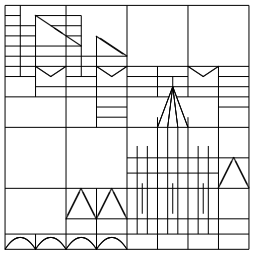
Universität Konstanz
Fachbereich Mathematik und Statistik
Prof. Dr. Stefan Volkwein
Martin Gubisch, Sabrina Rogg
Wintersemester 2015/2016



Ausgabe: Freitag, 29.01.2016
Abgabe: Freitag, 05.02.2016, 10:00 Uhr, Büro G413 bzw. per Email

POD für linear-quadratische Optimalsteuerung
6. Übungsblatt – Programmierteil

Ziel ist es, einen Löser `SolverOpt` für das folgende linear-quadratische Optimierungsproblem mit PDE-Beschränkungen zu implementieren:

$$\min_{(y,u)} J(y,u) = \frac{\sigma_Q}{2}\int_0^T \|y(t)-y_Q(t)\|^2_{\mathcal{L}^2(\Omega)}\,\mathrm{d}t + \frac{\sigma_\Omega}{2}\|y(T)-y_\Omega\|^2_{\mathcal{L}^2(\Omega)} + \frac{\kappa}{2}\int_0^T \|u(t)\|^2_{\mathcal{L}^2(\Omega)}$$

unter der Nebenbedingung

$$\dot{y}(t) - \sigma\Delta y(t) = \beta u(t) \text{ in } \Omega, \qquad y(t) = 0 \text{ auf } \partial\Omega, \qquad y(0) = y_0 \text{ in } \Omega.$$

1. Das Minimum u soll mittels Banachscher Fixpunktiteration aus dem Optimalitätssystem gewonnen werden. Funktionsaufruf: `data = SolverOpt(data)`.

2. Eingabe- und Ausgabeparameter:

```
data.input.u0 ............. NxxNyxNt initial control
data.input.sigmaQ ......... 1x1 Q weight of J
data.input.sigmaOm ........ 1x1 Om weight of J
data.input.zQ ............. NxxNyxNt desired Q state
data.input.zOm ............ NxxNy desired Om state
data.input.kappa .......... 1x1 regularization parameter
data.input.beta ........... 1x1 right-hand side scalar
data.input.sigma .......... 1x1 diffusion coefficient
data.input.z0 ............. NxxNy initial value
data.input.Max ............ 1x1 maximal iteration number
data.input.Tol ............ 1x1 termination tolerance

data.output.U ............. NxxNyxNt optimal control
data.output.Z ............. NxxNyxNt optimal state
data.output.P ............. NxxNyxNt optimal adjoint state
data.output.Count ......... 1x1 number of required iterates
data.output.Res ........... 1xCount residuals max(abs(Fu-u))
data.output.Success ....... char in {TRUE,FALSE}
```

3. Terminierung: Die Iteration soll enden, wenn die maximale Iterationszahl `Max` erreicht ist – in dem Fall soll dem Anwender durch die Flag `Success=FALSE` mitgeteilt werden, dass der erhaltene Punkt `U` noch nicht nahe genug beim gesuchten Minimum liegt – oder wenn die Differenz zwischen dem aktuellen und dem vorherigen Punkt kleiner als `Tol` ist – in dem Fall wird natürlich `Success=TRUE` gesetzt.

4. Zum Lösen der Zustandsgleichung und der adjungierten Gleichung soll die bereits implementierte Funktion `SolverPde` (mit Rannacher-Smoothing) aufgerufen werden.

5. Schreiben Sie ein Script `program03`, um Ihren Löser mit den folgenden Daten zu testen: $(x,y) \in [0,1]\times[0,1]$, $t \in [0,1]$, $N_x = 100$, $N_y = 100$, $N_t = 100$, $\sigma_Q = 1$, $\sigma_\Omega = 1$, $\beta = 1$, $\sigma = 1$, $z_0(t,x,y) = \chi_{[0.25,0.75]^2}(x,y)$, $z_Q(t,x,y) = e^{-t}z_0(x,y)$, $z_\Omega(x,y) = 0$, $u_0(t,x,y) = 1$ und $\kappa \in \{0.1, 0.03, 0.01\}$. Setzen Sie `Max=25` und `Tol=1.0e-05`.

6. Visualisieren Sie zu jedem Wert κ die erhaltenen Residuen. Kommentieren Sie das Ergebnis.

7. Setzen Sie nun POD-Modellreduktion ein. Bestimmen Sie die Residuen für den Fall $\kappa = 0.1$ und dem zur initialen Kontrolle $u_0 \equiv 1$ gerhörigen POD-Modell sowie die Residuen, die sich nach einem Basis-Update ergeben.