Praktikum 8

Christoph Kirsch, Simon Stingelin

19.02.2024

Inhaltsverzeichnis

1 Numerische Methoden für Anfangswertprobleme

1.1 Lernziele

- Sie können das explizite und implizite Euler-Verfahren zur Lösung skalarer Anfangswertprobleme

$$\begin{aligned} y'(x) &= f(x, y(x)), \\ y(x_0) &= y_0, \end{aligned}$$

  anwenden.
- Sie verstehen den Unterschied zwischen den beiden Verfahren.
- Sie können modularen Code erstellen, um eine hohe Wiederverwendbarkeit zu erhalten.
- Sie können die Methoden systematisch testen.

1.2 Theorie

alle Referenzen aus dem Skript von Herrn Stingelin

Die Theorie zum Praktikum ist im Skript Kapitel 3.1 - 3.2.2 gegeben.

Das Praktikum ist wie folgt aufgebaut:

- Implementieren **explizites** Euler-Verfahren
  - Anwenden auf Testproblem mit bekannter analytischer Lösung
  - Kontrolle der Konvergenz (1. Ordnung vgl. Satz 3.1 im Skript)
- Implementieren **implizites** Euler-Verfahren

- Anwenden auf Testproblem mit bekannter analytischer Lösung
- Kontrolle der Konvergenz (1. Ordnung vgl. Satz 3.1 im Skript)

- Anwendung auf Anfangswertproblem mit unbekannter analytischer Lösung
  - Unterschied zwischen den beiden Verfahren

1.3 Aufträge

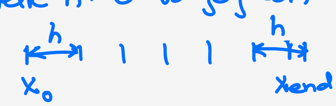
$k = f(x_{i-1}, y_{i-1})$

1. Implementieren Sie das **explizite** Euler-Verfahren Schrittindex k $y_i = y_{i-1} + hk$

$$y_{k+1} = y_k + h \cdot f(x_k, y_k)$$

(Formel 3.2 im Skript) als Funktion. Übergeben Sie die rechte Seite $f(x, y)$ der Differentialgleichung als Parameter:

Schrittweite $h > 0$ vorgegeben




```
def explizitEuler(xend, h, x0, y0, f):
    x = [x0]
    y = [y0]

    while x[-1] < xend-h/2:
        <<snipp>>
    return np.array(x), np.array(y)
```

Benutzer muss aufpassen, dass xend überhaupt erreicht wird!

oder meinen Pseudocode verwenden S.47

2. Testen Sie Ihr Programm mit dem Modellproblem

$$\begin{aligned} y'(x) &= -4\,y(x) \quad x \in [0, 2] \\ y(0) &= 1. \end{aligned} \tag{1}$$

$|y_i - y(x_i)|$

Berechnen Sie dazu die analytische Lösung und den absoluten Fehler, visualisieren Sie diesen.

3. Ein sehr guter Test für numerische Verfahren ist die Kontrolle der Konvergenzordnung. Gehen Sie dazu wie folgt vor:

```
n = 10**np.linspace(2,5)
hs = 2/n
err = []
for h in hs:
    x, y = explizitEuler(2,h,0,1,f)
    err.append(np.linalg.norm(y-ya(x),np.inf)) # ya(x) ist die exakte Lösung

plt.loglog(hs,err,'-')
plt.xlabel('h')
plt.ylabel(r'$\max_k \|e(x_k,h)\|$')
plt.grid()
plt.show()
```

nach n Schritten ist man bei $x_n = 2$

betrachte den maximalen absoluten Fehler

$\max_i |y_i - y(x_i)|$

Steigung $\hat{=}$ Konvergenzordnung

$k = f(x_{i-1} + h, y_{i-1} + hk)$ Newton

4. Implementieren Sie das **implizite** Euler-Verfahren

$y_i = y_{i-1} + hk$

nichtlineare Gleichung für y_{k+1}

$$y_{k+1} = y_k + h \cdot f(x_{k+1}, y_{k+1})$$

(Formel 3.4 im Skript) als Funktion. Übergeben Sie die rechte Seite $f(x, y)$ der Differentialgleichung sowie die partielle Ableitung $\partial_y f(x, y)$ als Parameter. Ziel ist eine möglichst modulare Implementierung. Dazu implementieren wir das Newton-Verfahren aus der letzten Woche als Funktion einer generischen Abbildung $G(s)$ deren Nullstelle gesucht wird. Die generische Abbildung $G(s)$ ist im Fall des impliziten Euler-Verfahren gegeben durch

