MTH 351 Spring 2007 – Lab 3

Due: Before 5pm May 11

To see how good and bad various interpolation methods can be, use Matlab's interpolation routines on data generated from Runge's function:

$$f(x) = \frac{1}{1+x^2}.$$

In Matlab, do the following:

1. Problem setup:

   Generate $N+1 = 11$ equally-spaced nodes x_i in the interval $[-5, 5]$

   ```
   N = 10;
   x = linspace(-5,5,N+1); %to see values, omit the ;
   ```

   and then evaluate $f(x)$ at these nodes

   ```
   f = inline('1./(1+x.*x)','x');
   y = f(x);
   ```

   The $N+1$ points (x_i, y_i) are the data points to be interpolated by various methods. Plot them to see where they are

   ```
   plot(x,y,'o')
   title('N+1 = 11 equally-spaced data points')
   ```

   Also generate lots of points t_i at which to evaluate f, and the interpolants, for plotting

   ```
   t = [-5:.1:5];
   ```

   Evaluate f at these t_i's and plot $f(t)$ in a new figure window

   ```
   figure;
   plot(t,f(t),'-')
   title('Runge function')
   ```

2. Nth degree interpolating polynomial:

   Use Matlab's `polyfit` to construct (the coefficients of) the Nth degree interpolating polynomial

   ```
   PN = polyfit(x,y,N);
   ```

   Now this can be evaluated anywhere in the interval [-5,5], e.g., at the t_i's

   ```
   v = polyval(PN,t);
   ```

   Find the inf-norm error $\|f(t) - v\|_\infty$

   ```
   err = norm(f(t)-PN(t),inf)
   ```

   and plot both $f(t)$ and $PN(t)$ on the same plot as the data points

   ```
   figure;
   plot(x,y,'o',t,f(t),'-',t,v,'--')
   title(sprintf('f(t) and P_{10}(t), err=%g',err))
   ```

3. Interpolation at Chebychev nodes:

   Generate $N + 1 = 11$ Chebychev nodes

   ```
   K = N+1;
   a=-5;
   b=5;
   for i=1:K
   xcheb(i)=(a+b)/2 + (b-a)/2 * cos( (i-.5)*pi/K );
   end
   ycheb = f(xcheb);
   ```

   Follow the steps in 2. to produce the Nth degree interpolating polynomial `PNcheb` based on the Chebychev nodes, its values `vcheb` at the t_i's, and the error $\|f(t) - PNcheb(t)\|_\infty$, and plot both $f(t)$ and $PNcheb(t)$ on the same plot as the Chebychev data. Compare the error and the plot with those from 2. Comment on why one works better than the other.

4. Piecewise linear interpolation:

   Use Matlab's `interp1` to construct the piecewise linear interpolant evaluated at the t_i's

   ```
   vlin = interp1(x,y,t,'linear');
   ```

   Repeat the steps of 2. to compute the error and plot. Compare error and plot with those from the previous examples.

5. Piecewise cubic interpolation:

   Use Matlab's `interp1` to construct the piecewise cubic interpolant evaluated at the t_i's

   ```
   vcub = interp1(x,y,t,'cubic');
   ```

   Repeat the steps of 2. Compare errors and plots.

6. Cubic spline interpolation:

   Use Matlab's `interp1` to construct the cubic spline interpolant evaluated at the t_i's

   ```
   vspl = interp1(x,y,t,'spline');
   ```

   Repeat the steps of 2. Compare errors and plots.

7. To see that the error gets worse for equally-spaced nodes but not for Chebychev nodes (for this $f(x)$ at least), repeat 1., 2. and 3. with $N = 20$.

Turn in a hard copy of the results of the above commands, including plots, and your answers to all of the questions for each problem. Include a printout of the script which runs the above commands, in addition to uploading the script to Blackboard.