R. Hiptmair
S. Hosseini
C. Pagliantini

Spring Term 2014

Numerical Analysis I

ETH Zürich
D-MATH

Homework Problem Sheet 8

Introduction. This problem sheet is devoted to spline interpolation and involves a great deal of MATLAB implementation. Thus, it offers essential practice of implementation skills, which are supposed a major learning outcome of the course.

Problem 8.1 Quadratic Splines

Beyond the cubic splines presented in class, splines of general polynomial degree can be defined. One member of the resulting family of piecewise polynomial functions are the *quadratic splines*, which we consider on the interval $[0,1]$ in this problem.

Definition. Given a knot set $\mathcal{N} := \{0 =: x_0 < x_1 < \ldots < x_{n-1} < x_n := 1\}$ we define the space of quadratic splines over $\mathcal{N}$ as

$$\mathcal{S}_{2,\mathcal{N}} := \{s \in C^1([0,1]) : \ s_j = s|_{[x_{j-1},x_j]} \in \mathbb{P}_2 \ \ \forall j = 1,\ldots,n\} \ .$$

In particular, we deal with quadratic spline interpolation of periodic functions: Consider a 1-periodic function $f : \mathbb{R} \to \mathbb{R}$, that is, $f(t+1) = f(t)$ for all $t \in \mathbb{R}$, and a set of nodes

$$\mathcal{N} := \{0 = x_0 < x_1 < x_2 < \cdots < x_{n-1} < x_n = 1\} \subset [0,1] \ .$$

We want to approximate f using a *1-periodic* quadratic spline function $s \in \mathcal{S}_{2,\mathcal{N}}$, which interpolates f in the *midpoints* of the intervals $[x_{j-1}, x_j]$ for $j = 0,\ldots,n$.

Similarly to what was done in Section 3.9 of the lecture for cubic splines, see NMI_3, we locally parameterize a quadratic spline function $s \in \mathcal{S}_{2,\mathcal{N}}$ as follows. For all $j = 1,\ldots,n$,

$$s|_{[x_{j-1},x_j]}(x) = d_j\,\tau^2 + c_j\,4\,\tau(1-\tau) + d_{j-1}\,(1-\tau)^2 \ , \quad \tau := \frac{x - x_{j-1}}{x_j - x_{j-1}} \ , \tag{8.1.1}$$

with $c_j, d_k \in \mathbb{R}$, $j = 1,\ldots,n,\ k = 0,\ldots,n$.

(8.1a) How are the coefficients $\{d_j\}_{j=0}^n$ related to particular point values of s?

(8.1b) What is the dimension of the subspace of *1-periodic* spline functions in $\mathcal{S}_{2,\mathcal{N}}$?

HINT: Rely on a heuristic counting argument based subtracting the number of linear constraints implied by the continuity requirements from the dimension of the space of $\mathcal{N}$-piecewise quadratic polynomials.

(8.1c) What kind of continuity is already guaranteed by the use of the representation (8.1.1)?

(8.1d) Derive a linear system of equations (system matrix and right hand side) whose solution provides the coefficients c_j and d_j in the local representation (8.1.1) of a *1-periodic quadratic spline* s that satisfies the interpolation conditions

$$s(\tfrac{1}{2}(x_{j-1}+x_j)) = y_j \ , j = 1,\ldots,n \ , \tag{8.1.2}$$

where the values y_j, $j = 1,\ldots,n$ are given.

HINT: We know $\mathcal{S}_{2,\mathcal{N}} \subset C^1([0,1])$, which provides linear constraints at the knots x_j, $j = 1,\ldots,n$. The constraints attached to x_n are due to periodicity.

(8.1e) Implement an efficient MATLAB function

```
s = quadspline(x,y,t)
```

which takes as input a (sorted) knot vector `x` (of length $n-1$, because $x_0 = 0$ and $x_n = 1$ will be taken for granted), a n-vector `y` containing the values y_j of the 1-periodic quadratic spline at the midpoints $\frac{1}{2}(x_{j-1}+x_j)$, $j = 1,\ldots,n$, and a *sorted* N-vector `t` of evaluation points in $[0,1]$.

The function returns the values of the interpolating quadratic spline s at the evaluation points `t`.

HINT: You should aim for an efficient implementation making use of the fact that the vectors `x` and `t` are sorted.

