Numerik 2: Abgabe 1

Knoll Alexander

24.04.2016

Inhaltsverzeichnis

1 Aufgabe 1

1.1 Mehrschrittverfahren untersuchen

Untersuchen Sie für die folgenden Mehrschrittverfahren, ob sie mindestens von der Ordnung 2 sind (d. h. lokale Konsistenzordnung 3 besitzen) und ob sie stabil sind:

(a) $y_{i+1} = y_{i-1} + 2hf_i$

(b) $y_{i+1} = 3y_i - 2y_{i-1} + \frac{h}{12}[7f_{i+1} - 8f_i - 11f_{i-1}]$

(c) $y_{i+1} = \frac{4}{3}y_i - \frac{1}{3}y_{i-1} + \frac{h}{9}[4f_{i+1} + 4f_i - 2f_{i-1}]$

Zunächst sollen die Verfahren auf Stabilität hin untersucht. Hierfür werden characteristische Polynome aufgestellt und diese auf Nullstellen hin untersucht, wobei gilt $f(t, y) = 0$. Somit wird ersichtlich ob die Verfahren Nullstabil sind, oder nicht.

$$\begin{aligned} y_{i+1} - y_{i-1} &= 0 \\ \mu^2 - 0\mu - 1 & \\ \{\mu_1 = 1, \mu_2 = -1\} & \end{aligned}$$

Somit ist dieses Verfahren Nullstabil.

$$\begin{aligned} y_{i+1} - 3y_i + 2y_{i-1} &= 0 \\ \mu^2 - 3\mu + 2 & \\ \{\mu_1 = 1, \mu_2 = 2\} & \end{aligned}$$

Somit ist dieses Verfahren nicht Nullstabil.

$$\begin{aligned} y_{i+1} - \frac{4}{3}y_i + \frac{1}{3}y_{i-1} & \\ \mu^2 - \frac{4}{3}\mu + \frac{1}{3} & \\ \{\mu_1 = \frac{1}{3}, \mu_2 = 1\} & \end{aligned}$$

Auch hier liegen die Lösungen im interval $[-1, 1]$. Dies bedeutet dass auch dieses Verfahren Nullstabil ist.

Taylor-Entwicklung

Durch Taylor-Entwicklung ist es möglich die Konsistenzordnung der jeweiligen verfahren zu bestimmen.

$$\begin{aligned} T(y(t_{i+1})) - T(y_{i+1}) &= O(h^{p+1}) \\ &= \frac{1}{3}h^3 y'''(t) \end{aligned}$$

Somit folgt dass dieses Verfahren nicht konsistent ist (konsistenz ordnung 0).

$$\begin{aligned} T(y(t_{i+1})) - T(y_{i+1}) &= O(h^{p+1}) \\ y + hy' + \frac{1}{2}h^2 y'' + O(h^3) = 3y - 2(y - hy' + \frac{1}{2}h^2 + O(h^2)) + \frac{h}{12}[7T(f_{i_+1}) - 8y' - 11T(f_{i-1})] \\ &= \frac{1}{6}h^3 y''' + 3O(h^3) + \frac{1}{3}O(h^4) \end{aligned}$$

Somit hat dieses Verfahren mindestens Konsistenzordnung 3. Abschließend wird dass letzte Verfahren untersucht

$$\begin{aligned} T(y(t_{i+1})) - T(y_{i+1}) &= O(h^{p+1}) \\ y + hy' + \frac{1}{2}h^2 y'' + O(h^3) = \frac{4}{3}y - \frac{1}{3}(y - hy' + \frac{1}{2}h^2 + O(h^2)) + \frac{h}{9}[4T(f_{i+1}) - 4y' - 2T(f_{i-1})] \\ &= \frac{h^3}{9}y''' - O(h^3) + \frac{2}{9}O(h^4) \end{aligned}$$

Somit ist auch gezeigt, dass dieses Verfahren Konsistenzordnung 3 besitzt.

1.2 Verfahren implementieren

Implementieren Sie das erste der drei Verfahren sowie das Verfahren

$$y_{i+1} = \frac{4}{3}y_i - \frac{1}{3}y_{i-1} + \frac{2h}{3}f_{i+1}$$

und verwenden Sie bei letzerem ein geeignetes Verfahren zum Lösen des nichtlinearen Gleichungssystems.

