MATH 3650 Programming Assignment

Due 30 Nov. 2018

Please submit a printout of your final `R` code, as well as numerical results of any computations you do to support your answers.

1. The file `trapadapt.R` on the course website is a simple recursive implementation of adaptive quadrature based on the trapezoid rule.

   (a) Modify `trapadapt.R` to evaluate
   $$\int_0^1 e^{-x^2}\,dx$$
   to within 10^{-3}. Sketch the graph of $y = e^{-x^2}$ on $[0, 1]$ together with the trapezoids whose areas are summed to estimate the value of the integral. Verify that the error bound of 10^{-3} is achieved, e.g. by calculating the exact value of the integral via Wolfram Alpha.

   (b) Modify `trapadapt.R` (saving the result as `simpadapt.R`) to implement adaptive quadrature based on Simpson's rule.

   (c) Use `simpadapt.R` to evaluate the integral in part (a) again, and sketch the parabolas whose areas are summed to estimate the integral.

   (d) Evaluate the integral above to within 10^{-8} using both `trapadapt.R` and `simpadapt.R`. Simpson's rule results in fewer evaluations of e^{-x^2}, by what factor?

2. The file `newton_sys.R` on the course website gives an `R` implementation of Newton's method to solve the non-linear system
   $$\begin{cases} 3x^2 - y^2 = 0 \\ 2xy^2 - x^3 - 1 = 0. \end{cases}$$

   (a) Modify `newton_sys.R` so that it automatically performs as many iterations as needed to obtain a solution accurate to within 10^{-8} in each variable. The following bits of `R` syntax might be useful:
   - `while (x > 0.1) { ... }` evaluates a block of code repeatedly until the test condition fails.
   - If `x` is a vector then `max(x)` returns the component of `x` with the maximum value.
   - If `x` is a vector then `abs(x)` returns a vector whose components are the absolute values of the components of `x`.

   (b) Modify your code from problem 2a to find a solution (x_1, x_2, x_3) of the following system, accurate to within 10^{-8}:
   $$\begin{cases} x_1^3 + x_1^2 x_2 - x_1 x_3 + 6 = 0 \\ e^{x_1} + e^{x_2} - x_3 = 0 \\ x_2^2 - 2x_1 x_3 = 4 \end{cases}$$
   The `R` function `exp(x)` can be used to evaluate e^x.

3. The file `trapezoid.R` on the course website gives an `R` implementation of the composite trapezoid rule.

   (a) Determine a rigorous bound on n (the number of trapezoids) needed to evaluate $\int_{-1}^{1} e^{-x}\,dx$ with absolute error less than 10^{-4}. Use `trapezoid.R` to estimate the integral using your value of n, and confirm that the absolute error is indeed less then 10^{-4}.

(b) Modify `trapezoid.R` to implement the composite Simpson's rule, and use it to re-do part (a).

(c) Modify `trapezoid.R` to implement the 5-point Gaussian quadrature with weights and nodes found in Table 4.12 in the text. Use your code to estimate $\int_{-1}^{1} e^{-x}\,dx$ and compare the error to what you obtained using the trapezoid and Simpson's rules.

(If you're curious, you might install the `R` package `statmod`. It contains a function `gauss.quad` that returns weights and nodes for Gaussian quadrature of any order. For example, `gauss.quad(5)` returns weights and nodes for the 5-point rule above.)