Fachgebiet Prozessoptimierung

TECHNISCHE UNIVERSITÄT ILMENAU

Fakultät für Informatik und Automatisierung
Institut für Automatisierungs- und Systemtechnik

Praktikum Statische Prozessoptimierung

VERSUCH StatPO-1, Sommersemester 2020

„Nichtlineare Optimierung“

Verantwortlicher Hochschullehrer: Prof. Dr.–Ing. habil. P. Li
Versuchsverantwortliche: Dr.–Ing. S. Hopfgarten, M. Sc. B. Juris

Name,Vorname	**Matrikel-Nr.**
Mitarbeiter in der Praktikumsgruppe	
Note, Unterschrift	

1 Ziel

Das Praktikum dient der Vertiefung der Kenntnisse aus den zugehörigen Lehrveranstaltungen und veranschaulicht das Vorgehen bei der Lösung unbeschränkter nichtlinearer Optimierungsprobleme

$$\min_x f(x), \quad x \in \mathbb{R}^n, \quad f : \mathbb{R}^n \to \mathbb{R}^1$$

mit verschiedenen Methoden. Basierend auf dem Softwarepaket MATLAB®[1] gestattet es die Untersuchung der Eigenschaften von numerischen Verfahren der unbeschränkten nichtlinearen Optimierung. Diese Verfahren können an vorbereiteten Testfunktionen oder vom Nutzer definierten Optimierungsproblemen hinsichtlich ihres Aufwandes, der Konvergenzgeschwindigkeit und anderer Kriterien bewertet werden.

Es wird ein Visualisierungsprogramm mit einer Bedienoberfläche bereitgestellt, das die Darstellung von Gütegebirgen und Güteliniendiagrammen für zweidimensionale Optimierungsprobleme ($n = 2$) gestattet. Startpunkte können grafisch gesetzt oder per Zahlenwert vorgegeben werden. Suchpfade unterschiedlicher Algorithmen oder mehrerer Rechenläufe mit ein und demselben Algorithmus können verglichen werden. Die grafische Gegenüberstellung des Iterationsverlaufs erleichtert hierbei die Bewertung.

2 Realisierung des Praktikums

Die Grundlage für den Aufbau dieses Praktikums bildet das Softwarepaket MATLAB®. Dieses Softwarepaket ermöglicht wissenschaftliche und ingenieurtechnische numerische Berechnungen (Numerische Analysis, Matrizenberechnungen, Signalverarbeitung, grafische Darstellungen usw.) in einer einfach zu handhabenden Entwicklungsumgebung. Das Grund-Datenelement ist die Matrix (mit i. a. komplexen Elementen). Probleme, Ausdrücke, Algorithmen usw. können in einer an die mathematische Schreibweise angelehnten Formulierung notiert werden.

Im Rahmen dieses Praktikums stehen die folgenden ableitungsfreien und ableitungsbehafteten numerischen Verfahren der unbeschränkten nichtlinearen Optimierung zur Verfügung:

- Ableitungsbehaftete Verfahren:
  - Gradientenverfahren (Verfahren des steilsten Abstiegs),
  - Verfahren der konjugierten Gradienten nach
    FLETCHER-REEVES,
    POLAK-RIBIERE,
    HESTENES-STIEFEL
  - Quasi-Newton-Verfahren nach
    WOLFE (Rang-1-Korrektur),
    DAVIDON-FLETCHER-POWELL (Rang-2-Korrektur),
    BROYDEN-FLETCHER-GOLDFARB-SHANNO (Rang-2-Korrektur),
    jeweils mit (näherungsweise) exakter Richtungsminimierung
  - Quasi-Newton-Verfahren nach BROYDEN-FLETCHER-GOLDFARB-SHANNO mit ARMIJO-Schrittweitenregel
- Ableitungsfreie Verfahren:
  - Verfahren nach GAUSS-SEIDEL (Koordinatensuchverfahren mit Richtungsminimierung)
  - Verfahren nach HOOKE-JEEVES („pattern search“, Mustersuche)
  - Verfahren nach ROSENBROCK („rotierende Koordinaten“)

[1] MATLAB® ist eingetragenes Warenzeichen der The MathWorks, Inc.

