MATH 2130 Midterm Test 3 — Solutions

Prof. A. Willms, Dept. of Mathematics and Statistics, University of Guelph

Mar. 27, 2019

SECTION A.
ANSWER QUESTIONS 1–9 IN THE SPACE PROVIDED.

12 pts. 1. Consider the points $(-3,3)$, $(-1,2)$, and $(2,-3)$.

(a) Find the Lagrange form for the lowest order polynomial that interpolates these three points.

(b) Find the Newton form for the same polynomial.

Solution:

(a) Define

$$L_1(x) = \frac{(x-(-1))(x-2)}{(-3-(-1))(-3-2)} = \frac{(x+1)(x-2)}{10}, \quad L_2(x) = \frac{(x-(-3))(x-2)}{(-1-(-3))(-1-2)} = \frac{(x+3)(x-2)}{-6},$$
$$L_3(x) = \frac{(x-(-3))(x-(-1))}{(2-(-3))(2-(-1))} = \frac{(x+3)(x+1)}{15}.$$

Then the Lagrange form of the polynomial is

$$L(x) = 3L_1(x) + 2L_2(x) - 3L_3(x).$$

(b) The Newton divided difference table is

$$\begin{array}{r|ll} -3 & 3 & \frac{2-3}{-1-(-3)} = -\frac{1}{2} \qquad \frac{-5/3-(-1/2)}{2-(-3)} = \frac{-10/6+3/6}{5} = -\frac{7}{30} \\ -1 & 2 & \frac{-3-2}{2-(-1)} = -\frac{5}{3} \\ 2 & -3 & \end{array}$$

Therefore the Newton form for the polynomial is

$$y(x) = 3 - \frac{1}{2}(x+3) - \frac{7}{30}(x+3)(x+1).$$

7 pts. 2. Use a finite difference table to determine the Hermite polynomial that interpolates the points $(0,2)$, $(2,6)$, and $(3,9)$, and has derivative equal to 3 at $x=0$.

Solution:
The finite difference table is

$$\begin{array}{r|llll} 0 & 2 & 3 & -\frac{1}{2} & \frac{5}{18} \\ 0 & 2 & 2 & \frac{1}{3} & \\ 2 & 6 & 3 & & \\ 3 & 9 & & & \end{array}$$

Therefore the polynomial is

$$p(x) = 2 + 3(x-0) - \frac{1}{2}(x-0)^2 + \frac{5}{18}(x-0)^2(x-2).$$

12 pts. 3. Below is MATLAB code intended to implement the steepest descent algorithm to find a minimum point for $f : \mathbb{R}^n \to \mathbb{R}$. The (non-comment) lines are numbered on the left. There are six errors on six different lines. Circle the line numbers of the lines with errors and, in the space below or to the right, write corrections to these lines. No points are lost for circling line numbers for lines that in fact contain no errors; however, you must not circle more than six line numbers.

```
1       function steepestdescent(fun,x0,TOLX,TOLG,maxiter)
        %  steepestdescent -  find minimum of multivariable f by steepest descent
        %  INPUT:
        %    fun : a handle to a function of n variables, x, and a flag, called as
        %          fun(x,flag). If flag=1 then fun should return f(x) otherwise
        %          it should return g(x), the gradient of f.
        %    x0 : initial guess for x
        %    TOLX : absolute convergence criterion for x, iteration will cease if
        %           two successive guesses are closer than this
        %    TOLG : convergence criterion for the gradient of f, iteration will cease
        %           if norm of gradient is below this
        %    maxiter : maximum number of iterations, iteration will cease if number of
        %              iterations exceeds this
        %  OUTPUT:
        %    x : estimate of solution to f(x)=0
        %    K : number of iterations performed
2       x = x0;
3       s = 1;   % initial step will equal gradient
4       f = fun(x,2);
5       g = fun(x,1);
6       normg = norm(g);
7       K = 0;
8       while K < maxiter && normg > TOLG && s*normg > TOLX
9           K = K + 1;
            % Compute trial point, xtry, and f at xtry.
10          xtry = x - s*g;
11          ftry = fun(xtry,1);
12          while ftry <= f
                % reduce step size by a half
13              s = 0.5*s;
14              if s*normg < TOLX   % no further improvement likely, return to caller
15                  break
16              end
                % update xtry and ftry
17              xtry = x + s*g;
18              ftry = fun(xtry,1);
19          end
20          x = xtry;
21          f = ftry;
            % Compute the gradient at the new point, and the norm of the gradient.
22          g = fun(x,2);
23          normg = norm(g);
24      end
25      if K >= maxiter
26          warning('maximum number of iterations reached')
27      end
28      end  % end of function
```

