AME 60614: Numerical Methods
Fall 2021

Problem Set 1
Due: September 23, 2021

Prof. Jonathan F. MacArt

Submission guidelines: A hard copy of written work is due in class. Additionally, submit through Sakai an archive (`tar` or similar) of all files requested in this assignment. Place files from individual problems into separate folders within your archive, *i.e.*, `p1/`, `p2/`, etc. Please give this archive a name of the form `lastname_firstname_ps1.tar`.

Contact `jmacart@nd.edu` with any questions.

1 Finite-Difference Schemes

For each of the following stencils:

(i) f_{i-2}, f_{i-1}, f_i, f_{i+1}, f_{i+2},

(ii) f_i, f_{i+1}, f_{i+2}, f_{i+3}, f_{i+4},

(iii) f_{i-4}, f_{i-3}, f_{i-2}, f_{i-1}, f_i,

(iv) f_{i-1}, f_i, f_{i+1}, f'_{i-1}, f'_i, f'_{i+1},

find finite-difference schemes *and* their leading-order error terms to compute each of the following three derivatives (12 schemes total). Your schemes should maximize order-of-accuracy for the stencils given.

(A) f''_i,

(B) $f_i^{(iv)}$,

(C) $f'''_i - 3f'_i$.

2 Richardson Extrapolation

Just as Richardson extrapolation is useful for developing higher-order quadrature rules, it can also be applied to the derivation of higher-order finite-difference schemes.

a. Based on the fourth-order central-difference approximation to the first derivative, use Richardson extrapolation to develop sixth-, eighth-, and tenth-order central-difference approximations to the first derivative. Your Richardson extrapolation algorithm should use successively finer grids.

b. Implement your algorithms (in the language of your choice; could be a scripting language) and verify that they achieve the desired order-of-accuracy. Choose a suitable test function, *e.g.*, $f(x) = e^{-x^2}$.

Include in your `.tar` archive your source code (in any programming language), a Makefile (if necessary), and a README.

3 Integral Equations

Along with differential equations, integral equations are important in many areas of science and engineering. An inhomogeneous Volterra integral equation of the second kind for can be (semi-)generically written as:

$$f(x) + \int_0^x K(x,t) f(t)\,\mathrm{d}t = g(x),$$

where the kernel $K(x,t)$ and the right hand side $g(x)$ are given functions.

a. Develop an algorithm to solve a generic Volterra integral equation for $f(x)$ using a numerical method of your choice. Derive the order of accuracy of this method (*i.e.*, the order of the approximation to $f(x)$).

b. Implement this algorithm to solve the integral equation with $K(x,t) = (x-t)^2$ and $g(x) = e^{-x^2}\cos(2\pi x)$ on the domain $x \in [0,1]$. Demonstrate that your numerical implementation matches your predicted order of accuracy.

Include in your `.tar` archive your code (in any programming language), a Makefile (if necessary), and a README.

4 Gauss–Hermite Quadrature

One of the most common applications of Gauss quadrature is the integration of functions against certain probability density functions. Consider integration of a function $f(x)$ against the standardized normal distribution:

$$I = \int_{-\infty}^{\infty} f(x) \left[\frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}\right] \mathrm{d}x.$$

a. Using Simpson's rule, compute the zeroeth ($f(x) = 1$), second ($f(x) = x^2$), and fourth ($f(x) = x^4$) moments of the normal distribution. (Approximately) how many quadrature points are required to compute each of these integrals to within an absolute error of 10^{-6} compared to their known values? (Their known values can be computed analytically or looked up.)

b. How many total quadrature points would be required to ***exactly*** compute the second and fourth moments using Gauss–Hermite quadrature? Verify this is the case by looking up (or computing) the weights and abscissas of Gauss–Hermite quadrature from an appropriate source (not Wikipedia!) and computing the integrals.

c. Use both Simpson's rule and Gauss quadrature to compute the integral of $f(x) = \cos(x)$, a non-polynomial function, against the standardized normal distribution. The exact integral (which is challenging to compute) is $e^{-1/2}$. For both Simpson's rule and Gauss–Hermite quadrature, determine the number of quadrature points needed to compute the integral to within an absolute error of 10^{-6}. Explain your results.

Include in your `.tar` archive your code (in any programming language), a Makefile (if necessary), and a README.