Übungsblatt 11

1. (a) Sei $f : [a,b] \to \mathbb{R}$ eine stetige Funktion mit $f(a) \cdot f(b) < 0$. Wir definieren das Bisektionsverfahren, das gegen eine Lösung von $f(x) = 0$ konvergiert, als
ALGORITHMUS (BISEKTIONSVERFAHREN)
`for` $k = 1, 2, \cdots$

- `Setze`
$$x_k = a + \frac{b-a}{2}$$
- `If` $f(x_k) = 0$
`dann ist` x_k `die Lösung.`
`end if.`
- `If` $f(a) \cdot f(x_k) > 0$
$$a = x_k$$
`else`
$$b = x_k$$
`end if.`

`end for`
Zeigen Sie, dass die Folge von Intervall-Mittelpunkten $\{x_k\}_{k=1}^{\infty}$ gegen eine Nullstelle $x^* \in (a,b)$ von $f(x) = 0$ konvergiert, mit der Eigenschaft
$$|x_k - x^*| \leq \frac{b-a}{2^k}, \quad k = 1, 2, ...$$

(b) Gegeben die Funktion $f(x) = x^3 - 2x - 5$, approximieren Sie eine Lösung der Gleichung $f(x) = 0$ mit den ersten drei Iterationsschritten
i. des Bisektionsverfahrens im Intervall $[2,3]$,
ii. des Sekantenverfahrens mit $x_0 = 3$ und $x_1 = 3.5$,
iii. des Newton-Verfahrens mit $x_0 = 3$.

2. Berechnen Sie die ersten zwei Iterationsschritte des Newtonverfahrens zur Lösung des Gleichungssystems
$$F(x,y) := \begin{pmatrix} xy \\ xy^2 + x - y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

mit Startwert $(x_0, y_0) = (1/2, 1)$.

3. Es sei $f\colon \mathbb{R} \to \mathbb{R}$ dreimal stetig differenzierbar und x^* eine Nullstelle von f. Es gelte $f'(x^*) \neq 0$. Zeigen Sie, dass die Iteration

$$x_{n+1} = x_n - \frac{f(x_n)}{g(x_n)} \qquad \text{mit } g(x) = \frac{f(x + f(x)) - f(x)}{f(x)}$$

lokal quadratisch gegen x^* konvergiert.

4. Sei $F : \mathbb{R}^n \to \mathbb{R}^m$, $m \geq n$ eine hinreichend glatte Funktion. Wir betrachten das nichtlineare Ausgleichsproblem:

$$\text{minimiere} \quad \Phi(x) = \frac{1}{2}\|F(x)\|_2^2. \tag{1}$$

Wir approximieren das Minimum x^* von (1) durch eine Iterationsfolge $\{x^{(k)}\}$, die das Newton-Verfahren:

$$x^{(k+1)} = x^{(k)} - \left[F'(x^{(k)})^t F'(x^{(k)})\right]^{-1} F'(x^{(k)})^t F(x^{(k)})$$

ergibt, wobei F' die Jacobi-Matrix von F bezeichnet. Gegeben sei die Funktion

$$F : \mathbb{R} \to \mathbb{R}^2, \quad F(x) = \begin{pmatrix} x + 1 \\ \lambda x^2 + x - 1 \end{pmatrix}, \ \lambda \in \mathbb{R}.$$

Zeigen Sie, dass das Problem (1) für $\lambda < 1$ in $x = 0$ ein lokales Minimum besitzt. Für welche λ ist dies das einzige lokale Minimum?

Hinweis: Wir erinnern, dass x^* ein lokales Minimum von Φ ist, wenn die Bedingungen

$$\nabla\Phi(x^*) = 0 \ \text{ und } \ \Phi''(x^*) \ \text{ positiv definit}$$

erfüllt werden, wobei Φ'' die Hesse-Matrix ist.

5. Implementieren Sie in Matlab das Verfahren aus Aufgabe 3 zur Lösung der Gleichung $e^{-x} - \sin(x) = 0$.

6. Implementieren Sie in Matlab das Newtonverfahren zur Lösung der Gleichung $F(x) = 0$, wobei $F\colon \mathbb{R}^n \to \mathbb{R}^n$ eine beliebige (differenzierbare) Funktion ist. Dabei sollen dem Programm die Funktion F, ihre Ableitung ∇F, sowie ein Startwert x_0 übergeben werden.

Testen Sie Ihr Programm an den Funktionen:

$$f\colon \mathbb{R}^2 \to \mathbb{R}^2, \quad f(x, y) = \begin{pmatrix} 4.72\sin(2x) - 3.14e^y - 0.495 \\ 3.61\cos(3x) + \sin(y) - 0.402 \end{pmatrix},$$

$$g\colon \mathbb{R}^3 \to \mathbb{R}^3, \quad g(x, y, z) = \begin{pmatrix} x^2 + y^2 + z^2 - 1 \\ 2x^2 + y^2 - 4z \\ 3x^2 - 4y + z^2 \end{pmatrix}.$$

Hinweis: Speichern Sie in verschiedenen `m`-Files die Funktionen und die zugehörigen Ableitungen. In MATLAB haben Sie im Prinzip zwei Möglichkeiten, Funktionen als Argumente zu übergeben und dann in Ihrem Programm auszuwerten. Eine Möglichkeit ist, den Namen der Funktion (als String, also in einfachen Anführungszeichen) zu übergeben und den Befehl `feval` zur Funktionsauswertung verwenden. Alternativ können Sie auch ein *function handle* (also ein Argument der Form `@f`, wobei `f` die Funktion ist) übergeben und dann die Funktion entweder direkt (also in der Form `f(x)`) oder mittels `feval` auswerten.

Betrachten Sie beispielsweise die Funktion

```
function y = myfun(f)
y = feval(f,1);
end
```

Wenn Sie nun `myfun('sin')` aufrufen, erhalten Sie als Ergebnis sin(1) zurück. Genau dasselbe Ergebnis bekommen Sie auch mit dem Aufruf `myfun(@sin)`.

—