Solution:
The corrected lines are given below.

```
1       function [x,K] = steepestdescent(fun,x0,TOLX,TOLG,maxiter)
4       f = fun(x,1);
5       g = fun(x,2);
12          while ftry >= f
15                  return
17              xtry = x - s*g;
```

12 pts. 4. Write a MATLAB function that will generate the coefficients of the Newton form of the interpolating polynomial through the points (x_i, y_i), $1 \le i \le n$. The input to the function will be two column vectors of equal length, `x` and `y`. The output should be a vector `coef` that gives the coefficients of the polynomial (that is, the top row of the Newton finite difference table). You do not need to evaluate the interpolating polynomial at any points. The first part of the function is written for you below; it should not be modified. You must complete the remainder of the function.

```
function coef = newton_interp(x,y)
% newton_interp - generate coefficients for the Newton interpolation polynomial
%
%  coef = newton_interp(x,y) will determine the coefficients of the
%     Newton form of the interpolating polynomial through the points
%     (x_i, y_i)
%
%  INPUT:
%   x : a column vector of x values, length n
%   y : a column vector of y values, length n
%
%  OUTPUT:
%   coef : coefficients (top row of divided difference table)
%
[n,k] = size(x);
if any(size(y) ~= [n,k]) || k>1
    error('x and y must be column vectors of the same length');
end
% Compute the divided difference table
dt = zeros(n,n);
%%% COMPLETE THE REMAINDER
```

Solution:
The remainder of the code is below.

```
dt(:,1) = y;
for j = 2:n
    for i=1:n-j+1
        dt(i,j) = (dt(i+1,j-1) - dt(i,j-1))/(x(i+j-1) - x(i));
    end
end
% extract top row of table
coef = dt(1,:);
```

7 pts. 5. If one were to approximate the derivative of $f(x_i)$ using the forward difference formula: $A_1(h) = \dfrac{f(x_i + h) - f(x_i)}{h}$, then the error associated with this approximation is of the form

$$\frac{df}{dx}(x_i) - A_1(h) = K_1 h + K_2 h^2 + \cdots = O(h),$$

where the K_i are constants. For a given h, $A_1(h/2)$ is a backward difference approximation of the derivative using a stepsize half as long.

(a) Show how you should combine $A_1(h)$ and $A_1(h/2)$ so that the combination gives an approximation with error that is $O(h^2)$?

(b) Write out and simplify the formula for the derivative (in terms of f evaluated at various points) corresponding to the combination you obtained in (a).

Solution:

(a) We have

$$\frac{df}{dx}(x_i) = A_1(h) + K_1 h + K_2 h^2 + \cdots$$
$$\frac{df}{dx}(x_i) = A_1(h/2) + K_1 \frac{h}{2} + K_2 \frac{h^2}{4} + \cdots$$

Taking two times the second equation minus the first equation yields

$$\frac{df}{dx}(x_i) = 2A_1(h/2) - A_1(h) - \frac{K_2}{2} h^2 + \cdots$$

Thus $A_2(h) \equiv 2A_1(h/2) - A_1(h)$ is an approximation of $\frac{df}{dx}(x_i)$ with error of order $O(h^2)$.

(b) We have

$$A_2(h) = 2A_1(h/2) - A_1(h) = 2\frac{f(x_i + h/2) - f(x_i)}{h/2} - \frac{f(x_i + h) - f(x_i)}{h}$$
$$= \frac{-f(x_i + h) + 4f(x_i + h/2) - 3f(x_i)}{h}.$$

6 pts. 6. Briefly compare conditions imposed at the *interior* nodes for cubic spline interpolation and piecewise cubic Hermite interpolation of the data $\{(x_i, y_i)\}_{i=1}^n$.

