Questions

Numerical Methods for CSE
Repeat Exam
FS 2018

Prof. Rima Alaifari

ETH Zürich, D-MATH

27 August 2018

1. *Periodic quadratic splines* (32pts)

   Template: `1.cpp`.

   Given knots $x_i \in \mathbb{R}$ and data points $y_i \in \mathbb{R}$ for $i = 0, \dots, N$ we want to find a periodic quadratic spline f which interpolates them. Assume that $0 < x_0 < x_1 < \cdots < x_N$, N is odd and $N \geq 3$.

   (a) (6pts) Let
   $$f(x) = a_i(x - x_i)^2 + b_i(x - x_i) + c_i \text{ for } x \in [x_i, x_{i+1}]. \tag{1}$$
   Derive on paper the coefficients a_i, b_i, c_i, either explicitly, recursively or as solutions to a linear system of equations which depends only on $\{(x_j, y_j), j = 0, \dots, N\}$, so that

   (α) $f(x_i) = y_i$ for every i,

   (β) f has continuous first derivative on the interval (x_0, x_N),

   (γ) $f'(x_0) = f'(x_N)$.

   Hint: simplifying as much as possible the linear system derived in this subproblem could be very useful in the following subproblems.

   (b) (2pts) Prove/explain concisely why an f satisfying the requirements of Point (a) always exists and is unique.

   (c) (2pts) What part of the statement in Point (b) fails when N is even? Why?

   (d) (10pts) Implement a `C++/Eigen` function `quadraticSpline` which given the datapoints `VectorXd x`, `VectorXd y` returns a matrix with N rows and 3 columns, which has in position $(i, 0), (i, 1), (i, 2)$ respectively the coefficients a_i, b_i, c_i of (1).

   (e) (12pts) Implement a `C++/Eigen` function `quadraticSplineFast` which takes the same input and returns the same output of `quadraticSpline`, but runs in only $O(N)$ time.

   Hint: if you are unable to derive an explicit $O(N)$ procedure, in this exercise you can assume that the `SparseLU` solver from `Eigen` runs in $O(n)$ time when implemented correctly for an $n \times n$ matrix which has $O(1)$ non-empty diagonals.

2. *Data fitting* (22pts)

Template: `2.cpp`.

Let

$$f(k) := \alpha + \beta \sin\left(\frac{2\pi}{366}k\right) + \gamma \cos\left(\frac{2\pi}{366}k\right) + \delta \sin\left(\frac{\pi}{366}k\right)\cos\left(\frac{\pi}{366}k\right), \tag{2}$$

where $k \in \mathbb{N}$ and $\alpha,\ \beta,\ \gamma,\ \delta$ are some parameters.

Consider the following data on the number $H(k)$ of daylight hours in the k-th day of the year:

k	11	32	77	121	152
$H(k)$	9	10	12	14	15

We want to find the best approximation of this data in the least squares sense, using the function $f(k)$.

(a) (4pts) Derive on paper the linear least squares problem (LLSQ)

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2\,, \tag{3}$$

by specifying $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{b} \in \mathbb{R}^m$, for $m, n \in \mathbb{N}$.

(b) (3pts) Is the solution to the above LLSQ unique? Explain.

(c) (10pts) Implement a `C++/Eigen` function `llsq_gsol` which given `MatrixXd A` and `VectorXd b` returns the *generalized* solution `VectorXd x` and the least squares error `llsq_err`.
Hint: Use SVD.

(d) (5pts) Run your implementation of `llsq_solve` for the LLSQ in (a) and print the solution and the least squares error.

3. *Unconstrained optimization* (32pts)

   Template: `3.cpp`.

   According to the WHO growth reference data from the year 2007, the weight of 5-year-old girls in the world can be modelled by the Gaussian distribution function

$$f(w; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(w-\mu)^2}{2\sigma^2}\right). \tag{4}$$

   Given M randomized samples of weights

$$S = \{w_1, w_2, \ldots, w_M\}, \text{ for } M \in \mathbb{N}, \tag{5}$$

   the objective is to estimate the mean μ and variance σ.

   (a) (3pt) Formulate on paper the setting described above as an unconstrained minimization problem,

$$\mathbf{x}^* = \underset{\mathbf{x} \in \mathbb{R}^n}{\arg\min}\, \mathbf{F}_S(\mathbf{x}), \tag{6}$$

   by specifying $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{F}_S : \mathbb{R}^n \to \mathbb{R}^m$, for $m, n \in \mathbb{N}$.

   (b) (4pts) Derive on paper the gradient of $\mathbf{F}_S(\mathbf{x})$ in (a).

   (c) (5pts) Implement a `C++/Eigen` function:

```
VectorXd evalGradF(const VectorXd& S, const VectorXd& x);
```

   which computes the gradient of $\mathbf{F}_S(\mathbf{x})$ devised in (b).

   (d) (4pts) Derive on paper the Hessian of $\mathbf{F}_S(\mathbf{x})$ in (a).

   (e) (6pts) Implement a `C++/Eigen` function:

```
MatrixXd evalHessF(const VectorXd& S, const VectorXd& x);
```

   which computes the Hessian of $\mathbf{F}_S(\mathbf{x})$ devised in (d).

   (f) (2pts) Write explicitly on paper an iteration of the Newton method to solve subproblem (a).

   (g) (5pts) Implement a `C++/Eigen` function for the Newton optimization devised in (f):
   `void newtonOpt(const VectorXd& S, const double tol, const int maxItr, VectorXd& x);`
   Use appropriate termination criteria.

   (h) (3pt) Print the estimates for mean and variance obtained from your implementation, for the data set given in the template.