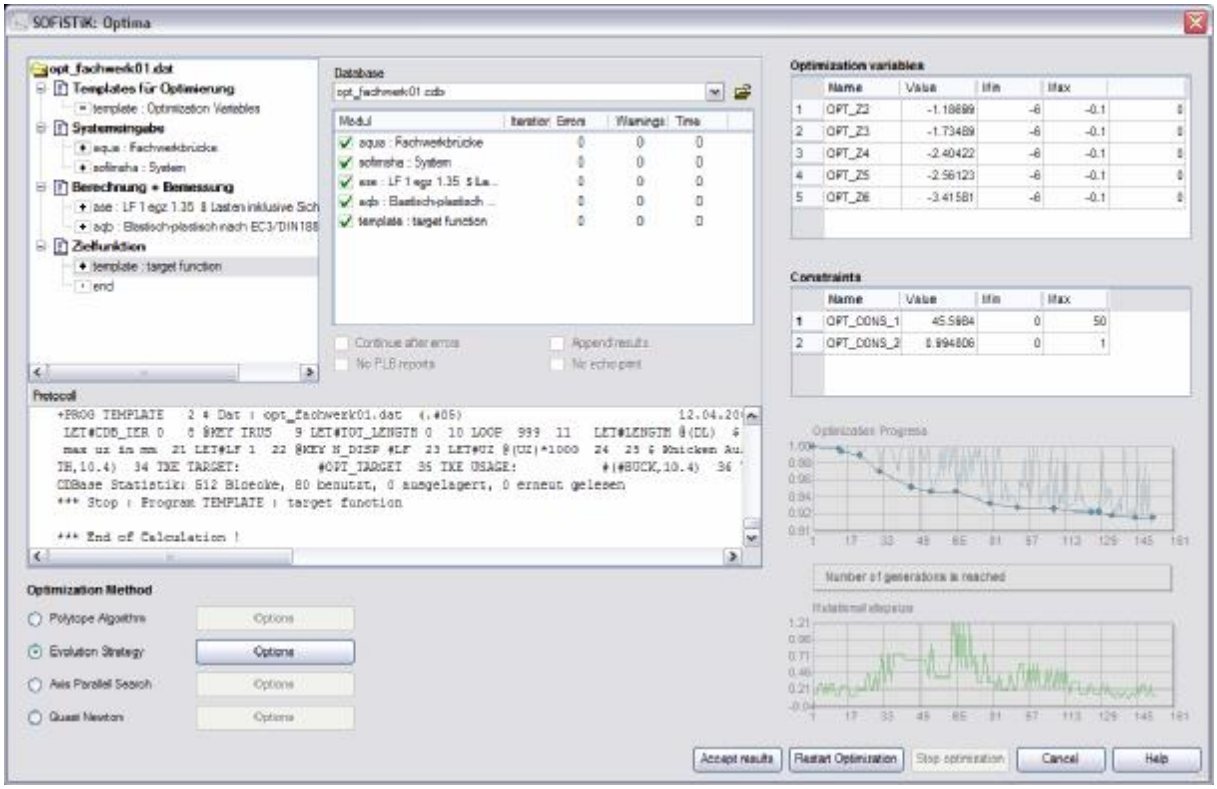


Abbildung 29: Ergebnis Evolutions Strategie nach 3 Optimierungsläufen

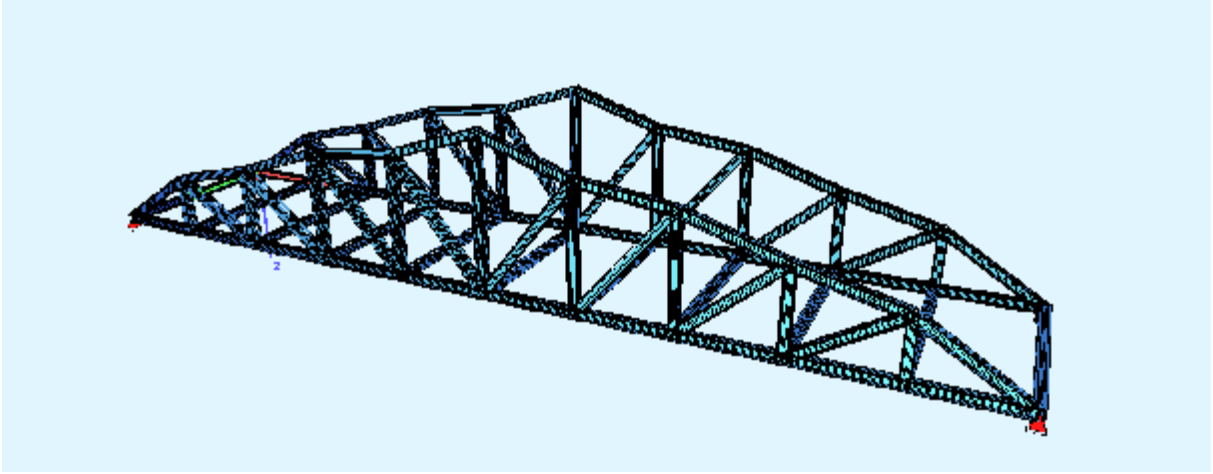


Abbildung 30: Fachwerkbrücke nach Optimierung mit Evolutions Strategie

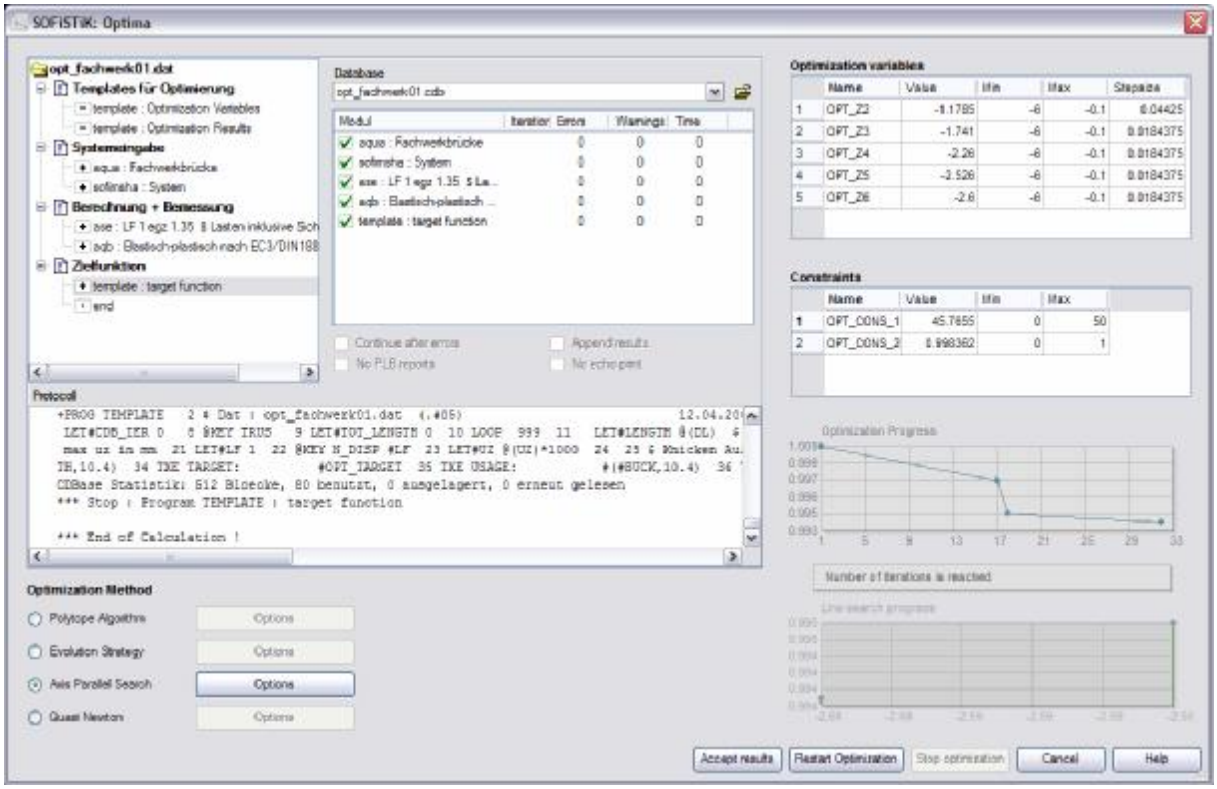


Abbildung 31: Ergebnis Achsen Parallele Suche nach 2 Optimierungsläufen

