Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Prof. Petros Koumoutsakos
CSE Lab, ETH Zürich
www.cse-lab.ethz.ch

Advanced Computational Science
SS 2009

Serie 1 Ausgabe: Februrary 24, 2009 Abgabe: March 3, 2009

Guidelines

▷ Submission: Submit a written or typed report with your answers and graphics.

▷ Code: Send your source code (and makefile) to the TA of your session. Use "[ACS] Homework1" in the subject of your email.

▷ Credit: Your report should list all the contributors.

▷ Bugs: Print-outs of your source code are *not* required in your report, unless you have bugs. Help us give you partial marks.

▷ Deadline: Assignments have to be handed in at the beginning of the next exercise session.

Exercise 1 (20 points) Lagrange interpolation

The polynomial interpolant through N points $(x_k, y_k = f(x_k))$ can be written in terms of Lagrange polynomials $L_k(x)$

$$P(x) = \sum_{k=1}^{N} y_k L_k(x)$$

where

$$L_j(x) = \frac{(x-x_1)\cdots(x-x_{j-1})(x-x_{j+1})(x-x_N)}{(x_j-x_1)\cdots(x_j-x_{j-1})(x_j-x_{j+1})(x_j-x_N)} \ .$$

In practice, however, the construction and evaluation of this interpolant can be carried out far more efficiently through the use of divided differences. Divided differences $y[i-m, i-m+1, \ldots, i-1, i]$ are defined (and can be computed...) through the recursive relations:

$$y[i] = y_i \ ,$$

$$y[i-m, i-m+1, \ldots, i-1, i] = \frac{y[i-m+1, \ldots, i-1, i] - y[i-m, i-m+1, \ldots, i-1]}{x_i - x_{i-m}}$$

The polynomial is then given by

$$P(x) = y[1] + y[1,2](x-x_1) + y[1,2,3](x-x_1)(x-x_2) + \ldots + y[1,2,\ldots,N](x-x_1)\ldots(x-x_{N-1}) \ ,$$

an expression which involves *only* $\mathcal{O}(N)$ floating point multiplications.

▷ Part 1 (10 points) Write your own polynomial interpolation code where the polynomial construction and representation uses the divided difference approach. The code should (1) build the polynomial from a series of x_k, y_k (=input), (2) print the divided differences involved in the polynomial and (3) evaluate the polynomial at any x or series x_l.

▷ Part 2 (5 points) Apply your code to the N points $(x_k, y_k = e^{x_k})$ for $x_k = -1+2(k-1)/(N-1)$ for $N = 5$ and 7. For each case, print the relevant divided differences. Produce two plots: (1) the exact function and the two interpolants; (2) the error of the interpolants. Comment your results.

▷ Part 3 (5 points) Apply your code to the N points $(x_k, y_k = (1+25x_k^2)^{-1})$ for $x_k = -1+2(k-1)/(N-1)$ for $N = 7$ and 11. For each case, print the relevant divided differences. Produce two plots: (1) the exact function and the two interpolants; (2) the error of the interpolants. Comment your results.

Exercise 2 (20 points) Spline interpolation

Polynomials can be problematic as global interpolants (cf Exercise 1). Thanks to their piecewise definition, cubic splines fix this. The spline P_i defined over the interval $[x_i, x_{i+1}]$ reads

$$\begin{aligned} P_i(x) &= \frac{f_i''}{6}\left(\frac{(x_{i+1}-x)^3}{\Delta_i} - \Delta_i(x_{i+1}-x)\right) + \frac{f_{i+1}''}{6}\left(\frac{(x-x_i)^3}{\Delta_i} - \Delta_i(x-x_i)\right) \\ &\quad + \frac{y_i}{\Delta_i}(x_{i+1}-x) + \frac{y_{i+1}}{\Delta_i}(x-x_i) \end{aligned}$$

with $\Delta_i = x_{i+1} - x_i$. The curvatures f_i'' are determined through the enforcement of the continuity of the derivative at x_i, $P_i'(x_i) = P_{i+1}'(x_i)$,

$$\frac{\Delta_{i-1}}{6} f_{i-1}'' + \frac{\Delta_{i-1}+\Delta_i}{3} f_i'' + \frac{\Delta_i}{6} f_{i+1}'' = \frac{y_{i+1}-y_i}{\Delta_i} - \frac{y_i - y_{i-1}}{\Delta_{i-1}} \,. \tag{1}$$

These $N-2$ equations are completed with 2 end conditions, e.g. *natural* splines have $f_1'' = f_N'' = 0$. The resulting system is tridiagonal

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ \frac{\Delta_1}{6} & \frac{\Delta_1+\Delta_2}{3} & \frac{\Delta_2}{6} & 0 & \cdots & 0 \\ 0 & \frac{\Delta_2}{6} & \frac{\Delta_2+\Delta_3}{3} & \frac{\Delta_3}{6} & 0 & \vdots \\ \vdots & & & \frac{\Delta_{N-2}}{6} & \frac{\Delta_{N-2}+\Delta_{N-1}}{3} & \frac{\Delta_{N-1}}{6} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} f_1'' \\ f_2'' \\ f_3'' \\ \vdots \\ f_N'' \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{y_3-y_2}{\Delta_2} - \frac{y_2-y_1}{\Delta_1} \\ \frac{y_4-y_3}{\Delta_3} - \frac{y_3-y_2}{\Delta_2} \\ \vdots \\ 0 \end{pmatrix}$$

and can be easily solved in two passes of elimination/substitution ($\mathcal{O}(N)$).

▷ Part 1 (15 points) Write your own spline interpolation code. Use natural splines. The code should (1) build the splines from a series of x_k, y_k (=input), (2) print the f_i'', and (3) evaluate the polynomial at any x or series x_l.

▷ Part 2 (5 points) Apply your code to the N points $(x_k, y_k = (1+25x_k^2)^{-1})$ for $x_k = -1+2(k-1)/(N-1)$ for $N = 7$ and 11. For each case, print the f_i''. Produce two plots: (1) the exact function and the two interpolants; (2) the error of the interpolants. Comment your results.