Institut für Informatik
Dr. Tobias Neckel
Christoph Kowitz
Alfredo Parra

Technische Universität München
WS 2015/2016

Einführung in die wissenschaftliche Programmierung

Übungsblatt 10

1.) Minimierung der Rosenbrock Funktion

Die Rosenbrock Funktion $f : \mathbb{R}^2 \to \mathbb{R}$ ist gegeben durch

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

i) Implementieren Sie die Funktion `rosenbrock(x)`, die eine Liste `x = [x1,x2]` als Parameter übergeben bekommt und den Wert der Funktion an der Stelle `[x1,x2]` zurückgibt.

ii) Bestimmen Sie mit `scipy.optimize.fmin` das Minimum dieser Funktion. Verwenden Sie als Anfangswert die Koordinate `x = [0,2.5]`.

iii) Speichern Sie die Iterationsschritte mit Hilfe einer *Callback Funktion*. Definieren Sie dafür eine Funktion `store_iter(x)`, die die aktuelle Koordinate als Parameter übergeben bekommt und sie zu einer global definierten Liste `schritte` hinzufügt. Übergeben Sie `store_iter` an die Funktion `fmin` mit dem Keyword Argument `callback = store_iter`. Es wird dann `store_iter` nach jedem Iterationsschritt aufgerufen.

iv) Stellen Sie die Funktion und die Iterationsschritte mit `matplotlib` dar.

2.) Interpolation und Regression (Hands-On)

Gegeben sind die Daten

x	1	2	3	4	5	6	7
y	1	2	1.5	2	3	2.5	4

.

a.) Polynominterpolation

Finden Sie die Koeffizienten des Interpolationspolynoms p

$$p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6,$$

mit $p(x_i) = y_i$ für die Daten in der Tabelle.

i) Berechnen Sie die Koeffizienten mit Hilfe der Vandermonde Matrix. Stellen Sie hierzu das folgende Gleichungssystem auf (für $n = 6$) und lösen Sie es mit der Funktion `scipy.linalg.solve`:

$$\begin{pmatrix} x_0^n & x_0^{n-1} & x_0^{n-2} & \dots & x_0 & 1 \\ x_1^n & x_1^{n-1} & x_1^{n-2} & \dots & x_1 & 1 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ x_n^n & x_n^{n-1} & x_n^{n-2} & \dots & x_n & 1 \end{pmatrix} \begin{pmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_0 \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

ii) Benutzen Sie die Funktion `numpy.polyfit` zur Ermittlung der Polynomialkoeffizienten.

iii) Stellen Sie die Lösungen mit `matplotlib` dar. Was passiert weit außerhalb des Intervalls $[1, 7]$?

b.) Lineare Regression

Bestimmen Sie die lineare Ausgleichsgerade der Daten, d.h. bestimmen Sie Koeffizienten m und t der Funktion $f(x) = mx + t$, sodass

$$\sum_{i=0}^{6} \left(y_i - f(x_i)\right)^2, \tag{1}$$

minimal ist.

i) Lösen Sie dazu das Gleichungssystem $Bz = b$ mit $B = A^T A$, $b = A^T y$ und

$$A = \begin{pmatrix} x_0 & 1 \\ x_1 & 1 \\ \vdots & \vdots \\ x_6 & 1 \end{pmatrix}, \qquad y = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_6 \end{pmatrix}, \qquad z = \begin{pmatrix} m \\ t \end{pmatrix}.$$

Der Vektor z minimiert dann Gl. (1).

ii) Verwenden Sie nun zum Lösen alternativ die Funktion `scipy.linalg.lstsq`.

iii) Stellen Sie wieder die Lösungen graphisch dar.

3.) Parameter fitten (Hausaufgabe)

Auf der Webseite zur Vorlesung finden Sie eine Datei mit zwei Datensätzen. Die erste Spalte enthält die Werte t_i, die zweite den ersten Datensatz y_i und die dritte den zweiten Datensatz z_i. Für jeden der zwei Datensätze sollen nun Koeffizienten $p^* = (p_0, p_1, p_2, p_3) \in \mathbb{R}^4$ der Funktion

$$f(t) = p_0 \cos\left(\frac{2\pi}{p_1} t - p_2\right) + p_3 t.$$

so bestimmt werden, dass die Summe der Fehlerquadrate minimal ist, d.h. für den ersten Datensatz muss gelten

$$p^* = \arg\min_{p \in \mathbb{R}^4} \sum_{i=0}^{n} \left(f(t_i) - y_i\right)^2.$$

Verwenden Sie die Funktion `scipy.optimize.leastsq` und stellen Sie Ihr Ergebnis mit `matplotlib` dar.