Assignment 6, Math 475A

Part I Matlab Section: (do not hand in this work).

MATLAB has special functions to deal with polynomials. Using these commands is usually recommended, since they make the code easier to write and understand and are usually more efficient. In this HW assignment you should try to use MATLAB polynomial commands (and avoid for loops) as much as possible.

The polynomial \( P(x) = 2x^2 + 2x - 4 \) and \( Q(x) = x^2 - 6 \) are represented in MATLAB by:

\[
\begin{align*}
P &= [2 2 -4]; \\
Q &= [1 0 -6];
\end{align*}
\]

\( P(4.7) \) is evaluated, for example, using:

\[
polyval(P,4.7)
\]

You can plot \( Q(x) \) in the interval \([-6, 6]\) using:

\[
x = [-6:0.1:6]; \\
plot(x,polyval(Q,x))
\]

The polynomial \( S(x) = P(x) + Q(x) \) is calculated using

\[
S = P + Q;
\]

(but addition or subtraction of polynomials with different degrees takes somewhat more effort). Multiplication is easy and the degrees do not have to be equal. The multiplication \( T(x) = P(x) \times Q(x) \) is represented by

\[
T = conv(P,Q);
\]

Additional useful commands are prod, roots (finds the roots of a polynomial) and poly (constructs a polynomial with specified roots). For vectors, roots and poly are inverse functions of each other, up to ordering, scaling, and roundoff error. For more info look at the table of the MATLAB Primer and use the online help or ask for technical assistance.

In this assignment you are asked to hand in many plots. You can save paper by combining several plots on one page using the subplot command. Use the help feature to find out how to use subplot

Part II Theory Part:

1. Construct the Lagrange interpolating polynomials for the following functions and find a bound for the absolute error on the interval \([x_0, x_n]\):

   (a) \( f(x) = \cos x + \sin x \), where \( x_0 = 0, x_1 = 0.25, x_2 = 0.5, x_3 = 1.0 \), where \( n = 3 \).

2. Find the polynomial of least degree that interpolates these sets of data:

   (a) \( (x_i, y_i) = (3, 12), (7, 146), (1, 2), (2, 1) \)
Part III Experimental Part:

1. For \( P \) and \( Q \) defined in the introduction evaluate \( T = P \ast Q \) by hand and compare with the result you get from MATLAB. Plot \( P, Q, T \) and the x-axis (e.g. using \texttt{grid on}) on the same graph. What do you observe about the roots of the three polynomials?

2. To approximate the function \( f(x) = \sqrt{x} \), we will use the points (1,1) (4,2) and (9,3).
   
   (a) Write the formula for the Lagrange Polynomial \( P_2(x) \) that interpolates these three points.
   
   (b) Write a .m function file that implements \( P_2(x) \) using the MATLAB polynomial commands.
   
   (c) To see how well the approximation is:
      
      i. Plot \( f \) and \( P_2 \) in the interval \([0.01, 12]\).
      
      ii. Plot \( f - P_2 \) in the interval \([0.01, 12]\).
      
      iii. Plot \( \text{abs}((f - P_2)/f) \) in the interval \([0.01, 12]\).
   
   (d) Using the plots determine where the approximation is better/worse.
   
   (e) Why do you get these results? Do they agree with the formula derived in class for the error bound?
   
   (f) Add the point (0,0) and repeat (a)–(c). Do you see any improvement? Deterioration? Where? Why?

3. (This problem is optional and extra credit. It will be used to boost your grade on an older assignment). We will also pretend that the answer that MATLAB provides, for \( F := e^{0.743} \) is exact (although we know that it is only good, to double precision).
   
   \begin{itemize}
   \item Using matlab (and again pretending these are exact), produce a table of data points:
     \( (x = 0, \exp(0)), (x = 0.1, \exp(0.1)), (x = 0.2, \exp(0.2)), ... \)
   \item Using this table, construct Lagrange polynomials \( p_n(x) \) of degree \( n = 0, 1, 2, 3, 4 \). You need to decide what data pairs to use (the number of data pairs required are crucial, and you need to know what that number should be).
   \item Compute the relative error of \( F \) and \( p_0(x = 0.743), p_1(x = 0.743), ..., p_4(x = 0.743) \).
   \item Plot the logarithm of the relative error vs. \( n \). What is your conclusion, based upon this plot?
   \end{itemize}

4. In this exercise you will analyze the problems that arise when the interpolation polynomial is evaluated using the Vandermonde matrix. We will find the interpolation polynomial \( P_n \) for the nice smooth function \( \sin(x) \) and see that as \( n \) increases problems arise.

To do so, run the following program (available as Vandermon.m on the class home page)

\begin{verbatim}
\% Comparison of the interpolation polynomial Pn
\% for sin(x) in the interval [1 2] with sin(x)
\% Pn(x) = c_1 x^n + ..+ c_n x + c_{n+1}
\%
\end{verbatim}

2
for i = 0:3
    Nx = 10*3^i;
    dx = 1/Nx;
    x = [1:dx:2]’;
% Find the interpolation polynomial using
% the Vandermonde matrix
    V = vander(x);
    C = V\sin(x);  % Warning: use \, not /
% Plot the results at grid values other than the
% ones used in the interpolation.
    y = x+0.1*dx*rand(size(x));
    subplot(2,2,i+1)
    plot(y,polyval(C,y)-sin(y))
    xlabel(’x’)  
    ylabel(’P_n(x)-f(x)’)  
    title([’Nx = ’,num2str(Nx)])
end
figure(gcf);

Do not submit the plots or the program, but answer the following questions:

(a) Using the error formula derived in class, show that $|\sin(x) - P_n(x)| \to 0$ as $n \to \infty$ for $x \in [0, 2]$.

(b) Does this agree with your plots?

(c) In practice, is larger $n$ good or bad?

(d) What is the source of the problem with large $n$? (Hint: Run the program again but use the grid points $x$ rather than $y$ in the plot command. Also note the MATLAB error messages.)