R. Hiptmair
S. Hosseini
C. Pagliantini

Spring Term 2014

Numerical Analysis I

ETH Zürich
D-MATH

Homework Problem Sheet 5

Introduction. This problem sheet is devoted to polynomial interpolation.

Problem 5.1 Evaluating the Derivatives of Interpolating Polynomials

This problem is dedicated to fundamental algorithms for polynomial interpolation as discussed in Section 3.1 of the lecture. We will also learn about some of their extensions and applications. This problem involves substantial implementation in MATLAB.

(5.1a) Write a MATLAB function

```
p = AitNevpoleval(x,y,t)
```

that, using the Aitken-Neville scheme from [NMI, Thm. 3.7], evaluates the polynomial $p \in \mathbb{P}_n$ interpolating the data points (x_i, y_i), $i = 0, \ldots, n$, for pairwise different $x_i \in \mathbb{R}$ and data values $y_i \in \mathbb{R}$, in $t \in \mathbb{R}$. The data points are passed through the vectors `x` and `y`.

(5.1b) Write an efficient MATLAB function

```
dp = dipoleval(x,y,t)
```

that returns the value $p'(t)$ of the *derivative* of the polynomial $p \in \mathbb{P}_n$ interpolating the data points (x_i, y_i), $i = 0, \ldots, n$, for pairwise different $x_i \in \mathbb{R}$ and data values $y_i \in \mathbb{R}$, evaluated at $t \in \mathbb{R}$.

HINT: Use the recursion formula for the Aitken-Neville scheme from [NMI, Thm. 3.7] and differentiate it.

(5.1c) Test the implementation of `dipoleval` in subproblem (5.1b) by comparing with the result obtained using simple *difference quotients* applied to the point values computed in subproblem (5.1a) through the routine `AitNevpoleval`. That is, we approximate

$$p'\left(\tfrac{1}{2}(t_i + t_{i+1})\right) \approx \frac{p(t_{i+1}) - p(t_i)}{t_{i+1} - t_i} \,.$$

Use $m = n + 1 = 10$ interpolation points `x = linspace(0,1,m)`, `y = rand(1,m)` and evaluation points `t = linspace(0,1,N)` for $N = 100$. Plot the results for the two implementations.

HINT: You may use the MATLAB command `diff`.

Problem 5.2 Interpolation with Exponentials

This problem demonstrates a "nostandard" use of polynomial interpolation in the context of the so-called *exponential interpolation*. The task of exponential interpolation is the following:

> Given a set of data points $\{(x_i, y_i)\}_{i=1}^{2n+1} \in \mathbb{R}^2$ and a parameter $\lambda > 0$, find coefficients $c_j \in \mathbb{R}$ such
>
> $$\sum_{j=-n}^{n} c_j \mathrm{e}^{\lambda x_i j} = y_i, \qquad \forall i = 1, \ldots, 2n+1. \tag{5.2.1}$$

(5.2a) Show that the coefficients c_j, $j = -n, \ldots, n$, are uniquely determined by the interpolation conditions (5.2.1), provided that no two nodes x_i coincide.

HINT: Transform the sum in (5.2.1) into a sum over $j = 0, \ldots, 2n$ by extracting a suitable factor. Relate the reformulation of (5.2.1) and standard polynomial interpolation.

(5.2b) Write a MATLAB function

```
function A = ExpInterpMatrix(x,lambda)
```

that returns the $(2n+1) \times (2n+1)$ matrix of the linear system of equations $\mathbf{A}\mathbf{c} = \mathbf{b}$ for the unknown coefficients c_j equivalent to (5.2.1). The vector `x` passes the interpolation nodes x_i, $i = 1, \ldots, 2n+1$.

HINT: Use the reformulation derived in subproblem (5.2a).

(5.2c) The MATLAB command `polyfit` can be used to compute the coefficients of an interpolating polynomial with respect to the monomial basis. Study the documentation of `polyfit` to find out, how this can be done in detail.

(5.2d) Write a MATLAB function

```
c = expip(x,y,lambda)
```

that uses MATLAB's built-in function `polyfit` to compute the coefficients c_j as defined by (5.2.1). MATLAB's "backslash" operator ("\") must not be used.

(5.2e) Write an *efficient* MATLAB function

```
q = expipeval(x,y,lambda,t)
```

that evaluates

$$q(t) = \sum_{j=-n}^{n} c_j \mathrm{e}^{\lambda t j}$$

for a single point $t \in \mathbb{R}$. Here the arguments `x` and `y` pass the data points, and q is the interpolant from subproblem (5.2d).

HINT: You may use the Aitken-Neville algorithm from theorem [NMI, Thm. 3.7].

