Homework 10

1. a. Suppose that a polynomial of degree 200 has 100 real and simple roots in the interval $[-10, 10]$, separated one from another by a distance of at least $4 \times 10^{-3}$. Describe a simple and reliable procedure to find all the real roots to double precision (no actual programming is required).

   b. Write down the analytic expression for the n Chebyshev nodes in the interval $[1, 2]$

   c. Find out an integer that is the minimum number of Chebyshev nodes required for interpolating the Natural Logarithm $\ln(x)$ in the interval $[2, 4]$ to precision $10^{-12}$.

2. Write a Matlab script of no more than 10 lines to calculate the Newton-Cotes quadrature weights $w_j$, $j = 1, 2, \ldots, n, n + 1$; go to the end of this handout for more details about $w_j$. For uniformity, we require that the script starts with the four lines $n=2; \ h=1/n; \ nodes=[0:h:1]' ; \ f=eye(n+1) ;$ where the j-th column of the matrix f are needed for the Lagrange interpolation in order to find the Lagrange Basis functions $L_{n,j}$; see bottom for more details.

   a. Provide the script.

   b. Print off w (to 16 digits) for $n = 1$ (trapezoidal rule), $n = 2$ (Simpson’s rule), $n = 7$ (call it border-line rule), $n = 8$ (call it misery's rule) and $n = 20$ (call it the Devil’s rule)

   c. For each rule, print out the Lebesgue constant = norm(h*w,1)

   Hint: Make use of the three Matlab functions polyval, polyint, and polyfit in the order polyval(polyint(polyfit(···)))

3. Use the Matlab function fzero to write a code solving the nonlinear equation for $\alpha$ - the order of convergence (Do not provide the code).

   a. Show the nonlinear equation $f(\alpha) = 0$ used in your implementation for finding $\alpha$ (this function f is not unique, so just give your version)

   b. Apply the trapezoidal and Simpson composite quadrature rules to

   \[ \pi = \int_0^1 \frac{4}{1 + x^2} dx \]  

   to compute the approximate value for $\pi$ (see Problem 8.1, page 387).

   c. Check the order of convergence of the trapezoidal and Simpson rules with $h_1 = 1/100,$ $h_2 = 1/76 \ h_3 = 1/135$.

Appendix – Lagrange Interpolation. The j-th Lagrange Basis function associated with the $n + 1$ distinct nodes \{ $x_k$, $k = 0, 1, \ldots, n$ \} is defined by the formula

\[ L_{n,j}(x) = \prod_{k \neq j} \frac{x - x_k}{x_j - x_k} \]  

for $j = 0, 1, \ldots, n$. The interpolating polynomial $P_n(x)$ to a function $f(x)$ at the nodes \{$x_j$\} is given by the formula

\[ P_n(x) = \sum_{j=0}^{n} f(x_j)L_{n,j}(x) \]
In the associated quadrature with nodes \( \{ x_j \} \)

\[
\int_a^b f(x)dx \sim \int_a^b P_n(x)dx = \sum_{j=0}^{n} w_j f(x_j),
\]  

(4)

the weights are obviously given by

\[
w_j = \int_a^b L_{n,j}(x)dx
\]

(5)

In the special case of Newton-Cotes, \( a = x_0 \), \( b = x_n \), and \( \{ x_j \} \) are equispaced. Moreover, the weights \( w_j \) are traditionally defined as

\[
w_j = \frac{1}{h} \int_a^b L_{n,j}(x)dx
\]

(6)

and consequently, the quadrature assumes the form

\[
\int_a^b f(x)dx \sim h \sum_{j=0}^{n} w_j f(x_j)
\]

(7)