Erstes Verfahren

Für das erste Verfahren ist es recht einfach einen algorithmus zu implementieren. Python selbst verwendet dynamische Datentypen und ermöglicht somit die Übergabe von Funktionen (bzw. Methoden) als Parameter. Definiert wird eine routine $explicit_m iddle$, welche als argumente dass zu verwendende interval, die Schrittweite h, sowohl y' als funktion $f(t, y)$ und einen anfangswert $y(0)$

bekommt. Eine mögliche implementierung könnte wie folgt aussehen

```
1       def explicit_middle(interval, f, h, y_0):
2
3         length = len(interval)
4         ys = [None for _ in range(length)]
5         ys[0] = y_0
6
7         # anlaufrechnung mit dem expliziten euler verfahren
8         ys[1] = y_0 + h * f(interval[0], y_0)
9
10        for i in range(1, length-1, 1):
11            ys[i+1] = ys[i-1] + 2*h*f(interval[i], ys[i])
12
13        return ys
```

Zurückgegeben werden die y-werte an den stützstellen des übergebenen intervals in form eines Arrays.

Zweites Verfahren

Das zweite Verfahren lässt sich bereits nicht so einfach umsetzen wie das vorherige. Da auf beiden Seiten der Gleichung der Term y_{i+1} vorkommt ist es notwendig ein weiteres lösungsverfahren, wie zum Beispiel das Newton- oder dass Sekanten-Verfahren, einzubeziehen. Die routine nimmt die selben argumente wie das vorherige Verfahren auch. Als zusätzliches lösungsverfahren wurde in der folgenden implementierung ein einfaches Sekanten-Verfahren gewählt.

```
1       def implicit_bdf(interval, f, h, y_0):
2
3           length = len(interval)
4           ys = [None for _ in range(length)]
5           ys[0] = y_0
6           # anlaufrechnung mit dem expliziten euler verfahren
7           ys[1] = y_0 + h * f(interval[0], y_0)
8
9           for i in range(1, length-1, 1):
10              U = (ys[i-1], ys[i])
11              func2 = lambda u: u - 4/3 * ys[i] + 1/3 * ys[i-1]
12                              -  h * 2/3 * f(interval[i+1], u)
13
14              # anwenden des sekanten verfahrens
15              ys[i+1] = U[1] - (U[1]-U[0])/(func2(U[1])
16                              - func2(U[0]))*func2(U[1])
17
18          return ys
```

Durch iteratives anwenden des Sekantenverfahrens könnte die Genauigkeit weiter erhöt werden, erkauft wird diese Genauigkeit jedoch mit einem höheren Risiko für Auslöschung. Wieder werden die y-werte in einem array zurückgegeben.

1.3 Numerische Lösung

Berechnen Sie (mit Python) numerische Lösungen mit $h = 1; 10^{-1}; 10^{-2}, ...$ ausgehend von den exakten Startwerten für $y(0)$ und $y(h)$ für den Wert $y(10)$ und bestimmen Sie jeweils die Fehler. Bestimmen Sie daraus auch die Fehlerordnung und vergleichen Sie sie mit Ihrem theoretischen Ergebnis.

Um die verfahren tatsächlich auszuwerten wird eine Testgleichung benötigt. Mithilfe dieser bestimmen wir den exakten Wert für $y(10)$ und vergleichen diesen mit den numerischen. Dabei werden die Schrittweiten $h = 1; 10^{-1}; 10^{-2}; 10^{-3}, 10^{-4}$ und 10^{-5} verwendet. Die betrachtete Differentialgleichung ist hierbei $\dot{y} = -2y(t) + 1$, mit dem anfangsbedingung $y(0) = 1$. Die Auswertung des Verfahresfehlers an der Stelle $t = 10$ liefert folgende Tabelle (das Erste Verfahren wird mit A bezeichnet, dass Zweite mit B). Mithilfe von wolfram alpha wird der exakte Wert der Lösung auf $y(10) = 0.5000000010305768$ festgelegt

h	**A**	**B**
1	-257114.49	9.70059897618114e-05
10^{-1}	-2066217.22	3.1363700525588456e-10
10^{-2}	-24218.67	3.101852108500225e-12
10^{-3}	-242.57	4.929390229335695e-14
10^{-4}	-2.42	2.0106138975961585e-13
10^{-5}	-0.024257224873262873	2.679634292235278e-12

Aus obiger Tabelle wird ersichtlich wie ungeeignet Verfahren A zur Lösung des gestellten Problems ist. Es besteht immer die möglichkeit dass sich fehler in der Umsetzung des numerischen Verfahrens einschleichen, wenn es jedoch um die Wahl der Methode geht, so ist liegt man in diesem Fall mit Verfahren B auf der sicheren Seite. Durch die Theoretischen überlegungen war bereits klar dass das erste Verfahren nicht gegen die Lösung konvergieren wird, da es nicht konsistent ist.