  - Simplexverfahren nach NELDER-MEAD
- Evolutionsstrategien:
  - Eingliedrige (1+1)-Evolutionsstrategie nach SCHWEFEL [7]
  - (5/5,20)-Evolutionsstrategie nach RECHENBERG [8]
  - Geschachtelte (1,5(5/5,20))-Evolutionsstrategie nach RECHENBERG [8]
- Hybride Verfahren:
  - Hybrid aus (1,5)-Evolutionsstrategie und Verfahren nach ROSENBROCK (kombiniert durch die Methode der direkten Integration)
  - Hybrid aus (1,5)-Evolutionsstrategie und Simplexverfahren nach NELDER-MEAD (kombiniert durch die Methode der direkten Integration)

Neben diesen Optimierungsmethoden ist eine Reihe von Testproblemen hinterlegt, beispielsweise:

$$\begin{aligned}
f(x) &= \frac{1}{2}x^T Px, && P \text{ - symmetrische } (n,n)\text{-Matrix} \\
f(x) &= (x_1^2 + x_2^2 - 2x_1)^2 + \frac{1}{4}x_1 && \text{(Funktion von ZETTL)} \\
f(x) &= 100(x_2 - x_1^2)^2 + (x_1 - 1)^2 && \text{(ROSENBROCK'sches Tal)} \\
f(x) &= \sum_{i=1}^{n} x_i^{10}
\end{aligned}$$

Bezüglich ausführlicherer Angaben zu den implementierten Suchverfahren, einer vollständigen Aufstellung der Testprobleme und Hinweisen zur grafischen Bedienoberfläche sei auf den Anhang verwiesen.

3 Versuchsvorbereitung

3.1 Stellen Sie für den zweidimensionalen Fall ($x \in \mathbb{R}^2$) je eine positiv definite, indefinite sowie negativ definite quadratische Form auf! Beschreiben die das Kriterium zum Test auf die Definitheit! Stellen Sie die drei Formen jeweils als 3D-Grafik mittels MATLAB® oder octave dar!

3.2 Ermitteln Sie für die folgenden Zielfunktionen den Gradienten sowie Lage und Art der stationären Punkte:

$$\begin{aligned}
\text{a)} \quad f(x) &= 100(x_2 - x_1^2)^2 + (x_1 - 1)^2 && \text{(ROSENBROCK'sches Tal)} \\
\text{b)} \quad f(x) &= x_1 \exp\left(-x_1^2 - x_2^2\right) && \text{(Problem „Nice")} \\
\text{c)} \quad f(x) &= 2x_1^3 - 3x_1^2 - 6x_1x_2(x_1 - x_2 - 1) && \text{(Problem „Fletcher25")}
\end{aligned}$$

3.3 Wiederholen Sie theoretische Grundlagen, Arbeitsweise und wesentliche Eigenschaften ausgewählter numerischer Verfahren der unbeschränkten Optimierung (ableitungsfreie Verfahren: GAUSS-SEIDEL-Koordinatensuchverfahren, ableitungsbehaftete Verfahren: Gradientenverfahren, Konjugierte-Gradienten-Verfahren, Quasi-Newton-Verfahren)!

3.4 Im Ergebnis einer theoretischen Prozessanalyse wurde für ein gegebenes System die statische Kennlinie

$$\hat{y} = (1 - a_1 u)^{a_2 - 1}$$

mit den noch unbekannten Parametern a_1 und a_2 ermittelt.
Unter Verwendung der Messergebnisse

u_i	2	5	10	20	30	50
$\hat{y}_i$	0.9427	0.8616	0.7384	0.5362	0.3739	0.3096

sollen mit der Methode der kleinsten Fehlerquadrate Schätzwerte für a_1, a_2 bestimmt werden. Formulieren Sie eine für dieses Optimierungsproblem geeignete Zielfunktion $f(x)$ mit $x = [a_1\, a_2]^T$ und das zugehörige MATLAB® M-File (alternativ: in Octave), das für das Beispiel „Nice" folgendermaßen aussieht:

```
function f=f_nice(x)
f=x(1)*exp(-x(1)^2-x(2)^2)
```

mit x als Optimierungsvariable und f als Zielfunktion.

4 Versuchsdurchführung

Alle folgenden Untersuchungen werden mit Hilfe des MATLAB®-Programms `opt1` (Visualisierung/Bedienoberfläche durchgeführt, siehe Anhang)! Benutzen Sie zur Auswertung des Konvergenzverhaltens numerischer Verfahren die Tabelle 1 aus dem Anhang!

4.1 Lösen Sie folgende zweidimensionale quadratische Optimierungsprobleme (Zielfunktion `f_quad` bzw. Datei `Quad.mat`, Parameter P1: Hesse-Matrix) mit dem GAUSS-SEIDEL-Verfahren, dem Gradientenverahren (Steepest desc.), dem Konjugierte-Gradienten-Verfahren (nach Polak-Ribiere) und dem Quasi-Newton-Verfahren (BFGS) ausgehend von den angegebenen Startpunkten! Machen Sie sich Notizen in der Tabelle 1 des Anhangs A! Beantworten Sie anschließend die unten stehenden Fragen!

