M3315/CSE3365
Fall 2015
Project 3 - Interpolation
Due 4pm Tue, Apr 7, 2015

The SMU honor code applies.

The completed project comprises of:

- An m-script and one m-function files which must be submitted in Blackboard under Assignments→Project 3.
- A printout of the html-published m-script that shows the code, the figures and your answers as comments. The script should be grouped into cells. One cell for each part. The print out can be either turned in before class or placed by the due date in Prof. Tausch's mailbox in Clements 208.

Part 1. Write a Matlab routine that computes the interpolating polynomial of degree $n-1$ for given data points $(x_1, f_1), .., (x_n, f_n)$. Use Lagrange polynomials (Section 4.3). Do not copy the algorithm on page 217, it does not what we want to compute here. Instead, implement formulas (7) and (9) in this section, but start your loops at $k = 1$ because Matlab does not have arrays beginning with zero. The interface to your routine should be

```
function p = lagrange(x, xNodes, fNodes)
%
% compute the Lagrange interpolating polynomial
% Parameters
%    x      points where the interpolating polynomial is to be evaluated
%    xNodes the x-values of the data points
%    fNodes the y-values of the data points
%    p      the function values of the interpolating polynomial at x
```

The routine must be able to handle multiple evaluation points. In this case `x` is a vector with the input values and `p` is an array of the same length with the output values. You can use Matlab's vectorized operations to handle these arrays efficiently. Do not pass the length of the input arrays, instead use Matlabs `length`-command inside your program to determine how many data points and evaluation points are given.

To test the correctness of your code plot an interpolating polynomial as follows:

```
xNodes = [0, 2, 3, 5, 6];
fNodes = [1, 5, 0,-3, 2];
x = linspace(-2,8,500);
y = lagrange(x, xNodes, fNodes);
plot(x,y,'b', xNodes,fNodes,'o');
```

Annotate and comment your plot. You should proceed only if the circles are on the graph of the polynomial.

Part 2. Interpolate the function $f(x) = \exp(-x^2)$ in the interval $[-\pi, \pi]$, using $N+1$ equally spaced points, which include the endpoints. Plot the interpolation errors $|f(x)-p_{10}(x)|$, $|f(x)-p_{20}(x)|$, and $|f(x)-p_{30}(x)|$ into one `semilogy` plot, where x is `linspace(-pi,pi,500)`. Note that for p_{10} you need 11 equispaced interpolation points! Explain in one sentence what happens when you use more points.

Part 3. Repeat Part 2 with the function $f(x) = 1/(1 + 25 * x^2)$ in the interval $[-1, 1]$. Is the result acceptable?

Part 4. Repeat Part 3, but this time use the interpolation points

```
xNodes = cos(linspace(0,pi,N+1));
```

Is your interpolating polynomial is acceptable? Explain.

Part 5. Create a plot similar to the one in Part 1 that shows the interpolating polynomial p_{30} and the datapoints from Part 4. Explain how the nodes are distributed in the interval.