Solution:
Both methods require the interpolating function to pass through the points (x_i, y_i). For cubic splines, the first and second derivatives at all interior nodes, x_i, $1 < i < n$, are forced to be continuous. In contrast, for piecewise cubic Hermite interpolation, the first derivatives at the interior nodes are forced to be equal to the slope of lines joining the two neighbours, that is, the slope at x_i is forced to be $\frac{y_{i+1} - y_{i-1}}{x_{i+1} - x_{i-1}}$, $1 < i < n$.

4 pts. 7. The Newton-Cotes formulas for numerical integration are all based on the same idea to estimate the integral $I = \int_a^b f(x)\,dx$. The only difference between the various formulas is the number of points used (e.g. the trapezoid method uses 2 points, Simpson's rule uses 3 points, etc.). What is the basic idea for how an n-point Newton-Cotes formula estimates I?

Solution:
An n-point Newton-Cotes formula is determined by finding the interpolating polynomial (of degree $n-1$ or less) through the n evenly-spaced points on the interval $[a, b]$ and then integrating this polynomial exactly to get an estimate for I.

4 pts. 8. If f is a function from $\mathbb{R}^n$ to $\mathbb{R}$, what is the direction of greatest increase of f? And what is the direction chosen by the method of steepest descent if it is currently at the point $x^{(n)}$?

Solution:
The direction of greatest increase of f is given by $\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$. The method of steepest descent chooses the direction $-\nabla f(x^{(n)})$.

5 pts. 9. Briefly describe how you can use Newton's method to minimize a function $f : \mathbb{R}^n \to \mathbb{R}$. Also, what is the name of the matrix that Newton's method needs for this problem?

Solution:
A minimum point of f will be a point where all its partial derivatives are zero. This is a system of n equations:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} = 0.$$

Newton's method is then used on this system to find the vector x that satisfies it. (Such a solution could also be a maximum point of f or a saddle point, so an additional check would need to be done to ensure it is a minimum.) Newton's method needs the Jacobian of the above system; this would be a matrix of all the second partial derivatives of f. It is called the Hessian of f.

SECTION B.

ANSWER THESE QUESTIONS USING THE BUBBLE SHEET. You may tear off this page.

2 pts. 1. The basic ideal of Richardson extrapolation is

(a) to combine two evaluations of a low order method to obtain a high order method.

(b) to compute a polynomial that interpolates a set of points (x_i, y_i).

(c) to reduce step size by a factor of 1/2 to improve accuracy.

(d) to combine a low and a high order method in a way to get a more accurate answer.

(e) to compute values of the interpolating polynomial at points that were not interpolated.

Solution:
The correct answer is (a).

2 pts. 2. The value obtained from the centre difference formula for the first derivative of $f(x) = x + 1/x$ at $x = 5/4$ using a step size of $h = 1/4$ is

(a) 9/25

(b) 1/3

(c) 13/3

(d) 2/3

(e) 7/15

Solution:
The correct answer is (b).

2 pts. 3. Suppose one wishes to approximate $I = \int_a^b f(x)\,dx$ using $n+1$ evenly spaced points $x_i = a + ih$, $0 \le i \le n$, where n is even. If you interpolate each set of three consecutive points ($i \in \{0,1,2\}$, $i \in \{2,3,4\}$, etc.) and then integrate this interpolating polynommial exactly and sum up the result, this method is called

(a) Richardson extrapolation.

(b) an open Newton-Cotes formula.

(c) Simpson's rule.

(d) Trapezoid rule.

(e) Hermite interpolation.

Solution:
The correct answer is (c).

2 pts. 4. Which of the following algorithms uses a weighted average of the steepest descent algorithm and Newton's method applied to $\nabla f = 0$ in order to find the minimum of a function $f : \mathbb{R}^m \to \mathbb{R}$?

(a) Hessian method

(b) Broyden's method

(c) Levenberg-Marquardt method

(d) Nelder-Mead Simplex search

(e) Quasi-Newton method

Solution:
The correct answer is (c).

DID YOU ANSWER THE QUESTIONS IN SECTION B USING THE BUBBLE SHEET? IF NOT, YOU WILL GET ZERO FOR THEM.