$a)\quad P1 = \begin{bmatrix} 7 & 0 \\ 0 & 7 \end{bmatrix} \qquad b)\quad P1 = \begin{bmatrix} 4 & 0 \\ 0 & 8 \end{bmatrix} \qquad c)\quad P1 = \begin{bmatrix} 4 & 3 \\ 3 & 4 \end{bmatrix}$

Vorgeschlagene Startpunkte:

$\alpha)\quad x_0 = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \qquad \beta)\quad x_0 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$

Wie wirkt sich eine Verschiebung des Startpunkts auf das Konvergenzverhalten des Gradienten-, Konjugierte Gradienten- und Quasi-Newton-Verfahrens aus?

Welchen Einfluss hat die Achsenlage der Niveaulinien bezüglich des Koordinatensystems auf das Konvergenzverhalten des GAUSS-SEIDEL-Verfahrens?

4.2 Untersuchen Sie die Arbeitsweise augewählter ableitungsfreier und ableitungsbehafteter Verfahren an folgenden einfachen nichtquadratischen Optimierungsproblemen:

a) 3.2a (ROSENBROCK'sches Tal; Zielfunktion `f_rose` bzw. Datei `Rose.mat`); Startpunkt: $[-1,1]^T$

b) Problem von ZETTL (Zielfunktion `f_zettl` bzw. Problem `Zettl.mat`); Startpunkte: $[2,0.25]^T$, $[1.2,0]^T$

Stellen Sie Vor- und Nachteile der untersuchten Verfahren zusammen und leiten Sie daraus Einsatzempfehlungen ab!

4.3 Lösen Sie das Modellbildungsproblem 3.4 mit dem Quasi-Newton-verfahren (BFGS+Armijo) Verwenden Sie den Datensatz Walsh.mat! Berechnen Sie verschiedene lokale Lösungen ausgehend von den Startpunkten $[0.4,0.5]^T$, $[-0.4,0.5]^T$ bzw. $[1,0.5]^T$ und notieren Sie sich die jeweiligen

Lösungen und Zielfunktionswerte! Welches ist das globale Minimum? Stellen Sie die identifizierten statischen Kennlinien des Systems für die drei Fälle zusammen mit den Messwerten grafisch dar. Nutzen Sie dazu die Befehlsfolge:

```
figure(100)
f_walsh(<loesung1>,'b')
hold on
f_walsh(<loesung2>,'r')
f_walsh(<loesung3>,'g')
hold off
```

mit <loesungi> - x-Vektor der i-ten Lösung, i=1,2,3, 'b', 'r', 'g' - Farbattribute

4.4 **Optional:** Testen Sie ausgewählte Optimierungsverfahren an „pathologischen" Zielfunktionen:

a) $f(x) = \sum_{i=1}^{n} x_i^{10}$ (`f_10`, Datei `F_10.mat`)

b) $f(x) = \sum_{i=1}^{n} |x_i|$ (`f_abs`, Datei `Abs.mat`)

c) Zielfunktion `f_patho` bzw. Datei `Patho.mat`.

Literatur

[1] P. Li. *Vorlesung Statische Prozessoptimierung.* TU Ilmenau

[2] *Taschenbuch Elektrotechnik.* 1. Auflage, Berlin 1977, Bd. 2; 3. Auflage, Berlin 1987 Bd. 1.

[3] R. Fletcher. *Practical Methods of Optimization. Vol. 1: Unconstrained Optimization.* Wiley, Chichester 1980.

[4] THE MATHWORKS, INC., Natick, Massachusetts: Using *MATLAB®* , 2000.

[5] THE MATHWORKS, INC., Natick, Massachusetts: *Optimization TOOLBOX for use with MATLAB®*, 2000.

[6] GNU Octave Software and Documentation. `https://www.gnu.org/software/octave/`, `https://www.gnu.org/software/octave/octave.pdf`. 2020

[7] H.-P. Schwefel. *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie.* Birkhäuser, Basel 1977.

[8] I. Rechenberg. *Evolutionsstrategie ´94.* frommann-holzboog, Stuttgart 1994.

A Anhang: Ergebnistabelle

Siehe nächste Seite.

Der Anhang B (MATLAB®-Programme) ist nicht zwangsläufig zur Durchführung des Praktikums erforderlich. Zum näheren Verständnis der Bezeichnung, des Aufbaus und des Aufrufes der Optimierungsroutinen (auch für Probleme mit mehr als 2 Optimierungsvariablen), der Zielfunktions- und Gradientprozeduren, der Beispiele von Prozeduraufrufen und der Visualisierung/Bedienoberfläche liefert der Anhang nützliche Hinweise und kann bei Bedarf herangezogen werden.