1.4 Anfangswertproblem untersuchen

Gegeben ist das Anfangswertproblem

$$\dot{y} = -2y(t) + 1, y(0) = 1 \tag{1}$$

Bestimmen sie die exakte Lösung des AWPs und vergleichen Sie sie mit der numerischen.

1.4.1 Analytische Lösung

Zunächste muss Gleichung (1) in Standart Notation gebracht werden, welche gegeben ist durch:

$$\dot{y} + p(t) = Q(t)$$

Somit kann Gleichung (1) überführt werden in

$$\dot{y} + 2y = 1 \tag{2}$$

Eine allgemeine lösung zu einer Gleichung in Standart Notation kann mithilfe folgender ausdrücke bestimmt werden

$$u(x) = e^{\int p(x)dt}$$
$$\frac{d}{dt}(uy) = Q(t)$$

Angewandt auf Gleichung (2) erhält man einen Ausdruck welcher durch beidseitiges integrieren und weiterem umformen, die allgemeine Lösung des Problems beschreibt.

$$\begin{aligned} u &= e^{\int 2dt} \\ \frac{d}{dt}(e^{2t}y) &= e^{2t} && | \int \\ \int \frac{d}{dt}(e^{2t}y) &= \int e^{2t} \\ e^{2t}y &= \frac{1}{2}e^{2t} + C && | \div e^{2t} \\ y &= \frac{1}{2} + \frac{C}{e^{2t}} \\ y &= \frac{1}{2} + Ce^{-2t} \end{aligned}$$

Einbeziehen der Anfangsbedingung $y(0) = 1$ liefert die exakte Lösung des Problems

$$\begin{aligned} y(0) &= \frac{1}{2} + C * e^{-2*0} \\ 1 &= \frac{1}{2} + C && | - \frac{1}{2} \\ \frac{1}{2} &= C \end{aligned}$$

Und somit die exakte Lösung

$$y(t) = \frac{1}{2}e^{-2t} + \frac{1}{2} \tag{3}$$

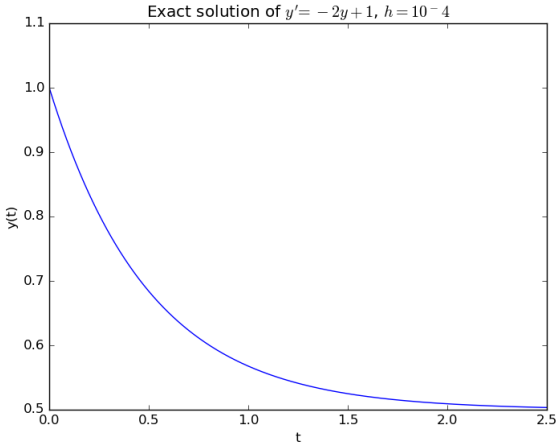

Figure 1: Analytische lösung im Intervall [0; 2,5]

Wenn nun Störungungen in die Anfangsbedingung eingeführt werden, kann man das Problem auf stabilität untersuchen. Hierfür wird ,ausgehend von der allgemeinen lösung, eine exakte Lösung unter verwendung von $y(0) = 1 + \epsilon$ bestimmt.

$$y_\epsilon(t) = (\frac{1}{2} + \epsilon)e^{-2t} + \frac{1}{2}$$

Es ist leicht zu sehen dass für sehr kleine ϵ, die lösung keine größeren abweichungen erfährt. Somit kann die Lösung als Stabil bezeichnet werden.

1.4.2 Numerische Lösung

Als folge der stabilität des AWPs, ist es sehr wahrscheinlich dass Einschrittverfahren für ausreichend kleine h gegen die Lösung des Problems konvergieren. Deshalb soll im folgend der vergleich dreier Verfahren, die Schrittweite $h = 0,5$ genutzt werden. Dies liefert eine bessere visuelle darstellung der tatsächlichen unterschiede der genutzen Verfahren. Genutzt werden dass implizite und explizite Euler-Verfahren als auch dass Adams-Moultan-Verfahren (Ordnung 2).

