Algorithmus Downhill-Simplex-Methode

1. Aufgabe: Finden des Minimums einer von n Koordinaten abhängigen Funktion mittels der Downhill-Simplex-Methode.

$$F(x,y) = (4\cdot(x-3)^2 + (y-7)^2)$$

$$P^0 = (0,0)$$

2. Algorithmus initialisieren

=> Konstruktion eines Start-Simplex aus n+1 Punkten X_i^0

eine Möglichkeit: beliebiger Basispunkt P^0 + Punkte die durch Basisvektoren e_i und eine Multiplikator λ bestimmt werden:

$$X_i^0 = P^0 + \lambda\cdot e_i, i = 1..n$$

=> Der Start-Simplex wird schrittweise (Index k) durch Konstruktion neuer Simplexe in Richtung des Minimums der Funktion verschoben.

3. Beschreibung des Iterationsschrittes

=> Berechnung der Funktionswerte $f\left(x_i^k\right)$

=> Bestimmung der Maximal- und Minimalwerte $f\left(x_h^k\right)$ und $f\left(x_l^k\right)$

=> Berechnung der Mitte der dem Maximalpunkt gegenüberliegenden Seite

$$x^k{}_{n+2,j} = \frac{1}{n}\left(\sum_{i=1}^{n+1} x^k{}_{ij} - x^k{}_{hj}\right), j = 1,...,n$$

j - Koordinatenrichtung

=> Spiegeln des Simplex: Projektion von x_h^k durch x_{n+2}^k mit Spiegelkoeffizient $\alpha>0$:

$$x^k_{n+3} = x^k_{n+2} + \alpha(x^k_{n+2} - x^k_h)$$

=> Dehnen des Simplex:

if $f(x^k_{n+3}) \leq f(x^k_l)$

Vektor $(x^k_{\mathbf{n+3}} - x^k_{\mathbf{n+2}})$ wird gestreckt

$$x^k_{n+4} = x^k_{n+2} + \gamma(x^k_{n+3} - x^k_{n+2})$$

if $f(x^k_{n+4}) \leq f(x^k_l)$

$x^k_{\mathbf{h}}$ wird ersetzt durch $x^k_{\mathbf{n+4}}$

Iterationsschritt beenden (k=k+1)

if $f(x^k_{n+4}) > f(x^k_l)$

$x^k_{\mathbf{h}}$ wird ersetzt durch $x^k_{\mathbf{n+3}}$

Iterationsschritt beenden (k=k+1)

=> Zusammendrücken des Simplex:

if $f(x^k_{n+3}) > f(x^k_i)$, $i \neq h$

Vektor $x^k_{\mathbf{h}} - x^k_{\mathbf{n+2}}$ wird verkürzt

$$x^k_{n+5} = x^k_{n+2} + \beta(x^k_h - x^k_{n+2})$$

$x^k_{\mathbf{h}}$ wird ersetzt durch $x^k_{\mathbf{n+5}}$

Iterationsschritt beenden (k=k+1)

=> Reduktion des Simplex

if $f(x^k_{n+3}) > f(x^k_h)$

alle Vektoren $(x^k_i - x^k_l), i = 1, ..., n+1$ werden halbiert,

der Punkt x^k_l bleibt fest

$$x^k_i = x^k_l - 0.5 \cdot (x^k_i - x^k_l), i = 1, ..., n+1$$

Iterationsschritt beenden (k=k+1)

4. Ende der Iteration

=> Abbruchbedingung

$$\frac{1}{n+1}\sum_{i=0}^{n+1}\left[f(x_i^k) - f(x_{n+2}^k)\right]^2 \le \varepsilon$$

=> da Algorithmus nicht immer erfolgreich: Abbruch nach gewisser Anzahl Iterationsschritte

5. Empfohlene Werte für Konstanten

α=1
0.4<β<0.6
2.8<γ<3.0

6. Quelle

Press, Teukolsky, Vetterling, Flannery: „Numerical Recipes in C“, Kap. 10.4

http://www.library.cornell.edu/nr/bookcpdf.html

7. Verwendung mehrdimensionaler Funktionen

```
function f=my_function(X)
f=(4.*(X(:,1)-3).^2+(X(:,2)-7).^2);
```

X-Matrix mit Dimension [n,2]

```
>> my_function([3 7])

ans =

     0
```

```
>> my_function([3 7; 3 7])

ans =

     0
     0
```

Des Weiteren sind folgende, in Matlab integrierte, Routinen empfehlenswert:

- *length(X)* … gibt die Länge des eingegebenen Vektors X als Zahl aus
- *diag(X)* … schreibt die Einträge des Vektors X (Dimension n) auf die Diagonale in eine n x n Matrix
- *ones(n,m)* … gibt eine Matrix mit n Zeilen und m Spalten aus, die mit dem Wert 1 gefüllt ist
- *sortrows(X,n)* ... sortiert die Matrix zeilenweise nach der n-ten Spalte in aufsteigender Ordnung
- *eye(n)* … erzeugt eine Einheitsmatrix der Dimension n

Bemerkung: diag(ones(1,n)) erzeugt somit eine Einheitsmatrix der Dimension n, deren Zeilen bzw. Spalten den n-dimensionalen Einheitsvektoren entsprechen, analog wie der Befehl *eye(n)*