Institut für Numerische und Angewandte Mathematik
FB Mathematik und Informatik der Universität Münster
Prof. Dr. Christian Engwer, MSc. Marcel Koch

23.11.2017

Übung zur Vorlesung

Wissenschaftliches Rechnen

WS 2017/18 — Blatt 7

Abgabe: 30.11.2017, 10:00 Uhr, Briefkasten 111
Code zusätzlich per e-mail an `marcel.koch@uni-muenster.de`

Aufgabe 1 (Turing-Instabilitäten) (6 Punkte)

Zwei wechselwirkende Substanzen mit Konzentrationen $a(x,t)$, $b(x,t)$ können durch ein Reaktions-Diffusions-System beschrieben werden, welches Sie in der Vorlesung als Turing-Modell kennengelernt haben. Wir betrachten den entdimensionalisierten Fall dieses Systems, d.h. das Anfangsrandwertproblem

$$\left.\begin{aligned} \partial_t a &= \Delta a + \gamma f(a.b) \\ \partial_t b &= d\Delta b + \gamma g(a,b) \end{aligned}\right\} \quad \text{in} \quad \Omega \subset \mathbb{R}^d \times [0,T] \tag{1a}$$

mit Neumann-Randbedingungen

$$\nabla a \cdot n = 0, \quad \nabla b \cdot n = 0 \qquad \text{auf} \quad \partial\Omega \times [0,T] \tag{1b}$$

und Anfangsbedingungen

$$a(\cdot,0) = a^0, \quad b(\cdot,0) = b^0 \qquad \text{in} \quad \Omega. \tag{1c}$$

Die Konstante $\gamma > 0$ gibt die relative Stärke der Reaktionsterme im Vergleich zu den Diffusionstermen wieder. Die Diffusionskonstante d sei positiv.

Wir wollen die Turing-Instabilitäten des Systems (1) untersuchen. Wir suchen also stationäre Lösungen (a_0, b_0) von (1), die instabil für das volle System sind, obwohl sie eine stabile stationäre Lösung des diffusionslosen Systems

$$\partial_t a = \gamma f(a,b), \quad \partial_t b = \gamma g(a,b) \qquad \text{in} \quad \Omega \times [0,T] \tag{2}$$

mit den Anfangsbedingungen (1c), sind.

(a) Zunächst linearisieren Sie das System (2) um eine stationäre Lösung (a_0, b_0), so dass sich ein Differentialgleichungssystem $\partial_t w = Aw$ mit $A = (a_{ij}) \in \mathbb{R}^{2\times 2}$ ergibt. Geben Sie hinreichende Bedingungen an f und g an, damit eine stationäre Lösung stabil ist.

(b) Betrachten Sie nun das Schnakenberg-System mit

$$f(a,b) := \alpha - a + a^2 b,$$
$$g(a,b) := \beta - a^2 b,$$

und $\alpha, \beta \in \mathbb{R}$. Bestimmen Sie die positiven, stationären Lösungen (a_0, b_0), die homogen im Raum sind.

(c) Um eine Turing-Instabilität zu erhalten, muss noch die Linearisierung von (1) um (a_0, b_0) auf Instabilität untersucht. Im Falle $d \neq 1$ führt dies zu den zusätzlichen Bedingungen

$$d\,a_{11} + a_{22} > 0, \quad (d\,a_{11} + a_{22})^2 - 4d\,(a_{11}a_{22} - a_{21}a_{12}) > 0.$$

Leiten Sie die notwendigen Bedingungen für α und β her, damit im Schnakenberg-System eine Turing-Instabilität auftritt.

Zum approximativen Lösen von Anfangswertproblemen haben Sie in der Übung bislang nur *explizite Verfahren* kennengelernt. Im Gegensatz zu expliziten Verfahren kommt die Approximation $y^{k+1} \approx y(t^{k+1})$ bei *impliziten Verfahren* auch auf der rechten Seite der Iterationsvorschrift vor. Das einfachste Beispiel eines solchen Verfahrens ist das implizite Euler-Verfahren.