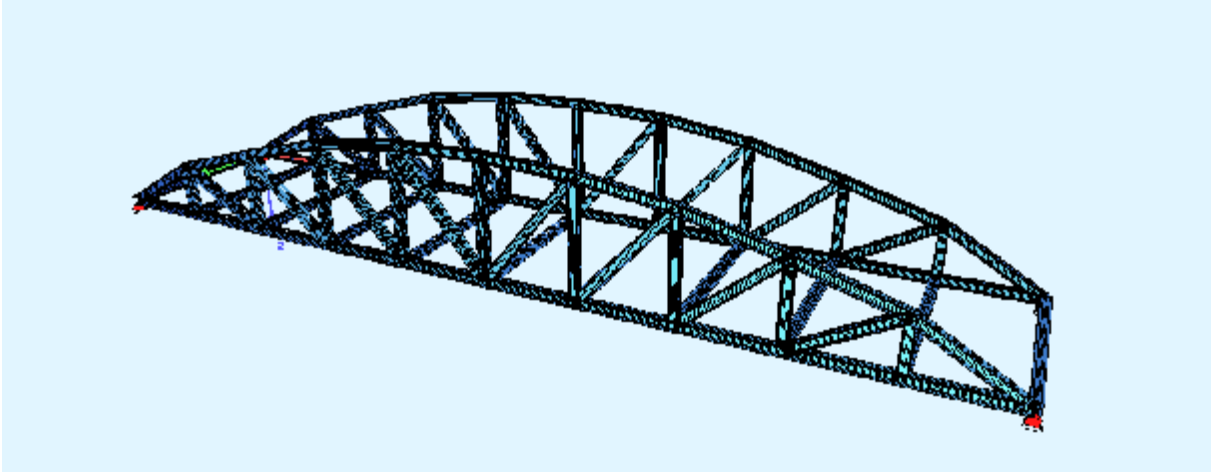


Abbildung 32: Fachwerkbrücke nach Optimierung mit Achsen Paralleler Suche

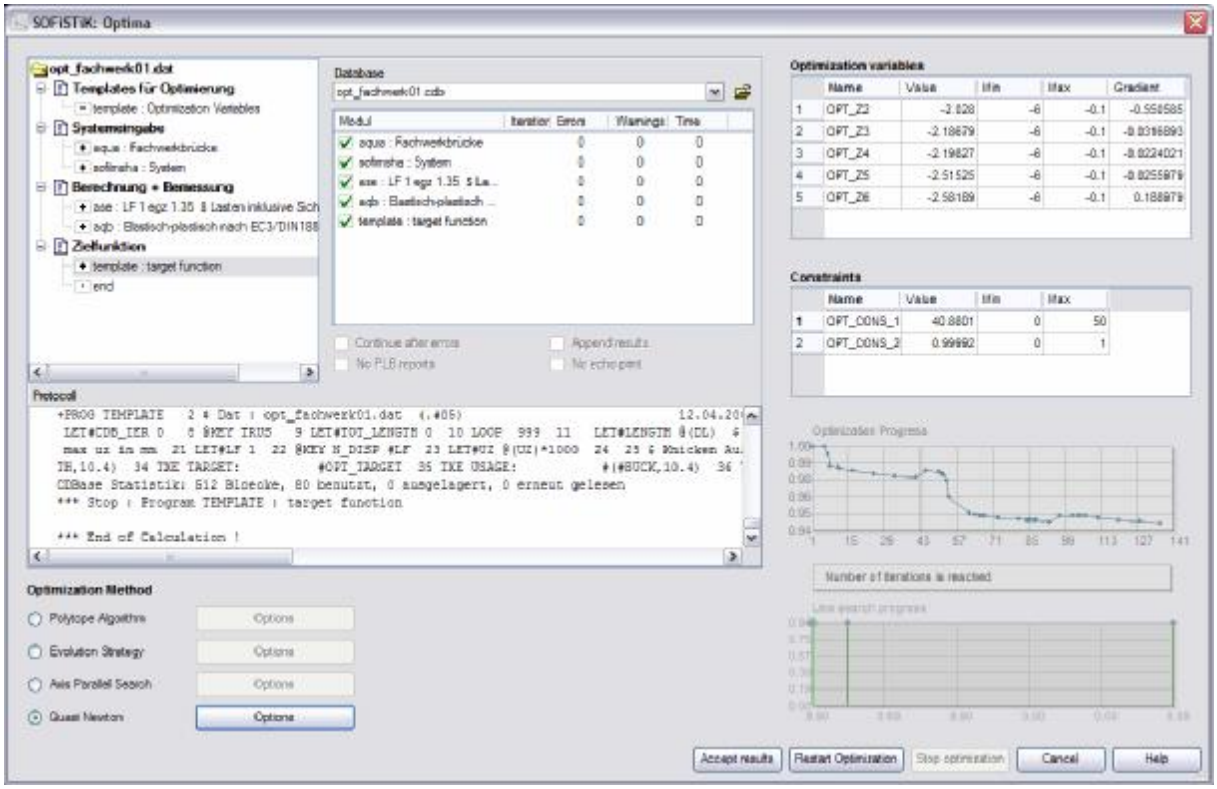


Abbildung 33: Ergebnis Quasi Newton Verfahren nach 3 Optimierungsläufen

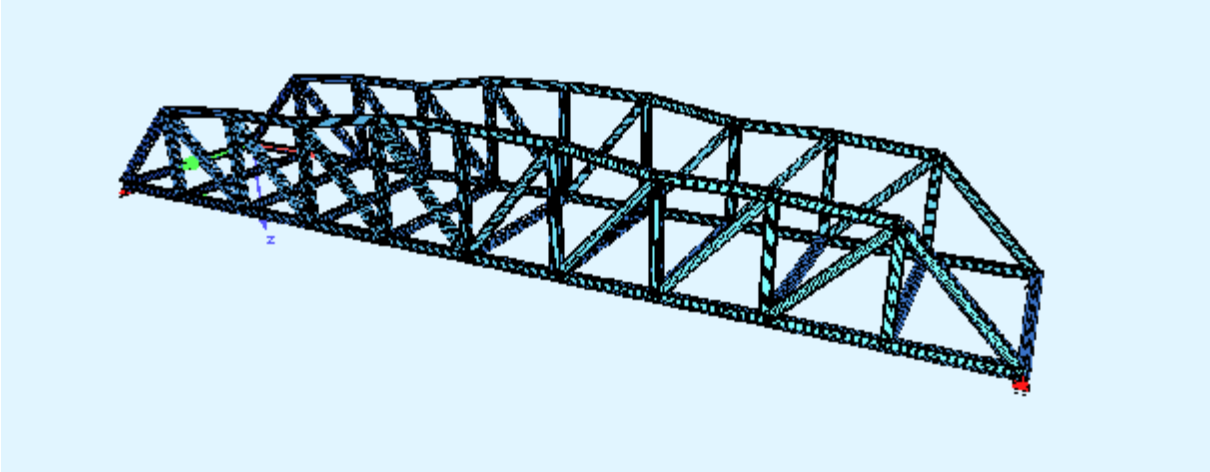


Abbildung 34: Fachwerkbrücke nach Optimierung mit Quasi Newton Verfahren

5.3 Querschnittsoptimierung Einfeldträger

Am Beispiel eines einfachen Balkens mit einer Spannweite von 10 m und einer Gleichlast von 27 kN/m soll eine Querschnittsoptimierung durchgeführt werden. Der Balken wird mit 10 Stäben idealisiert und unter Ausnutzung der Symmetrie werden insgesamt 6 Querschnitte definiert. Es wird ein Stahlquerschnitt HEA 200 aus Baustahl S 235 verwendet, der in der Höhe variabel ist. Am Auflager wird die Höhe mit 200 mm fest vorgegeben, die restlichen 5 Querschnittshöhen werden als Optimierungsvariablen definiert.