Verfahren	Start-punkt	Optimale Lösung	Opt. ZF-Wert	Anzahl Iterationen	Rechenzeit (ca.-Wert)	Anzahl ZF-Ber.	Anzahl Grad.-Ber.

Tabelle 1: Ergebnistabelle zum Konvergenzverhalten

B Anhang: MATLAB®-Programme

B.1 Optimierungsroutinen

`ovmeth`: Ableitungsbehaftete Suchverfahren mit (näherungsweise) exakter Richtungsminimierung:
- Gradientenverfahren (Verfahren des steilsten Abstiegs),
- Verfahren der Konjugierten Gradienten nach
  - FLETCHER-REEVES,
  - POLAK-RIBIERE,
  - HESTENES-STIEFEL,
- Quasi-Newton-Verfahren nach
  - WOLFE (Rang-1-Korrektur),
  - DAVIDON-FLETCHER-POWELL (Rang-2-Korrektur),
  - BROYDEN-FLETCHER-GOLDFARB-SHANNO (Rang-2-Korrektur)

`ovbfgs`: Quasi-Newton-Verfahren nach BROYDEN-FLETCHER-GOLDFARB-SHANNO mit ARMIJO-Schrittweitenregel

`ovevol`: Eingliedrige (1+1)-Evolutionsstrategie nach SCHWEFEL

`ovevol520`: (5/5,20)-Evolutionsstrategie nach RECHENBERG

`ovfmins`: Simplex-Verfahren von NELDER-MEAD, entspricht `fmins` aus der Optimization Toolbox [5]

`ovgs`: Verfahren nach GAUSS-SEIDEL (Koordinatensuchverfahren mit Richtungsminimierung)

`ovhoje`: Verfahren nach HOOKE-JEEVES („pattern search“, Mustersuche)

`ovrose`: Verfahren nach ROSENBROCK („rotierende Koordinaten“)

`oveses`: Geschachtelte (1,5(5/5,20))-Evolutionsstrategie nach RECHENBERG

`ovesrose`: Hybrides Verfahren aus (1,5)-Evolutionsstrategie und Verfahren nach ROSENBROCK (kombiniert durch die Methode der direkten Integration)

`ovesfmins`: Hybrides Verfahren aus (1,5)-Evolutionsstrategie und Simplexverfahren nach NELDER-MEAD (kombiniert durch die Methode der direkten Integration)

Die Parameterlisten der Optimierungsroutinen wurden weitgehend vereinheitlicht und entsprechen denen der MATLAB Optimization Toolbox^{TM}[2] [5]. Sie beinhalten für alle Verfahren die folgenden Übergabeparameter:

`fun`: Zielfunktionsprozedur; entweder Name eines M-Files (z. B. `f_rose`), das den Zielfunktionswert im vorgegebenen Punkt berechnet (`f=fun(x)`) oder Zielfunktion als Zeichenkette von MATLAB-Anweisungen (z. B. `'x(1)^2+2*x(2)^4'` mit der Optimierungsvariablen `x`)

`x`: Startpunkt (Spalten- oder Zeilenvektor)

`options`: Spezifikation von Abbruchschranken, Verfahrensparametern usw.
`options` ist ein Vektor der Länge 18, es ist jedoch ausreichend, Abweichungen von den jeweiligen Standardwerten (Zahlenangaben in eckigen Klammern) anzugeben, `options` wird bis zur Länge 18 ergänzt.
- `options(1)`: Ausgabesteuerung (-1: keine, 0: Standard, 1: Iterationsverlauf numerisch) [0]
- `options(2)`: Abbruchschranke Variablenänderung [1.0E-4]
- `options(3)`: Abbruchschranke Zielfunktionsänderung (und Gradientennorm für ableitungsbehaftete Verfahren) [1.0E-4]

[2] MATLAB® Optimization Toolbox^{TM} ist eine registrierte Handelsmarke der The MathWorks, Inc.

