Universität Konstanz *WS 11/12*
Fachbereich Mathematik und Statistik
S. Volkwein, O. Lass

Übungen zu **Modellreduktion mit Proper Orthogonal Decomposition**

http://www.math.uni-konstanz.de/numerik/personen/volkwein/teaching/

Submission: 16.12.2011, 10:00 o'clock
Codes by E-Mail and Reports together with the Homework

Program 2 (6 Points)

In this program we focus on the POD basis generation. For this we have to solve the following optimization problem

$$\min_{\psi_1,\ldots,\psi_\ell\in\mathbb{R}^{n^2}} \sum_{j=1}^{\texttt{tsteps}+1} \alpha_j \left\| y_j - \sum_{i=1}^{\ell} \langle y_j,\psi_i\rangle_W \psi_i \right\|_W^2 \quad \text{s.t.} \quad \langle \psi_i,\psi_j\rangle_W = \delta_{ij} \ \text{ for } \ 1\le i,j\le \ell, \ (1)$$

for given snapshots $y_j = y(t_j) \in \mathbb{R}^{n^2}, j = 1,\ldots,$ `tsteps` $+ 1$. The positive weights α_j are set to the trapezoidal weights for the one-dimensional integration over time. For the inner products we use the Euclidean inner product (`'E'`) and the $L^2(\Omega)$ inner product (`'L2'`), see Exercise 9. Structure your code as follows:

`main` ... Main script file, where all parameters are set and desired results are plotted. (Modification of `main` from Program 1.)

`W = weight_matrix(n,wtype)` ... Computes the weight matrix for the inner product, i.e. $\langle u,v\rangle_W = u^\top W v$. The input parameters are the number of inner points `n` and the weight matrix type `wtype` (`'E'`, `'L2'`). The weight matrix `W` for the inner points is the return value.

`[lambda,Psi,traceK] = pod_basis(Y,pod,W,ell)` ... Computes the POD basis by solving (1). The input variables are `Y` the matrix containing the snapshots excluding the boundary points, `pod` the method for computing the POD basis (`'eig'`,`'svd'`), `W` the weight matrix for the inner products and `ell` the number of desired POD basis. The outputs are the eigenvalues `lambda`, the POD basis `Psi` and the trace of the correlation matrix `traceK`. In case the function is called as `pod_basis(Y,pod,W)` all eigenvalues and POD basis should be computed. Use the MATLAB routines `eigs` and `svds` whenever possible.

To compute the snapshots use the code of Program 1 with Rannacher smoothing (`method = 'RS'`). Further we set `n`= 100 and `tsteps`= 100. For the diffusion coefficient c and the initial condition y_0 we choose

- $c = 0.01$ and $y_0(x) = \sin(2\pi x_1)\sin(\pi x_2)$,
- $c = 0.05$ and $y_0(x) = \begin{cases} 1, & \text{on } (0.25,0.75)\times(0.25,0.75), \\ 0, & \text{otherwise,} \end{cases}$

as in Program 1. For the two settings plot the decay of the eigenvalues scaled by `traceK` in semi-log scale. What do you observe? How do the two methods compare in performance? Note that for the case `'eig'` negative and complex eigenvalues can occur due to numerical issues. Hence only plot the absolute value of the real part. Further plot the first four POD basis functions (Do not forget to add the boundary). What do you observe? In the written report give details to your implementation and your observation. Submit your code as a ZIP or TAR/ZIP archive containing one folder named `your_lastname_prog02`.