Definition 1 (Implizites Euler-Verfahren)

Gegeben sei ein Anfangswertproblem (AWP) 1. Ordnung

$$\begin{aligned} y' &= f(t,y), \qquad \text{auf} \quad I := [t^0, T], \qquad t^0, T \in \mathbb{R}_0^+, \\ y(t^0) &= y^0, \qquad\qquad\qquad\qquad\qquad\quad y^0 \in \mathbb{R}^n, \end{aligned}$$

mit $f : I \times \mathbb{R}^n \to \mathbb{R}^n$ stetig, $n \in \mathbb{N}$. Auf I sei $I_{\Delta t} := \{t^k = t^0 + k\Delta t \mid k = 0, 1, 2, \ldots \ \wedge \ t^k \leq T\}$ ein gewähltes Gitter mit zugehöriger Schrittweite $\Delta t \in \mathbb{R}^+$. Dann liefert die Iterationsvorschrift

$$y^{k+1} = y^k + \Delta t f(t^{k+1}, y^{k+1}), \quad k = 0, 1, 2, \ldots$$

Approximationen $y^k \approx y(t^k)$ der Lösung des AWPs. Für autonome gewöhnliche Differentialgleichungen $y' = f(y)$ mit $f : \mathbb{R}^n \to \mathbb{R}^n$ reduziert sich die Iterationsvorschrift analog zum expliziten Euler-Verfahren, siehe Blatt 1.

Das explizite und das implizite Euler-Verfahren lassen sich als Spezialfälle des folgenden Verfahrens auffassen.

Definition 2 (Theta-Verfahren)

Gegeben sei ein AWP 1. Ordnung

$$\begin{aligned} y' &= f(t,y), \qquad \text{auf} \quad I := [t^0, T], \qquad t^0, T \in \mathbb{R}_0^+, \\ y(t^0) &= y^0, \qquad\qquad\qquad\qquad\qquad\quad y^0 \in \mathbb{R}^n, \end{aligned}$$

mit $f : I \times \mathbb{R}^n \to \mathbb{R}^n$ stetig, $n \in \mathbb{N}$. Auf I sei $I_{\Delta t} := \{t^k = t^0 + k\Delta t \mid k = 0, 1, 2, \ldots \ \wedge \ t^k \leq T\}$ ein gewähltes Gitter mit zugehöriger Schrittweite $\Delta t \in \mathbb{R}^+$. Dann liefert die Iterationsvorschrift

$$y^{k+1} = y^k + \Delta t\Big((1-\theta) f(t^k, y^k) + \theta f(t^{k+1}, y^{k+1})\Big), \quad k = 0, 1, 2, \ldots$$

für festes $\theta \in [0, 1]$ Approximationen $y^k \approx y(t^k)$ der Lösung des AWPs. Für autonome gewöhnliche Differentialgleichungen gilt das zum impliziten Euler-Verfahren gesagte.

Das explizite Euler-Verfahren ($\theta = 0$) ist das einzige explizite Theta-Verfahren. Für $\theta \neq 0$ ergeben sich implizite Verfahren. Die Wahl $\theta = 1$ liefert das implizite Euler-Verfahren und $\theta = 0.5$ das implizite Analogon zum Heun-Verfahren von Blatt 2, die sogenannte *Crank-Nicolson-Methode*.

Bemerkung 1 (Gleichungslöser)

Zur Anwendung eines impliziten Verfahrens sollen Sie in jedem Zeitschritt ihren Newton-Solver von Blatt 6 verwenden. Dazu wird das Lösen eines Gleichungssystems äquivalent formuliert als die Bestimmung der Nullstelle einer Funktion $r^k : \mathbb{R}^n \to \mathbb{R}^n$. Beim impliziten Euler-Verfahren z.B. ist $r^k(z) := z - y^k - \Delta t \cdot f(t^k + \Delta t, z)$.

Aufgabe 2 (Theta-Verfahren) (4 Punkte)

Implementieren Sie in C++ ein Klassentemplate `ThetaScheme` zur Durchführung des Theta-Verfahrens für den allgemeinen Fall eines nicht-autonomen AWPs 1. Ordnung.