`options(4)`:	nicht verwendet
`options(5)`:	nicht verwendet
`options(6)`:	Verfahrensvariante (`ovmeth`: Berechnung der Suchrichtung, Startapproximation Hesse-Matrix) [0]
`options(7)`:	Algorithmus zur Richtungsminimierung (Schrittweitenbestimmung, verfahrensabhängig) [0]
`options(8)`:	Zielfunktionswert im Punkt `x` bei Abbruch
`options(9)`:	Test Gradientenberechnung (0: kein Test, 1: Überprüfung des mit `gradfun` berechneten Gradienten mittels Differenzenapproximation; nur für ableitungsbehaftete Verfahren) [0]
`options(10)`:	Anzahl Zielfunktionsberechnungen
`options(11)`:	Anzahl Gradientenberechnungen
`options(14)`:	maximale Iterationszahl [100]
`options(16)`:	Schrittweitenfaktor (verfahrensabhängig)
`options(17)`:	Schrittweitenfaktor (verfahrensabhängig)
`options(18)`:	Startschrittweite eindimensionale Suche [0]

`gradfun`: (nur bei ableitungsbehafteten Verfahren) Prozedur zur Berechnung des Gradienten der Zielfunktion (Spaltenvektor), siehe `fun`

`P1,...,P10`: maximal 10 Parameter (Matrizen), die an die Zielfunktions- und Gradientenberechnungsprozeduren übergeben werden. Sie dienen zur Vermeidung von globalen Variablen.

Rückgabeparameter der Optimierungsroutinen:

`x`: Lösungsvektor bzw. Wert der Optimierungsvariablen bei Abbruch der Iteration

`options`: siehe oben

`xpath`: Suchweg. Die Matrix `xpath` enthält Optimierungsvariable, Zielfunktionswert, Rechenzeit und die kumulative Anzahl der Zielfunktionsberechnungen je Iteration.

B.2 Vorbereitete Zielfunktions- und Gradientenprozeduren

Die Namen der M-Files beginnen mit `f_` (Zielfunktionsprozeduren) bzw. `df_` (Gradientenprozeduren). In Klammern ist der Dateiname für die Verwendung in der grafischen Bedienoberfläche angegeben.

`f_abs` (`Abs.mat`) Betragssumme:

$$f(x) = \sum_{i=1}^{n} |x_i|$$

`f_ackley` (`Ackley.mat`) Problem von ACKLEY:

$$f(x) = -20 \exp\left(-0.2\|x\|/\sqrt{n}\right) - \exp\left(\sum_{i=1}^{n} \cos 2\pi x_i / n\right)$$

`f_beale` (`Beale.mat`) Problem von BEALE:

$$\begin{aligned} f(x) &= (1.5 - x_1(1-x_2))^2 + \left(2.25 - x_1(1-x_2^2)\right)^2 + \\ &\quad + \left(2.625 - x_1(1-x_2^3)\right)^2 \end{aligned}$$

(`Flet23.mat`) Problem von FLETCHER, S. 23:

$$f(x) = 2x_1^3 - 3x_1^2 - 6x_1x_2(x_1 - x_2 - 1)$$

(`Flet25.mat`) Problem von FLETCHER, S. 25:

$$f(x) = 2x_1^2 + x_2^2 - 2x_1x_2 + 2x_1^3 + x_1^4$$

(`Flet59.mat`) Problem von FLETCHER, S. 59:

$$f(x) = (x_2 - 1)^2 + \left((2x_1 - 1)^2 + (2x_2 - 1)^2 - \frac{2}{3}\right)^2$$

`f_kowa` Parameterschätzproblem mit der Methode der kleinsten Quadrate ($n = 4$)

(`Kursaw.mat`) Problem von KURSAWE:

$$f(x) = \sum_{i=1}^{n} \left(|x_i|^{0.8} + 5\sin x_i^3\right)$$

`f_leon` (`Leon.mat`) Problem von LEON:

$$f(x) = 100\left(x_2 - x_1^3\right)^2 + (x_1 - 1)^2$$

`f_nice` (`F_Nice.mat`, `Nice.mat`)

$$f(x) = x_1 \exp\left(-x_1^2 - x_2^2\right)$$

`f_patho` (`Patho.mat`) nichtdifferenzierbare Zielfunktion:

$$f(x) = \frac{1}{2}\max\{|x_1|, |x_2|\} + \min\{|[x_1] - x_1|, |[x_2] - x_2|\}$$

`f_quad` (`Quad.mat`) quadratische Zielfunktion (mit Zielfunktionsparameter Matrix P):

$$f(x) = \frac{1}{2}x^T P x$$

(`Peaks.mat`)

$$\begin{aligned} f(x) &= 3(1-x_1)^2 \exp\left(-x_1^2 - (x_2+1)^2\right) - \\ &\quad -10\left(\frac{1}{5}x_1 - x_1^3 - x_2^5\right)\exp\left(-x_1^2 - x_2^2\right) - \\ &\quad -\frac{1}{3}\exp\left(-(x_1+1)^2 - x_2^2\right) + 0.1\left(x_1^2 + x_2^2\right) \end{aligned}$$

`f_rast` (`Rast.mat`) Problem von RASTRIGIN:

$$f(x) = 10 \cdot n + \sum_{i=1}^{n} \left(x_i^2 - 10 \cdot \cos(2 \cdot \pi \cdot x_i)\right)$$