(5.2f) Write a MATLAB script

```
expipplot(x,lambda)
```

that plots the *functions*

$$\mathcal{I}_{\exp}(\mathbf{e}_i)\ , \quad (\mathbf{e}_i \mathrel{\hat{=}} i\text{-th unit vector})\ , \quad i = 1, \ldots, 2n+1$$

where $\mathcal{I}_{\exp} : \mathbf{y} \to \{x \mapsto \sum c_j \mathrm{e}^{\lambda x j}\}$ is the interpolation operator specified in subproblem (5.2e) for given $(x_i)_{i=1}^{2n+1}$ and $\lambda > 0$.

Plot these functions for $n = 3$, $x_i = \frac{i-1}{2n}$, $i = 1, \ldots, 2n+1$, $\lambda = 1$ and for $100(2n+1)$ equidistant evaluation points $t \in [0, 1]$.

Problem 5.3 Chebychev Polynomials

In this problem you will meet a special set of polynomials that form a basis for the spaces $\mathbb{P}_{n-1}$ of polynomials of degree $< n$, $n \in \mathbb{N}$. With respect to an also special set of interpolation nodes this basis has rather desirable properties.

Throughout this problem, for $n \in \mathbb{N}_0$ we set

$$T_n(t) := \cos(n \arccos(t))\ , \quad -1 \le t \le 1\ ,$$

and for $n \in \mathbb{N}$

$$\mathcal{Z}_n := \left\{ x_k := \cos\big((k + \tfrac{1}{2})\tfrac{\pi}{n}\big),\ k = 0, \ldots, n-1 \right\}\ . \tag{5.3.1}$$

(5.3a) Show that $T_0(t) = 1$, $T_1(t) = t$, and

$$T_{n+1}(t) + T_{n-1}(t) = 2tT_n(t)\ , \quad n \ge 1\ , \quad -1 \le t \le 1\ . \tag{5.3.2}$$

(5.3b) Show that for $n \ge 1$ the derivatives of the functions T_n satisfy the recursion

$$2T_n(t) = \frac{1}{n+1}\,\frac{d}{dt}T_{n+1}(t) - \frac{1}{n-1}\,\frac{d}{dt}T_{n-1}(t)\ .$$

(5.3c) Show that T_n for $n \ge 1$, is a polynomial of degree n with leading coefficient 2^{n-1}.

(5.3d) Show that

$$\mathcal{Z}_n = \{t \in \mathbb{R} :\ T_n(t) = 0\}\ .$$

(5.3e) Write a MATLAB function

```
function y = evalT(n,t)
```

that computes $y_i := T_n(t_i)$, $n \in \mathbb{N}_0$, for arguments t_i, $i = 1, \ldots, m$, passed in the row vector `t`. No special functions like cos and its inverse must be used.

The results are to be returned in the row vector `y`. What is the asymptotic computational effort of `evalT` for $n \to \infty$ and $m \to \infty$?

HINT: Use the recursion formula (5.3.2) for T_n.

(5.3f) Following the notations of Section 3.A of the lecture, for $n \in \mathbb{N}$ we consider the interpolation problem with $V = \mathbb{P}_{n-1}$ and node set $\mathcal{N} = \mathcal{Z}_n$.

Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be the interpolation matrix for the basis $\mathcal{B} = \{T_0, T_1, \ldots, T_{n-1}\}$ of V. Show that there is a regular diagonal matrix $\mathbf{D}$ such that $\mathbf{AD}$ is an orthogonal matrix.

(5.3g) Using the result of subproblem (5.3f), write an efficient (in terms of computational effort and memory!) MATLAB function

```
function c = chebcoeff(y)
```

that computes the coefficients c_j, $j = 0, \ldots, n-1$, in the representation

$$p(t) = \sum_{j=0}^{n-1} c_j T_j(t)$$

of the polynomial interpolant $p \in \mathbb{P}_{n-1}$ through the points (x_i, y_i), $i = 0, \ldots, n-1$, where the nodes x_i are defined in (5.3.1). The data y_i are passed in the row vector `y`, and the coefficients are returned in the row vector `c`.

(5.3h) Based on (5.3.2) write a MATLAB function

```
function y = chebsum(c,t)
```

that computes

$$y_i = \sum_{j=0}^{n-1} c_j T_j(t_i)$$

for $i = 1, \ldots, m$. The values t_i are the components of the row vector `t`, and the values y_i are made available in the row vector `y`.

What is the asymptotic computational effort of `chebsum` for $n \to \infty$ and $m \to \infty$?

Published on March 24, 2014.

To be submitted on April 1/2, 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 March 25, 2014