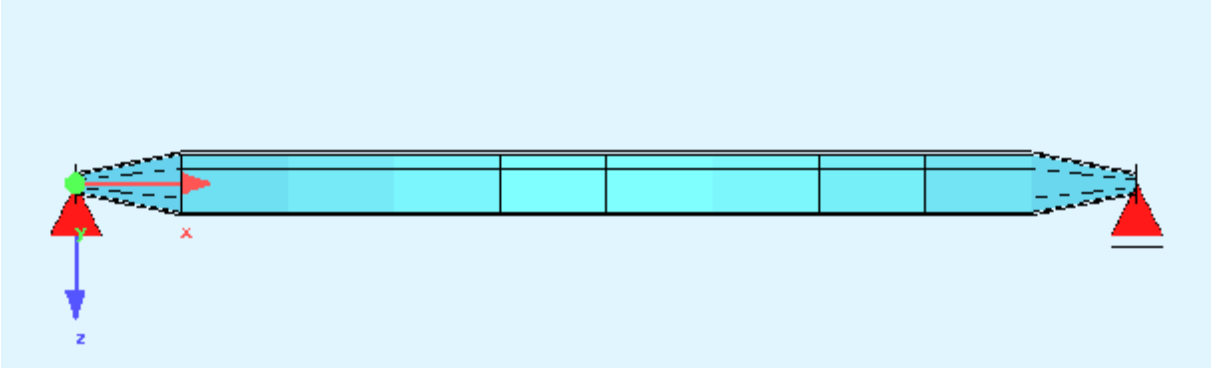


Abbildung 35: Balken vor und nach der Optimierung.

Die Optimierungsvariablen werden wie folgt definiert:

```
+prog template
kopf  Optimization Variables
sto#opt_h2   0.600,0.200,0.800
sto#opt_h3   0.600,0.200,0.800
sto#opt_h4   0.600,0.200,0.800
sto#opt_h5   0.600,0.200,0.800
sto#opt_h6   0.600,0.200,0.800
ende
```

Als Zielfunktion wird in diesem Beispiel die Gesamtmasse des Trägers verwendet.

```
+prog template
$ Zielfunktion:
$ Minimierung der Gesamtmasse
sto#opt_target   #MAT_MASS(0)

$ Nebenbedingung:
$ Ausnutzung der Querschnitte sollte
$ Zugriff auf die Ausnutzungen der Einzelnen Querschnitte
@key BEAM_DES 1
let#ausn1  @(1,tcf)
let#ausn2  @(2,tcf)
let#ausn3  @(3,tcf)
let#ausn4  @(4,tcf)
let#ausn5  @(5,tcf)
let#ausn6  @(6,tcf)


$ Nebenbedingungen werden gespeichert
sto#opt_cons_2   #ausn2,0,1
sto#opt_cons_3   #ausn3,0,1
sto#opt_cons_4   #ausn4,0,1
sto#opt_cons_5   #ausn5,0,1
sto#opt_cons_6   #ausn6,0,1
ende
```

Die Ergebnisse der Berechnung mit dem Polytop Algorithmus sind in der nachfolgenden Abbildung dargestellt.