`f_regler` Regelungstechnisches Beispiel (Bemessung eines PD-Reglers)

`f_rose` (`Rose.mat`) Problem von ROSENBROCK (ROSENBROCK'sches Tal, Bananenfunktion):

$$f(x) = 100\left(x_2 - x_1^2\right)^2 + (x_1 - 1)^2$$

`f_foxholes` (`Foxholes.mat`) Problem von SHEKEL (SHEKEL's foxholes):

$$\begin{aligned} \frac{1}{f(x)} &= \frac{1}{K} + \sum_{j=1}^{25} \frac{1}{c_j + \sum_{i=1}^{2}(x_i - a_{ij})^6} \\ K &= 500, \quad c_j = j \end{aligned}$$

$$(a_{ij}) = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 & -32 & \ldots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & \ldots & 32 & 32 & 32 \end{pmatrix}$$

`(Sixhump.mat)`

$$f(x) = x_1^2\left(4+x_1^2\left(-2.1+\frac{1}{3}x_1^2\right)\right)+x_1x_2+4x_2^2\left(-1+x_2^2\right)+1.0316$$

`f_walsh` `(Walsh.mat)` Modellbildungsproblem von WALSH

`f_zettl` `(Zettl.mat)` Problem von ZETTL:

$$f(x) = \left(x_1^2+x_2^2-2x_1\right)^2+\frac{1}{4}x_1$$

`f_10` `(F_10)` 10. Potenz:

$$f(x) = \sum_{i=1}^{n} x_i^{10}$$

B.3 Beispiele für Prozeduraufrufe

B.3.1 Zielfunktion

Die Optimierungsroutinen erfordern ein MATLAB M-File zur Berechnung der Zielfunktion, welches die Optimierungsvariable x als Argument übergeben bekommt und den Zielfunktionswert $f(x)$ als Ergebnis liefert.

```
function f=f_nice(x)
f=x(1)*exp(-x(1)^2-x(2)^2);
```

Der Name dieses M-Files (`f_nice`) ist beim Aufruf der Optimierungsroutine im MATLAB-Kommandofenster zu übergeben, sofern nicht die grafische Bedienoberfläche genutzt werden soll.

```
>>x0=[-1 -1]';
>>ovfmins('f_nice',x0)
```

`x0` gibt den Startpunkt für das hier verwendete Simplex-Verfahren von NELDER-MEAD vor.

Alternativ kann die Zielfunktion auch als MATLAB-Anweisung in einer Zeichenkettenvariable übergeben werden, dabei ist der Bezeichner `x` für die Optimierungsvariablen zu verwenden.

```
>>ovfmins('x(1)*exp(-x(1)^2-x(2)^2)',x0)
```

B.3.2 Gradient

Einige der implementierten Optimierungsalgorithmen verwenden den Gradienten $\nabla f(x)$ der Zielfunktion zur Bestimmung der Suchrichtung.

Die Gradientenberechnung kann in einem MATLAB M-File erfolgen, dem die Optimierungsvariablen x als Argument übergeben werden und das den n-dimensionalen Spaltenvektor $\nabla f(x)$ als Ergebnis liefert.

```
function df=df_nice(x)
df=exp(-x(1)^2-x(2)^2)*[1-2*x(1)^2; -2*x(1)*x(2)];

>>x0=[1 1]';
>>ovbfgs('f_nice',x0,[],'df_nice')
```

Alternativ kann der Gradient als MATLAB-Anweisung in einer Zeichenkettenvariablen übergeben werden. Dabei ist der Bezeichner `x` für die Optimierungsvariablen zu verwenden.

```
>>ovbfgs('x(1)*exp(-x(1)^2-x(2)^2)',x0,[],...
'exp(-x(1)^2-x(2)^2)*[1-2*x(1)^2; -2*x(1)*x(2)]')
```

Wird beim Aufruf eines ableitungsbehafteten Optimierungsalgorithmus kein Gradient übergeben, so werden die benötigten Ableitungen näherungsweise mittels finiter Differenzen berechnet.

```
>>ovbfgs('x(1)*exp(-x(1)^2-x(2)^2)',x0)
```

B.3.3 Zielfunktionsparameter

In vielen Fällen hängt die Zielfunktion neben den Optimierungsvariablen noch von weiteren Parametern ab, die selbst nicht optimiert werden sollen, deren Einfluss auf die optimale Lösung jedoch von Interesse ist.

