Problems 1 through 6 are worth 10 points each. Problem 7 is worth 15 points for a total of 75. *All work must be shown for full credit.* Please begin each problem on a new page and label what problem you are working on. No outside resources are permitted.

1

Consider the following MATLAB code. You may assume that `a` and `b` are positive integers.

```
%beginning of function
function out = mystery(a,b)
    if a < b %want a to be greater than or equal to b
        temp = b; b = a; a = temp;
    end
    while mod(a,b) != 0 %run until a is divisible by b
        temp = b;
        b = mod(a,temp);
        a = temp;
    end
    out = b;
end
%end of function
```

a) One of the lines above has a syntax error that will not allow the function to be run. Identify the error and briefly explain how to correct it.

b) If `c = mystery(32,12)`, what is the value of `c`? Explain how you came to your answer.

Solution

a) In the while loop, the exclamation point should be replaced with a tilde. "Exclamation point equals" is a common way to denote "not equal to" in many languages, but in MATLAB it is "tilde equals."

b) `c = 4`. The algorithm (known as Euclid's algorithm) divides `a` and `b`, then sets `b` to be the new `a` the remainder to be the new `b`. The process continues until the remainder is zero. So, on the first iteration, `a` is 32 and `b` is 12. On the second iteration, `a` is 12 and `b` is 8. On the third iteration, `a` is 8 and `b` is 4. Since `a` is now divisible by `b`, the loop does not run and the output is the final value of `b`.

2

a) Using the composite trapezoid rule, numerically estimate the integral

$$\int_0^4 (x-1)^2 dx$$

using uniformly spaced nodes at each integer in the domain of integration.

b) Compute the error from using the scheme in part a. Then, give two *different* ways to more accurately estimate this integral numerically.

Solution

a) The integral is approximated by four trapezoids, each of which has width 1. The height of each side is $(x_i - 1)^2$, where x_i is the ith node.

$$\int_0^4 (x-1)^2 dx \simeq \frac{1}{2}(1+0) + \frac{1}{2}(0+1) + \frac{1}{2}(1+4) + \frac{1}{2}(4+9) = 10$$

b) The true value is

$$\int_0^4 (x-1)^2 dx = \left.\frac{(x-1)^3}{3}\right|_0^4 = \frac{3^3}{3} + \frac{1}{3} = \frac{28}{3}$$

which makes the error $10 - 28/3 = 2/3$. Possible ways to increase the accuracy include: switching to a higher order Newton-Cotes rule (e.g. Simpson's rule), switching to Gaussian quadrature, increasing the number of nodes, etc.

3

a) Define what is meant by *cubic spline* and, for these, *natural* and *not-a-knot* conditions.

b) Determine the parameters a, b, c, d, and e so that the following function $s(x)$ is a *natural* cubic spline on the interval $[0, 2]$ with knots at $x = 0, 1$, and 2.

$$s(x) = \begin{cases} a + b(x-1) + c(x-1)^2 + d(x-1)^3 & 0 \le x \le 1 \\ (x-1)^3 + ex^2 - 1 & 1 \le x \le 2 \end{cases}$$

Solution

a) A cubic spline is a piecewise interpolating polynomial of degree 3 (or lower) with continuous derivative and second derivative. Natural conditions mean we also enforce that the second derivative at the leftmost and rightmost nodes is zero. Not-a-knot conditions mean we enforce that the third derivative is continous at the second and second to last nodes.

b) Differentiate to get

$$s'(x) = \begin{cases} b + 2c(x-1) + 3d(x-1)^2 & 0 \le x \le 1 \\ 3(x-1)^2 + 2ex & 1 \le x \le 2 \end{cases}$$

$$s''(x) = \begin{cases} 2c + 6d(x-1) & 0 \le x \le 1 \\ 6(x-1) + 2e & 1 \le x \le 2 \end{cases}$$

The spline must be continuous, so plugging in $x = 1$ gives $a = e - 1$. Furthermore, the first two derivatives are continuous at $x = 1$, so $b = 2e$ and $2c = 2e$. Finally, we enfoce natural conditions so $s''(0) = 2c - 6d = 0$ and $s''(2) = 6 + 2e = 0$. The solution to this system of equations is $a = -4$, $b = -6$, $c = -3$, $d = -1$, $e = -3$.

4

Suppose $c > 0$ and we want to numerically approximate the square root of c using Newton's method. In other words, we wish to numerically find the (positive) root of $f(x) = x^2 - c$ using the scheme:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

a) Prove that the error $e_n = x_n - \sqrt{c}$ satisfies the recurrence relation

$$e_{n+1} = \frac{1}{2}\frac{e_n^2}{e_n + \sqrt{c}}$$

b) Using the relation from part a, determine the order of convergence assuming the initial guess is sufficiently close to the root.

Solution

a) Let $f(x) = x^2 - c$ so that $f'(x) = 2x$. Plugging this in,

$$x_{n+1} = x_n - \frac{x_n^2 - c}{2x_n} = \frac{1}{2}\left(x_n + \frac{c}{x_n}\right)$$

Subtract $\sqrt{c}$ from both sides to get

$$e_{n+1} = x_{n+1} - \sqrt{c} = \frac{1}{2}\left(x_n - 2\sqrt{c} + \frac{c}{x_n}\right) = \frac{1}{2}\frac{x_n^2 - 2\sqrt{c}x_n + c}{x_n} = \frac{1}{2}\frac{e_n^2}{e_n + \sqrt{c}}$$

b) $e_n > 0$ implies $e_{n+1} > 0$ by the above relation. Then, for close enough initial guess,

$$|x_{n+1} - \sqrt{c}| = |e_{n+1}| = \frac{1}{2}\frac{|e_n|^2}{|e_n + \sqrt{c}|} \leq \frac{1}{2\sqrt{c}}|x_n - \sqrt{c}|^2$$

i.e. the order of convergence is 2.