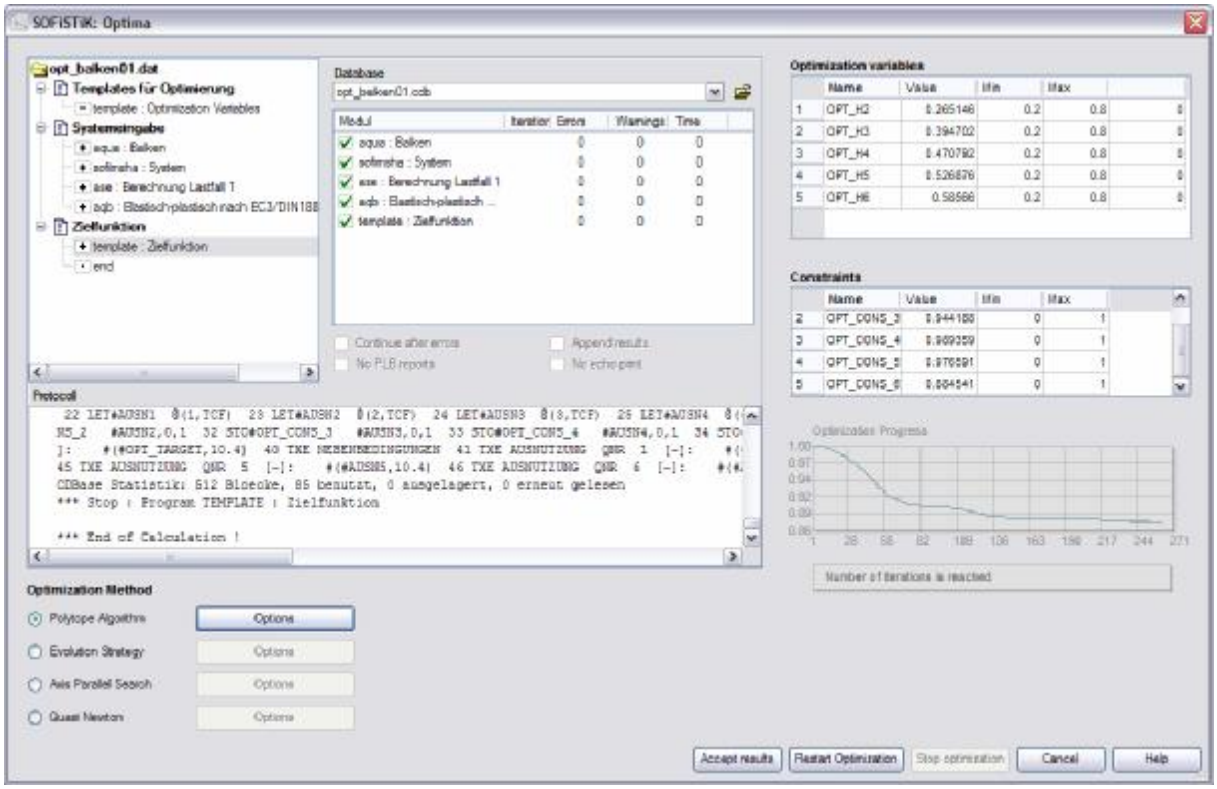


Abbildung 36: Ergebnis Polytop Algorithmus nach 2 Rechenläufen

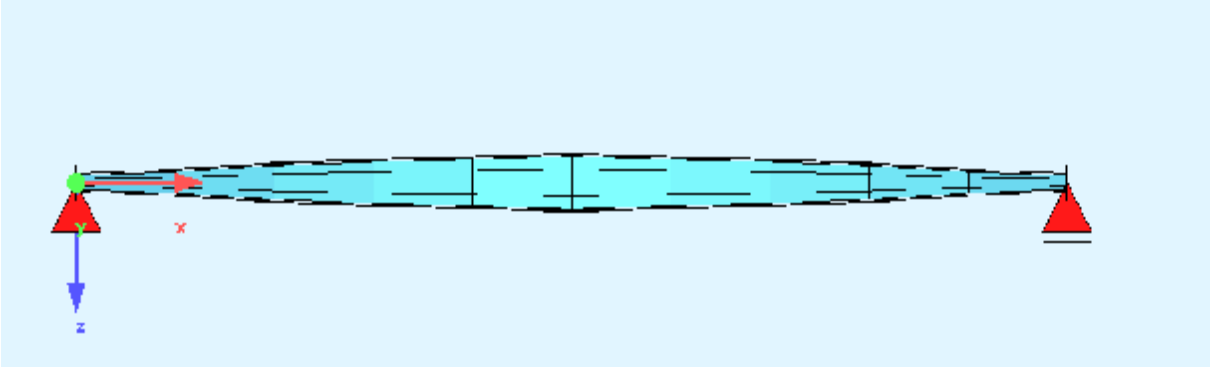


Abbildung 37: Einfeldträger nach Optimierung mit Polytop algorithmus

Da in diesem Beispiel nur die Querschnittshöhe variiert wird und damit die Gewichtsersparnis insgesamt sehr klein ist, ist die Numerik dieser Methode nicht unkritisch.

6 LITERATUR

[1] *Gill, Philip E.; Murray, Walter; Wright, Margareth H.;* Practical Optimization; Academic Press 1984

[2] *Stoer, Josef;* Einführung in die Numerische Mathematik I; Springer Verlag 1983

[3] *Rechenberg, Ingo;* Evolutionsstrategie; Frommann Verlage 1973

[4] *Quint, Marc;* Methoden der Optimierung im Bauwesen – Eine Einführung in die Form- (CAO) und Gestaltoptimierung (SKO); 17. SOFiSTiK Anwenderseminar 2004

[5] *Baier; Seeßelberg; Specht;* Optimierung in der Strukturmechanik; Vieweg Verlag 1994