Expliziter Euler

Das Verfahren wird beschrieben durch

$$u_{i+1} = u_i + hf_i$$

angewandt auf Gleichung (1) erhält man

$$\begin{aligned} y_{i+1} &= y_i + h(-2y_i + 1) \\ y_{i+1} &= (1 - 2h)y_i + h \end{aligned}$$

Impliziter Euler

Das implizite verfahren lautet wie folgt.

$$u_{i+1} = u_i + hf_{i+1}$$

Da $y' = f(t, y)$ bekannt ist es möglich dass verfahren durch umformungen auf eine einfach auswertbare Form zu bringen. Andernfalls wäre es notwendig in jedem iterations-schritt ein passendes Lösungsverfahren anzuwenden, wie beispielsweise dass Newton-Verfahren oder dass Sekanten-Verfahren.

$$\begin{aligned} y_{i+1} &= y_i + h(-2y_{i+1} + 1) & \\ y_{i+1} &= y_i - 2hy_{i+1} + h & \mid + 2hy_{i+1} \\ (1 + 2h)y_{i+1} &= y_i + h & \mid \div (1 + 2h) \\ y_{i+1} &= \frac{y_i + h}{1 + 2h} & \end{aligned}$$

Adams-Moultan

Dieses Verfahren existieren in verschiedenen Variationen. In diesem Fall wird die Methode der 2. Ordnung verwendet.

$$u_{i+1} = u_i + \frac{h}{2}[f_{i+1} + f_i]$$

Wie bereits beim implizieten Euler Verfahren, ist es möglich die gleichung in eine einfach auswertbare form zu bringen.

$$
\begin{aligned}
y_{i+1} = y_i + \frac{h}{2}[-2y_{i+1} + 1 - 2y_{i+1} + 1] & \\
y_{i+1} = y_i - hy_{i+1} - hy_i + h & \quad | + hy_{i+1} \\
(1+h)y_{i+1} = (1-h)y_i + h & \quad | \div (1+h) \\
y_{i+1} = \frac{(1-h)y_i + h}{1+h} &
\end{aligned}
$$

Vergleich

Unter einbezug von Python werden die resultate visualisiert.

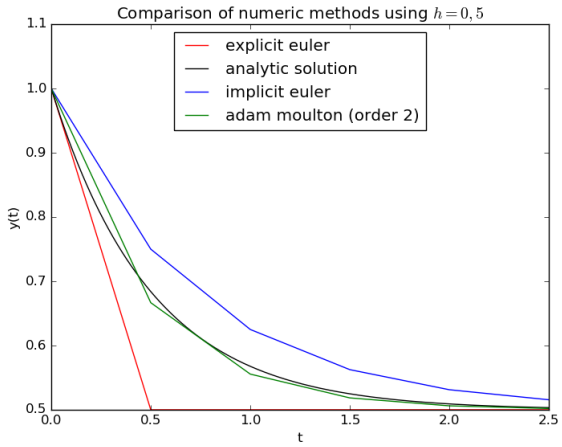

Figure 2: Vergleich der Numerischen Lösungen mit der Analytischen

Die Grafik zeigt deutlich, dass jedes der angewandten Verfahren letztendlich gegen die Lösung konvergiert. Jedoch konvergieren die Einschritt-Verfahren wesentlich langsamer als das Adams-Moultan-Verfahren, welches selbst für recht große h einen guten eindruck macht. Weiterhin scheint es so als würde dass expliziete Euler-Verfahre direkt nach dem ersten iterations schritt auf 0 fallen, wodurch es unbrauchbar wird. Abhilfe schafft eine kleinere Wahl von h.

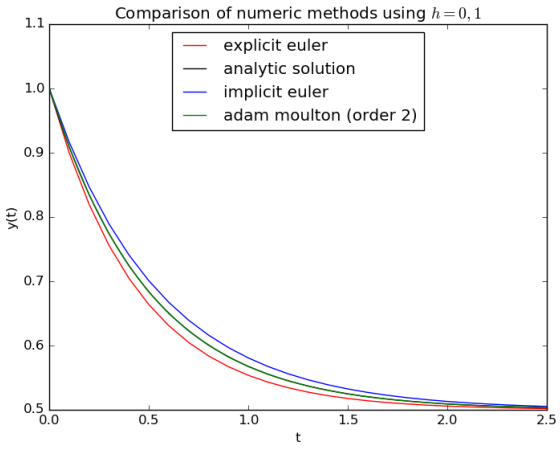

Figure 3: Numerische Lösungen mit kleinerem h

In der obigen Abbildung wird deutlich, wie gut das Adams-Moultan-Verfahren für das AWP geeignet ist.

1.5 Stellungnahme

KnollAlexamder : Mein Teil an der Umsetzung war die Bearbeitung der kompletten Zweiten Aufgabe, als auch der dokumentation in TeX. Teil der Umsetzung war die Implementierung der Python files "two.py" und "four.py".

Literaturverzeichnis

[http://www2.mathematik.hu-berlin.de/ gaggle/W0506/MATHINF/HANDOUT/handout4-4n.pdf]

[http://math-www.uni-paderborn.de/ walter/teachingSS04/VortragThema5.pdf]