D-ITET
Dr. R. Käppeli
L. Rueff

Serie 10

1. *Fixpunktiteration und Konvergenz*

Wir betrachten das Nullstellenproblem

$$f(x) = xe^x - 1 = 0$$

auf dem Intervall $[0, 1]$ und die Fixpunktiterationen

$$x^{(k+1)} = \phi_i(x^{(k)}) \quad , i = 1, 2, 3$$

mit den folgenden drei Fixpunktfunktionen

- $\phi_1(x) = e^{-x}$
- $\phi_2(x) = \frac{x^2e^x+1}{e^x(x+1)}$
- $\phi_3(x) = x + 1 - xe^x$.

a) Zeigen Sie für ϕ_i ($i = 1, 2, 3$), dass das Fixpunktproblem konsistent mit dem Nullstellenproblem ist.

b) Schreiben Sie eine MATLAB Funktion

```
x = fixpunkt(phi,x0,maxitr,atol,rtol)
```

welche eine Fixpunktiteration zu gegebener Fixpunktfunktion `phi` implementiert. Weiter sind `x0` ein Startwert, d.h. $x^{(0)}$, `maxitr` die maximale Anzahl von Iterationen und `atol` und `rtol` gegebene absolute und relative Toleranz für das Abbruchkriterium (ABK3) aus der Vorlesung (Kap. 4, Seite 6):

$$\epsilon^{(k)} = |x^{(k)} - x^*| \leq \texttt{atol} + \texttt{rtol}|x^{(k)}|$$

Als Referenzlösung x^* können Sie die letzte Iteration, d.h. wenn das Abbruchkriterium erfüllt ist, des Algorithmus benutzen. Die Funktion soll im Vektor `x` alle Glieder $x^{(k)}$ zurückgeben.
Hinweis: Arbeiten Sie im Template `fixpunkt.m`.

Bitte wenden!

c) Wenden Sie die `fixpunkt.m` Funktion auf die Fixpunktfunktionen ϕ von **a)** an. Benutzen Sie $x_0 = 0$, `maxitr`$= 1000$, `atol=1e-10` und `rtol=1e-8`. Was beobachten Sie?
Hinweis: Verwenden Sie das Skript `fixpunktproblem.m`.

d) Schreiben Sie eine MATLAB-Funktion, die die Konvergenzordnung p und die Konvergenrate C obiger Fixpunktiteration ϕ_1 und ϕ_2 mit Startwert $x_0 = 1$, `maxitr`$= 1000$, `atol=1e-10` und `rtol=1e-8` berechnet.
Hinweis: Arbeiten Sie im Template `fixpunktkonvergenz.m` und verwenden Sie die in der Vorlesungen gezeigten Schätzer welche mit dem Fehler der k-ten Iteration $\epsilon^{(k)} = |x^{(k)} - x^*|$ die Konvergenzordnung und den Konvergenzrate durch

$$p = \frac{\log(\epsilon^{(k+1)}) - \log(\epsilon^{(k)})}{\log(\epsilon^{(k)}) - \log(\epsilon^{(k-1)})}, \qquad C = \frac{\epsilon^{(k)}}{(\epsilon^{(k-1)})^p}$$

bestimmen.

2. *Skalare nichtlineare Gleichung*

Man berechne die positive Lösung der Gleichung

$$e^{2x} - \sin(x) = 2 \tag{1}$$

a) mit MATLAB 's `fsolve`[1];

b) mit dem Bisektions-Verfahren;

c) mit dem Newton-Verfahren;

d) mit dem Newton-Verfahren wobei die benötigten Ableitungen durch finite Differenzen geschätzt werden;

e) mit dem Sekanten-Verfahren

$$x^{(k+1)} = x^{(k)} - \frac{x^{(k)} - x^{(k-1)}}{f(x^{(k)}) - f(x^{(k-1)})} f(x^{(k)}).$$

3. *Newton Verfahren in mehreren Dimensionen*

In mehreren Dimensionen ist das Newton Verfahren zur Lösung des Nullstellenproblems $F(x) = 0$, $x \in \mathbb{R}^d$, in Algorithmus 1 gegeben.

Hier ist `x0` der Startwert, `DF` die Jacobi-Matrix der Funktion `F`, `nMax` die maximale Anzahl von Iterationen und `ATOL` und `RTOL` gegebene absolute und relative Toleranzen für das Abbruchkriterium (ABK3) (aus der Vorlesung).

[1] `https://www.mathworks.com/help/optim/ug/fsolve.html`

Siehe nächstes Blatt!

Algorithm 1 Newton Method

function NEWTON(x0, F, DF, nMax, ATOL, RTOL)
  Initialisierung: $n = 1$, $x = x0$
  while $n < nMax$ und $\|DF(x)^{-1}F(x)\| > ATOL + RTOL\|x\|$ **do**
    $\Delta x = -DF(x)^{-1}F(x)$
    $x = x + \Delta x$
    $n = n + 1$
  return x

a) Vervollständigen Sie die MATLAB-Funktion `newton.m`, die den obigen Algorithmus implementiert.
Hinweis: Sie sollten niemals die Inverse $DF(x)^{-1}$ explizit berechnen. Stattdessen sollten Sie das System $DF(x)\Delta x = -F(x)$ lösen. (Es ist durchaus sinnvoll sich zu überlegen wieso!)

b) Betrachten Sie die Funktion $F : D = [-1, 2]^2 \to \mathbb{R}^2$ gegeben durch

$$F(x, y) = \begin{pmatrix} x^2 + y - 2 \\ ye^x - 2 \end{pmatrix}.$$

Bestimmen Sie mit Ihrer Funktion aus **a)** alle Lösungen des Nullstellenproblems $F(x, y) = 0$.
Hinweis: Die Höhenlinien der Funktions-Komponenten können mit `hoehenlinien.m` zur Illustration gezeichnet werden.

4. *Explizites und Implizites Euler-Verfahren*

Für das Anfangswertproblem $\dot{y}(t) = f(t, y(t))$ ist ein Schritt des impliziten Euler-Verfahrens durch

$$y_{n+1} = y_n + hf(t_{n+1}, y_{n+1})$$

definiert.

a) Betrachten Sie folgendes AWP

$$\dot{y}(t) = -\lambda y(t), \qquad y(t_0) = y_0.$$

Führen Sie (analytisch) einen Schritt (mit Schrittweite h) mit dem expliziten und impliziten Euler-Verfahren aus.

b) Betrachten Sie folgendes AWP

$$\dot{y}(t) = -t(y(t))^2, \qquad y(t_0) = y_0 > 0,$$

Führen Sie (analytisch) einen Schritt (mit Schrittweite h) mit dem expliziten und impliziten Euler-Verfahren aus.

Bitte wenden!

c) Lösen Sie das AWP

$$\dot{y}(t) = -20y(t), \qquad y(0) = 1,$$

jeweils mit dem expliziten und impliziten Euler-Verfahren mit Schrittweiten $h = 2^{-1}, \ldots, 2^{-8}$ und Endzeit $T = 1$. Plotten Sie die Lösung für beide Verfahren und jede Schrittweite. Was beobachten Sie?

Hinweis: Arbeiten Sie in den Templates `expEulerlinear.m`, `implEulerlinear.m` and `KonvTestEuler.m`.

Abgabe: Online bis **Freitag** den 19.05.2023 unter `sam-up.math.ethz.ch`.