Um in solchen Fällen die Verwendung globaler Variabler zu vermeiden, können bis zu zehn solcher Zielfunktionsparameter direkt an die Optimierungsroutinen – als zusätzliche Argumente am Ende der Parameterliste – übergeben werden.

```
function f=f_nice(x,p)
if nargin<2, p=0; end
f=x(1)*exp(-x(1)^2-x(2)^2)+p/2*(x(1)^2+x(2)^2);
```

```
>>x0=[1 1]';
>>p=0.1;
>>ovfmins('f_nice',x0,[],p)
```

Wird die Zielfunktion (oder der Gradient) als MATLAB-Anweisung in einer Zeichenkettenvariablen übergeben, so sind für die Zielfunktionsparameter die Bezeichner `P1`, `P2` usw. vorgegeben.

```
>>ovfmins('x(1)*exp(-x(1)^2-x(2)^2)+P1/2*(x(1)^2+x(2)^2),x0,[],p)'
```

B.4 Visualisierung/Bedienoberfläche

Gütegebirge und Suchpfade der Lösungsverfahren können für Optimierungsprobleme mit zwei Variablen ($n = 2$) visualisiert werden. Hierzu wird die Bedienoberfläche `opt1` bereitgestellt, die aus MATLAB heraus gestartet wird.

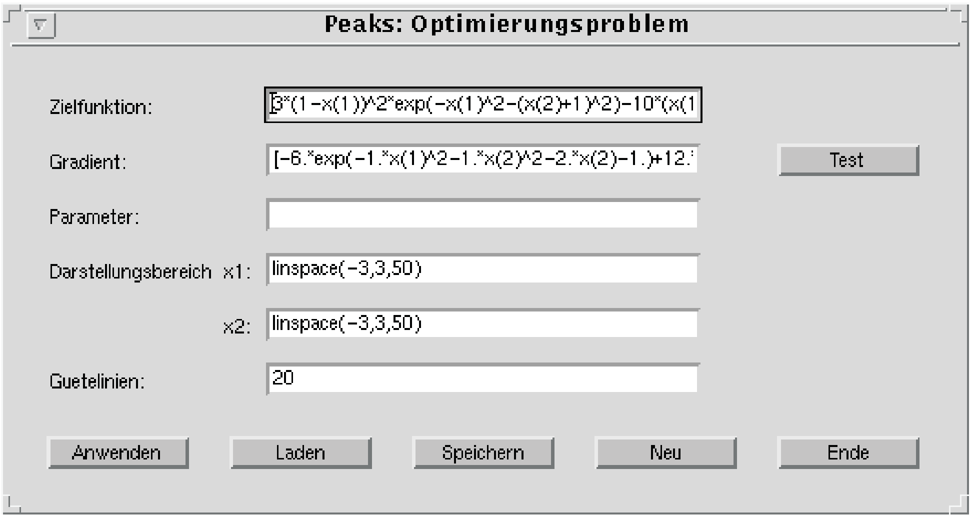
```
>>cd OPT1
>>opt1
```

Die Visualisierung/Bedienoberfläche besteht aus vier Fenstern:

- Im Fenster 'Optimierungsproblem' wird das zu untersuchende Optimierungsproblem definiert durch Angabe
  - der Zielfunktion (M-File oder MATLAB-Anweisung)
  - des Gradienten (M-File oder MATLAB-Anweisung)
  - von Zielfunktionsparametern
  - des grafischen Darstellungsbereichs (Gitterpunkte in der (x_1, x_2)-Ebene) sowie
  - der darzustellenden Gütelinien (Anzahl der Gütelinien oder Vektor von Güteniveaus).

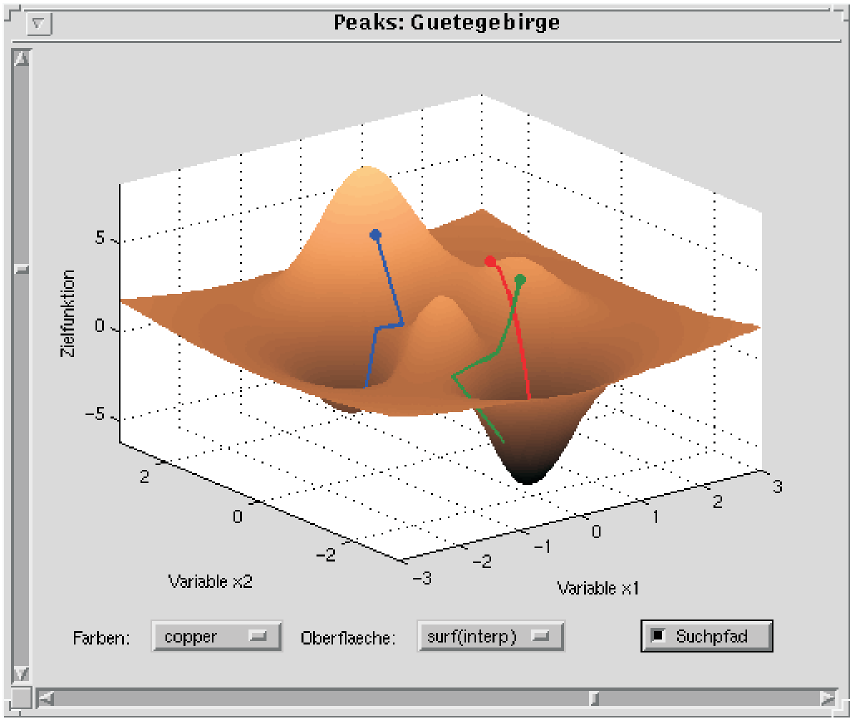
Die Gradientenberechnung kann durch Vergleich mit einer numerischen Approximation des Gradienten überprüft werden. Wird kein Gradient angegeben, so wird bei Vorgabe des Gradienten als MATLAB-Anweisung diese symbolisch differenziert, ansonsten verwenden die ableitungsbehafteten Verfahren eine näherungsweise Gradientenberechnung.