$$G(s) = s - y_k - h \cdot f(x_{k+1}, s). \overset{!}{=} 0$$ (dann ist $s = y_{k+1}$)

für Newton benötigen wir

$$G'(s) = 1 - h \frac{\partial f}{\partial y}(x_{k+1}, s)$$

Startwert s_0, Newtonfolge

$$s_\ell = s_{\ell-1} - \frac{G(s_{\ell-1})}{G'(s_{\ell-1})} = s_{\ell-1} - \underbrace{\frac{s - y_k - h f(x_{k+1}, s)}{1 - h \frac{\partial f}{\partial y}(x_{k+1}, s)}}_{\delta s} = s_{\ell-1} - \delta s$$

Sie können sich am folgenden Template orientieren:

oder Pseudocode auf S. 48/49

```
def implizitEuler(xend, h, x0 y0, f, df):
    x = [x0]
    y = [y0]

    # Verfahrensfunktion für implizit Euler
    def G(s, xk, yk):
        return <<snipp>>

    # Partielle Ableitung nach s der Verfahrensfunktion
    def dG(s, xk, yk):
        return <<snipp>>

    def newton(s, xk, yk, tol=1e-12, maxIter=20):
        k = 0
        delta = 10*tol
        while np.abs(delta) > tol and k < maxIter:
            delta = <<snipp>>    δs
            s -= delta
            k += 1
        return s

    while x[-1] < xend-h/2:
        y.append(newton(y[-1],x[-1],y[-1]))
        x.append(x[-1]+h)
    return np.array(x), np.array(y)
```

wichtige Aufgaben

5. Testen Sie Ihr Programm wieder mit dem Modellproblem (1) und kontrollieren / visualisieren Sie die Konvergenzordnung. wie Aufgaben 2 und 3, für Euler implizit

von mir aus optional

6. Berechnen Sie mit den beiden Verfahren auf dem Intervall $[0, 2]$ numerisch Lösungen für das Anfangswertproblem

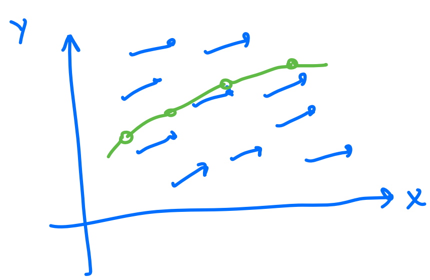
$$y'(x) + \frac{x^2}{y(x)} = 0, \quad y(0) = -4.$$

- Da die nichtlineare Differentialgleichung separabel ist, kann eine analytische Lösung berechnet werden. Berechnen Sie mit Hilfe der analytischen Lösung den absoluten Fehler für $x = 2$. Visualisieren Sie diesen in Abhängigkeit verschiedener Schrittweiten, gegeben durch die Anzahl Intervalle $N = 3^j$, $j \in \{1, 2, 3, 4, 5, 6, 7, 8\}$.

  (Zur Kontrolle $y(2) = -4\sqrt{\frac{2}{3}}$) exakter Endwert, vergleiche mit y_N : $|y_N - y(2)|$

- Visualisieren Sie Ihre numerische Lösung für $N = 3^8$ im **Richtungsfeld** der Differentialgleichung.

7. Verständnisfrage: Erklären Sie mit Hilfe des Richtungsfeldes den Unterschied zwischen dem expliziten und impliziten Euler-Verfahren.



1.4 Abgabe

Bitte geben Sie Ihre Lösungen bis spätestens vor dem nächsten Praktikum ab.

Downloads:

- PDF-Dokumentation:
  - `Anleitung Praktikum 8`