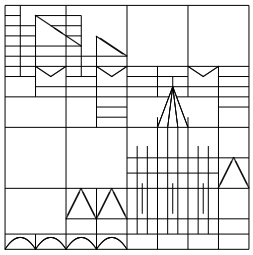
Universität Konstanz
Fachbereich Mathematik und Statistik
Prof. Dr. Stefan Volkwein
Martin Gubisch, Sabrina Rogg
Wintersemester 2015/2016



Ausgabe: Freitag, 13.10.2015
Abgabe: Freitag, 20.11.2015, 10:00 Uhr, Büro G413 bzw. per Email

POD für linear-quadratische Optimalsteuerung
2. Übungsblatt – Programmierteil

Schreiben Sie einen MATLAB-Löser für die Evolutionsgleichung

$$\begin{aligned} \dot{z}(t;x,y) - \sigma\Delta z(t;x,y) &= f(t;x,y) && \text{in } \Theta\times\Omega,\\ z(t;x,y) &= 0 && \text{in } \Theta\times\partial\Omega,\\ z(0;x,y) &= z_0(x,y) && \text{in } \Omega, \end{aligned}$$

wobei $\Omega = [a_x, b_x]\times[a_y, b_y] \subseteq \mathbb{R}^2$ und $\Theta = (0,T) \subseteq \mathbb{R}$.

Die Funktion soll wie folgt aufgerufen werden:

```
data = SolverPde(data).
```

`data` ist eine Struktur, die anfangs nur aus dem Feld `input` besteht; die Funktion `SolverPde` fügt ihr das Feld `output` hinzu. `input` wiederum besteht aus den Feldern `x`, `y`, `t`, `sigma`, `z0`, `f`, `method`, `assem`, `solve` und `output` aus den Feldern `Phi`, `Psi`, `z`. Die Belegung ist wie folgt:

```
data.input.x ..................... (Nx+2)x1 discretization of [ax,bx]
data.input.y ..................... (Ny+2)x1 discretization of [ay,by]
data.input.t ..................... Ntx1 discretization of [0,T]
data.input.sigma ................. 1x1 diffusion coefficient
data.input.z0 .................... (NxNy)x1 initial value
data.input.f ..................... (NxNy)xNt inhomogeneity
data.input.method ................ char in {'EE','IE','CN','RS'}
data.input.assem ................. char in {'TRUE','FALSE'}
data.input.solve ................. char in {'TRUE','FALSE'}

data.output.Phi .................. (NxNy)x(NxNy) identity operator
data.output.Psi .................. (NxNy)x(NxNy) Laplace operator
data.output.z .................... (NxNy)xNt solution
```

Die Funktion berechnet im Fall `assem=TRUE` die Matrizen Φ, Ψ aus Aufgabe 7 und löst im Fall `solve=TRUE` die Wärmeleitungsgleichung mit dem Expliziten Euler-Verfahren (`EE`), dem Impliziten Euler-Verfahren (`IE`), dem Crank-Nicolson-Verfahren (`CN`) oder mit Rannacher Smoothing (`RS`), d.h. vier impliziten Euler-Schritten zur halben Schrittweite $\frac{\Delta t}{2}$, gefolgt von regulären Crank-Nicolson-Schritten.

Schreiben Sie eine Datei `program01.m`, in der die Struktur `data` erzeugt wird, die Funktion `SolverPde` aufgerufen wird und eine geeignete Graphik der Lösung erzeugt wird. Vergessen Sie dabei nicht, die Randwerte hinzuzufügen. Testen Sie Ihr Programm mit den Daten $\Omega = [0,1]^2$, $\Theta = (0,1)$, $N_x = N_y = N_t = 100$ sowie

1. $\sigma = 0.01$ und $z_0(x,y) = \sin(2\pi x)\sin(2\pi y)$;
2. $\sigma = 0.05$ und $z_0(x,y) = 1$ auf $[0.25, 0.75]^2$, $z_0(x,y) = 0$ sonst;
3. $\sigma = 0.5$ und $z_0(x_i,y_j) = 1$ für $A_{ij} < 0.001$, $z_0(x_i,y_j) = 0$ sonst, $A = \mathrm{rand}(N_x, N_y)$ eine $N_x \times N_y$-Matrix, bestehend aus Werten einer Standard-Normalverteilung des Intervalls $(0,1)$.

Inwieweit unterscheiden sich die Methoden? Dokumentieren Sie Ihre Beobachtungen.