(8.1f) Test your code from subproblem (8.1e) by interpolating the function $f(x) := \mathrm{e}^{\sin(2\pi x)}$ in the interpolation nodes $= \left\{\frac{1}{20} + j/10\right\}_{j=0}^{10}$ with a 1-periodic quadratic splines over the knot set $\mathcal{N} = \{\frac{j}{10}\}_{j=0}^{10}$. Plot the function f and the interpolating quadratic spline s on the interval $[0,1]$ based on the evaluation of both functions in the points $t_j := \frac{j}{200}$, $j = 0,\ldots,200$.

(8.1g) As in subproblem (8.1f) we consider the interpolation of the function $f(x) := \mathrm{e}^{\sin(2\pi x)}$ in the interpolation nodes $= \left\{\frac{1}{2n} + j/n\right\}_{j=0}^{n}$, $n \in \mathbb{N}$, with a 1-periodic quadratic splines s_n over the knot set $\mathcal{N} = \{\frac{j}{n}\}_{j=0}^{n}$. This time we are interested in the behavior of maximum norm of the interpolation error

$$\|f - s_n\|_{\infty,[0,1]} = \sup_{x\in[0,1]} |f(x) - s_n(x)| \ ,$$

as a function of n and its asymptotic trend for $n \to \infty$.

Implement a MATLAB script that computes the interpolation error in the maximum norm for $n = 2^2,\ldots,2^{11}$. Create a loglog plot of the error versus the number n of knot intervals. Describe qualitatively and quantitatively the convergence $s_n \to f$ in the maximum norm for $n \to \infty$.

HINT: As explained in class, the maximum norm of the interpolation error can only be computed approximately. To obtain an approximation evaluate the difference $|f(t) - s_n(t)|$ at $N \gg n$ equidistant evaluation points in $[0,1]$ and then take the maximum. You may choose $N = 10.000$.

Problem 8.2 Curve Interpolation

The focus of [NMI, Ch. 3] was on the interpolation of data points by means of functions belonging to a certain linear space. A different interpolation task is to find a curve containing each point of a set $\{\mathbf{p}_0, ..., \mathbf{p}_n\} \subset \mathbb{R}^2$.

This task can be translated into a standard interpolation problem after recalling that a curve in $\mathbb{R}^2$ can be described by a mapping (*parameterization*) $\gamma : [\mathbf{0}, \mathbf{1}] \mapsto \mathbb{R}^{\mathbf{2}}$, $\gamma(\mathbf{t}) = \binom{\mathbf{s_1(t)}}{\mathbf{s_2(t)}}$. Hence, given the nodes $0 = x_0 < x_1 < ... < x_{n-1} < x_n = 1$, for $n > 1$, we aim at finding interpolating functions $s_1, s_2 : [0, 1] \to \mathbb{R}$, such that $s_i(x_j) = (\mathbf{p}_j)_i$, $i = 1, 2$ and $j = 0, ..., n$. This means that we separately interpolate the coordinates $(\mathbf{p}_j)_1$ and $(\mathbf{p}_j)_2$ of $\mathbf{p}_j$ for $j = 0, ..., n$.

A crucial new aspect is that there are infinitely many parameterizations of a given curve: for any strictly monotonous and surjective $h : [0, 1] \to [0, 1]$ the mappings γ and $\widetilde{\gamma} := \gamma \circ \mathbf{h}$ describe exactly the same curve. On the other hand, the selection of nodes affects the interpolants s_1 and s_2 and leads to different interpolating curves.

Concerning the choice of the nodes $\{x_j\}_{j=0}^n$, we consider two options:

(i) equidistant parameterization: $x_j = \frac{j}{n}$;

(ii) segment length parameterization: $x_j = \dfrac{\sum_{l=1}^{j} |\mathbf{p}_l - \mathbf{p}_{l-1}|}{\sum_{l=1}^{n} |\mathbf{p}_l - \mathbf{p}_{l-1}|}$.

Point data $\{\mathbf{p}_j\}_{j=0}^n$ will be generated by the MATLAB function `heart` available on the course webpage as `heart.m`. This script stores them in the output $2 \times (n + 1)$-matrix `xy_heart`, whose columns represent point coordinates.