5

a) Write a MATLAB function `lagbasis(nodes,j,x)` that has inputs

- a nonempty vector of x-coordinates `nodes` (representing interpolation nodes)
- a positive integer `j` less than or equal to `length(nodes)` (representing an index for the above vector)
- a real number `x` that is not necessarily an element of `node`

and returns the value of the jth *Lagrange basis function* evaluated at x, i.e. calculates $\ell_j(x)$.

b) Write MATLAB code that would make a plot that contains a curve representing the Lagrange polynomial interpolating the points (-1,1), (0, 0), and (2,4). You may call your function from the previous part. **Bonus**: add code to label the axes of the plot (any name you want) and make the Lagrange polynomial curve red.

Solution

a) Sample code:

```
function y = lagbasis(nodes,j,x)
```

```
    y = 1;
    for i = 1:length(nodes)
        if i~= j
            y = y*(x-nodes(i))/(nodes(j)-nodes(i));
        end
    end
end
```

b) Sample code:

```
xvec = [-1, 0, 2];
x = linspace(-1,2,101);
y = zeros(1,length(x));
for j = 1:length(y)
    y(j) = -1*lagbasis(xvec,1,x(j)) + 4*lagbasis(xvec,3,x(j));
end
plot(x,y,'r'); % r stands for red
xlabel('x-axis name')l ylabel('y-axis name');
```

It is also acceptable to first calculate the Lagrange interpolating polynomial, which is $f(x) = x^2$ and just plot that directly with `plot(x,x.*x,'r')`.

6

a) Derive the fourth order accurate centered difference approximation for the first derivative of a function $f(x)$, assuming uniform grid spacing h.

b) The third order accurate expression for the first derivative of $f(x)$ given by forward differences is

$$f'(x_0) = \frac{-\frac{11}{6}f(x_0) + 3f(x_1) - \frac{3}{2}f(x_2) + \frac{1}{3}f(x_3)}{h}$$

Here, $x_j = x_0 + jh$, where h is the spacing between adjacent nodes. What is the third order accurate expression for the first derivative of $f(x)$ at $x = x_0$ given by *backward* differences?

Solution

a) The expression will look like

$$f'(x_0) \simeq \frac{A(x_{-2}) + Bf(x_{-1}) + Cf(x_0) - Bf(x_1) - Af(x_2)}{h}$$

where we have used symmetry to conclude that the $f(x_1)$ and $f(x_{-1})$ terms have equal and opposite coefficients and likewise for the $f(x_2)$ and $f(x_{-2})$ terms. Taylor expansion yields

$$\begin{aligned}
f(x_2) &= f(x_0) + 2hf'(x_0) + 2h^2 f''(x_0) + \frac{4}{3}h^3 f'''(x_0) + \frac{2}{3}h^4 f^{(4)}(x_0) + O(h^5) \\
f(x_1) &= f(x_0) + hf'(x_0) + \frac{h^2}{2} f''(x_0) + \frac{h^3}{6} f'''(x_0) + \frac{h^4}{24} f^{(4)}(x_0) + O(h^5) \\
f(x_{-1}) &= f(x_0) - hf'(x_0) + \frac{h^2}{2} f''(x_0) - \frac{h^3}{6} f'''(x_0) - \frac{h^4}{24} f^{(4)}(x_0) + O(h^5) \\
f(x_{-2}) &= f(x_0) - 2hf'(x_0) + 2h^2 f''(x_0) - \frac{4}{3}h^3 f'''(x_0) + \frac{2}{3}h^4 f^{(4)}(x_0) + O(h^5)
\end{aligned}$$

Substituting these expressions gives the system

$$A + B + C - B - A = 0$$
$$-(2hA + hB + hB + 2hA) = 1$$
$$-\left(\frac{4}{3}h^3 A + \frac{h^3}{6}B + \frac{h^3}{6}B + \frac{4}{3}h^3 A\right) = 0$$

(the other equations are trivial). The first equation implies $C = 0$. The second and third together give $A = 1/12$ and $B = -2/3$. Plug the coefficients back in to get the answer.

b) The forward difference is $\frac{1}{h}\left[f(x+h) - f(x)\right]$ while the backward difference is $\frac{1}{h}\left[f(x) - f(x-h)\right]$. Thus, if we make the transformation $h- \to -h$ and multiply by -1, we will transform a forward difference into a backward difference. The result is therefore

$$f'(x_0) \simeq \frac{\frac{11}{6}f(x_0) - 3f(x_{-1}) + \frac{3}{2}f(x_{-2}) - \frac{1}{3}f(x_{-3})}{h}$$

7

Clearly indicate if each statement is true or false. No work needs to be shown for this section.

a) The best rootfinding algorithm is always the one of highest order.
b) Two shortcomings of polynomial interpolation include Gibbs's phenomenon and aliasing error.
c) In the Gaussian quadrature approximation of $\int_a^b f(x)dx$, the sum of the weights w_i always equals $b-a$.
d) The error due to polynomial interpolation can be minimized by using Chebyshev nodes.
e) The midpoint rule and the trapezoid rule are the same order of accuracy.

Solution

a) False, higher order usually requires more information about derivatives.
b) False, these are shortcomings of *trigonometric* interpolation.
c) True, since we require the quadrature to be exact when $f(x) = 1$.
d) True, as we saw in recitation/lecture.
e) True, both are $O(h)$.

—End of exam—