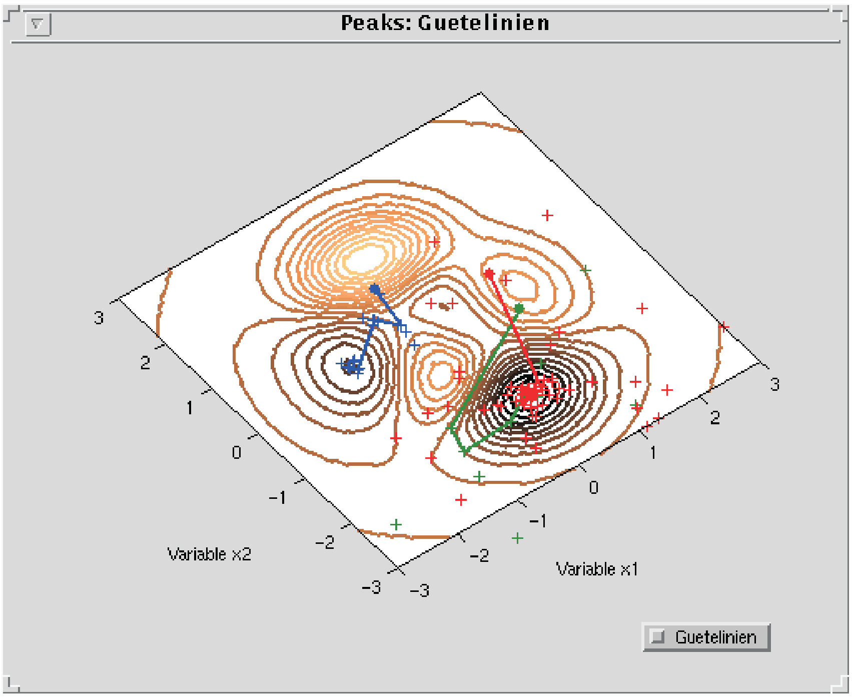
Weitere Dialogelemente gestatten die Abspeicherung bzw. das Einlesen einer vorbereiteten Problemstellung sowie das Beenden der Arbeit mit dem Programm.



- Im Fenster 'Gütegebirge' wird die Zielfunktion in Abhängigkeit von den Optimierungsvariablen (pseudo-) dreidimensional dargestellt. Die Farbpalette, die Art der Darstellung, der horizontale und der vertikale Betrachtungswinkel können mittels entsprechender Dialogelemente verändert werden. Optional können die Suchwege der Optimierungsläufe dargestellt werden.

- Im Fenster 'Gütelinien' wird das Güteliniendiagramm der Zielfunktion in Abhängigkeit von den Optimierungsvariablen sowie der Suchweg der Optimierungsläufe dargestellt.





- Im Fenster 'Optimierungslauf' können bis zu vier Optimierungsläufe (identifiziert über unterschiedliche Farben) ausgewählt werden.

  Das zu verwendende Verfahren, Optionen für den Verfahrensaufruf und der Startpunkt können ausgewählt werden. Der Startpunkt wird entweder numerisch vorgegeben oder grafisch im Fenster 'Gütelinien' gesetzt.

  Nach dem Ende eines Optimierungslaufs wird die gefundene Lösung numerisch ausgegeben und der Iterationsverlauf als Abhängigkeit des Zielfunktionswertes von der Iterationszahl, der Anzahl der Zielfunktionsberechnungen und der Rechenzeit in drei Diagrammen dargestellt.