(8.2a) Write a MATLAB function

```
[pol, spl] = curveintp(x,yy,t)
```

which takes as input the $n + 1$ points $\mathbf{p}_j \in \mathbb{R}^2$, $j = 0, \ldots, n$, whose coordinates are stored in the $2 \times (n + 1)$ matrix `yy`, the row vector `x` of the interpolation nodes $(x_0, x_1, \ldots, x_n) \in \mathbb{R}^{n+1}$ and a row vector `t` of N evaluation points. The function returns in a $2 \times N$ matrix `pol` the sampled values of the curve interpolated through global polynomial interpolation (use the barycentric formula as in NMI_3 which is available as `ipolevalbarycentric`) and the $2 \times N$ matrix `spl` containing the interpolated values obtained using *complete* cubic spline interpolation.

For the latter, the additional constraints are given by the slopes at the first and last nodes. The required derivatives $s_1'(0)$, $s_2'(0)$, $s_1'(1)$, and $s_2'(1)$ should be computed from the directions of the line segments connecting $\mathbf{p}_0$ and $\mathbf{p}_1$, and $\mathbf{p}_{n-1}$ and $\mathbf{p}_n$, respectively.

To compute the complete interpolating spline you may use the MATLAB *built-in function* `spline`.

HINT: Code for `ipolevalbarycentric` is available as `ipolevalbarycentric.m` on the course webpage.

Read the MATLAB help page about the `spline` command and learn how to impose the derivatives at the endpoints.

(8.2b) Plot the curves obtained by global polynomial interpolation and by spline interpolation of the `heart` data set. The interpolation nodes should be generated according to the two options (i) and (ii). Use as evaluation points `t=0:0.005:1`.

(8.2c) The two interpolation schemes introduced in subproblem (8.2b) combined with either of the choices (i) and (ii) of interpolation nodes in the parameter domain define a *curve interpolation*

operator

$$I_{\text{curve}} : (\mathbb{R}^2)^{n+1} \to \{\gamma : [0,1] \to \mathbb{R}^2\} \, .$$

Which of the four curve interpolation operators (two different interpolation methods $\times$ two options for choosing the nodes) is a linear mapping?

(8.2d) Another interesting aspect is how a transformation of the given points affects the interpolating curve. Let $T_A : \mathbb{R}^2 \to \mathbb{R}^2$ be the linear mapping $T_A(\mathbf{x}) := \mathbf{A}\mathbf{x}$ with regular matrix $\mathbf{A} \in \mathbb{R}^{2,2}$. Consider the two interpolation techniques given by global polynomial interpolation and cubic spline interpolation.

For a fixed $t \in [0,1]$, check for which choice of the interpolation nodes the following diagram commutes

$$\begin{array}{ccc} (\mathbb{R}^2)^{n+1} & \xrightarrow{T_A^{n+1}} & (\mathbb{R}^2)^{n+1} \\ \Big\downarrow{\scriptstyle I^t_{\text{curve}}} & & \Big\downarrow{\scriptstyle I^t_{\text{curve}}} \\ \{[0,1] \mapsto \mathbb{R}^2\} & \xrightarrow[T_A]{} & \{[0,1] \mapsto \mathbb{R}^2\} \end{array}$$

namely

$$I_{\text{curve}}(\{T_A \mathbf{p}_j\}_j)(t) = T_A(I_{\text{curve}}(\{\mathbf{p}_j\}_j)(t)),$$

where $T_A^{n+1}(\{\mathbf{p}_j\}_j) := \{T_A \mathbf{p}_j\}_j$, and $I^t_{\text{curve}}(\{\mathbf{p}_j\}_j) := I_{\text{curve}}(\{\mathbf{p}_j\}_j)(t)$.

Consider the two different options $\mathbf{A} \in \{\mathbf{A}_1, \mathbf{A}_2\}$, given by

$$\mathbf{A}_1 = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} \quad \text{with} \quad \theta = \frac{\pi}{3}, \quad \text{and} \quad \mathbf{A}_2 = \begin{pmatrix} 1 & \lambda \\ 0 & 1 \end{pmatrix} \quad \text{with} \quad \lambda = 3.$$

HINT: Distinguish two cases: equidistant interpolation nodes (i) and nodes given by segment length parameterization (ii). Note that $\mathbf{A}_1$ is a rotation matrix and $\mathbf{A}_2$ is a shear deformation matrix. Which matrix is orthogonal?

Published on April 14, 2014.

To be submitted on April 29/30, 2014.

MATLAB: Submit all files in the online system. Include the files that generate the plots. Label all your plots. Include commands to run your functions. Comment on your results.

Written exercises: Hand-in the solutions during the exercise class or in the labeled boxes in HG G 53.x.

References

[NMI] Lecture Slides for the course "Numerical Analysis I".

Last modified on April 14, 2014