- Das Verfahren soll für verschiedene Datentypen anwendbar sein, mit denen sich Vektoren in $\mathbb{R}^n$, Matrizen in $\mathbb{R}^{n\times n}$ und Zeiten in $\mathbb{R}_0^+$ repräsentieren lassen. Zu diesem Zweck soll `ThetaScheme` drei Template-Parameter `VectorType`, `MatrixType` und `TimeType` besitzen. Stellen Sie an `VectorType` und `MatrixType` die gleichen Vorraussetzungen wie an `Vector` und `Matrix` aus Blatt 6, Aufgabe 3.

- Der Konstruktor des Klassentemplates soll einen Gleichungslöser als Objekt einer Klasse erhalten, die das Interface `Solver` aus Bemerkung 1 erfüllt. Ferner soll er die Funktion $f : I \times \mathbb{R}^n \to \mathbb{R}^n$ als Objekt einer Klasse erhalten, die folgendes Interface für zeitabhängige, örtlich differenzierbare Funktionen geeignet erfüllt:

```
template <typename Domain, typename Range, typename JacobianRange,
          typename Time = double>
class DifferentiableTimeFunction
{
public:
  using DomainType = Domain;
  using RangeType = Range;
  using TimeType = Time;
  using JacobianRangeType = JacobianRange;

  virtual RangeType operator()(const TimeType& t, const DomainType& x) const = 0;

  virtual JacobianRangeType evaluateJacobian (const TimeType& t,
                const DomainType& x) const = 0;
};
```

  Auf der Vorlesungshomepage steht für Sie eine ausführlichere, kommentierte Version aller Interfaces zum Download bereit.

- Mit der Methode

```
void apply (const TimeType& t, const TimeType& dt,
            const VectorType& y_old, VectorType& y_new) const
```

  sollen die durch die Iterationsvorschrift festgelegten Schritte des Verfahrens ausgeführt werden können. Dabei bezeichnen `t` sowie `dt` den Zeitpunkt t_k sowie die Schrittweite Δt des aktuellen Zeitschritts und `y_old` sowie `y_new` entsprechen den Approximationen y_k und y_{k+1} zum alten und neuen Zeitpunkt.

- Überlegen Sie sich, wie r^k beim Theta-Verfahren aussieht und wie die Jakobi-Matrix von r^k mit der Jakobi-Matrix von f zusammenhängt.
- Verwenden Sie zur Bestimmung der Approximation y_{k+1} zum neuen Zeitpunkt die Approximation y_k zum alten Zeitpunkt als initial guess für den Gleichungslöser.
- Im Falle $\theta = 0$ soll aus Effizienzgründen der Gleichungslöser ignoriert werden und das explizite Euler-Verfahren direkt durchgeführt werden.

Aufgabe 3 (Test der Implementierung) (4 Punkte)

Testen Sie Ihre Implementierung des Theta-Verfahrens aus Aufgabe 2 an folgendem System, welches den diffusionslosen Spezialfall des Schnakenberg-Modells darstellt, das Sie in der Vorlesung kennengelernt haben:

$$\begin{aligned} a' &= c_a - r_a a + s a^2 b \\ b' &= c_b - s a^2 b. \end{aligned}$$

(a) Wie sehen in diesem Fall die Funktion y und die rechte Seite f in Definition 2 aus? Was ist die Jacobimatrix Df der rechten Seite?

(b) Leiten Sie vom Interface `DifferentiableTimeFunction` eine entsprechende Klasse für die rechte Seite ab.

(c) Plotten Sie die zeitliche Entwicklung der Konzentrationen a und b einmal für das explizite und einmal für das implizite Euler-Verfahren. Verwenden Sie dabei testweise die Parameter $s = r_b = c_a = c_b = 1, r_a = 2$, die Anfangswerte $a(0) = 0.5$ sowie $b(0) = 5$, die maximale Zeit $T = 50$ und die Schrittweite $\Delta t = 0.001$. Benutzen Sie als Gleichungssystemslöser für das implizite Euler-Verfahren Ihre Implementierung des Newton-Verfahrens aus Blatt 6, Aufgabe 3.