Matt McDade

Applied Numerical Methods HW 3

Not everything is correct, so here is what is working and what isn’t. I will submit another copy when it’s all correct.

1. Pretty sure I got everything working right here
2. Couldn’t figure out how to write out the equations by hand, but I think I got the Matlab working
3. Couldn’t figure out how to do any other nonlinear models, so I just put in the right data into Matlab
4. Calculated a matrix by hand, calculations are in the output of problem 4, but I couldn’t figure out why the points on the second spline wouldn’t line up.

**Problem 1:**

|  |  |
| --- | --- |
| CODE  f = @(x) exp(-(x.^2)); % function    % Newton  X = -5:(2/3):5; % equidistant points to compute the interpolating polynomial  Y = f(X); % corresponding y values  x = -5:0.01:5; % points to plot  y = f(x);  C = divdiff(X,Y); % computes the diagonal from the divided difference table  N = zeros(1,length(x));  for i=1:length(x)  N(i) = newtval(C,X,x(i)); % evaluates the newton polynomial at x(i)  end  subplot(1,2,1)  plot(X,Y,'o',x,y,'k',x,N,'r'),title('Newton Interpolation'),  legend('Data','Exact','Newton')  axis([-5 5 -0.01 1.2])    % Chebyshev  N = length(X); % same number of points as in Newton;  t = zeros(1,N);  c = zeros(1,N);  for k=1:N  t(k) = cos( (2\*k-1)\*pi/(2\*N) ); % Chebyshev points  c(k) = t(k)\*(5+5)/2 + (5-5)/2; % Chebyshev points in [-5,5]  end  Y = f(c);  C = divdiff(c,Y);  Ch = zeros(1,length(x));  for i=1:length(x)  Ch(i) = newtval(C,c,x(i));  end    subplot(1,2,2)  plot(c,Y,'o',x,y,'k',x,Ch,'r'),title('Chebyshev Interpolation'),  legend('Data','Exact','Chebyshev')  axis([-5 5 -0.01 1.2]) | OUTPUT |

**Problem 2:**

|  |  |
| --- | --- |
| CODE  % HW 5 Problems 2 - Example  % curve fitting    % data points  x = [0 1 2 3 4 5];  y = [4 -1 6 1 -4 -9];    % linear least squares using a quadratic model  C2 = lspoly(x,y,2); % finds the parameters  C3 = lspoly(x,y,3);  xx = 0:0.01:5;  y2 = zeros(1, length(xx));  y3 = zeros(1, length(xx));    % this loop evaluates the polynomial with the optimal parameters in the  % node points defined in x3  for i=1:length(xx)  y2(i) = horner(C2,xx(i));  y3(i) = horner(C3,xx(i));  end    % plots data points and best fitting curve  plot(x, y, 'o', xx, y2, '-', xx, y3, '--'), legend('Data','2nd Degree','3rd Degree')    polyfit(xx, y2, 2)  polyfit(xx, y3, 3) | OUTPUT |

**Problem 3:**

|  |  |
| --- | --- |
| CODE  % HW 5 Problems 3  % non-linear least squares using an exponential model f(x)=B\*exp(Ax)    % data points  x = [0 1 2 3 4 5];  y = [0.1 1 1.5 0.8 0.3 0.25];    S = @(R) sum((R(2)\*exp(R(1)\*x)-y).^2); % least squares function  initguess = [1 1]; % initial guess for  u = fminsearch(S, initguess); % finds the parameters that minimize S  yy = u(2)\*exp(u(1)\*xx);  plot(x, y, 'o', xx, yy, 'r-') | OUTPUT |

**Problem 3:**

|  |  |
| --- | --- |
| CODE  % HW 5 Problems 4  % Cubic Splines      % Create matrix  A = [1 0 0 0 0 0 0 0;  1 1 1 1 0 0 0 0;  0 0 0 0 1 0 0 0;  0 0 0 0 1 1 1 1;  0 1 2 3 0 -1 0 0;  0 0 2 6 0 0 -2 0;  0 1 0 0 0 0 0 0;  0 0 0 0 0 1 2 3;];    %Right-hand side  y = [0 1 1 .5 0 0 0 0]';    %Solving the system to find coefficients  a = A\y;    %Domain for each piece of cubic spline  x1 = 0:.01:1;  x2 = 1:.01:2;  %The three interpolating cubic splines  s1 = a(1) + a(2)\*x1 + a(3)\*x1.^2 +a(4)\*x1.^3;  s2 = a(5) + a(6)\*x2 + a(7)\*x2.^2 +a(8)\*x2.^3;    %Data points given  datax = [0, 1, 2];  datay = [0, 1, .5];    %Plot Results  plot(x1,s1,x2,s2,datax,datay,'ko','Markerfacecolor','k')  xlabel('x-values')  ylabel('y-values')  title('Cubic Spline Interpolation')  legend('S1(x)','S2(x)','Data Points')    fprintf('The coefficients are:\n')  for i = 1:8    fprintf('a(%d) = %6.1f\n',i,a(i))    end | OUTPUT |