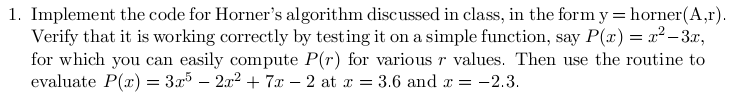
Ryan Brosnahan

Homework 4



function y = horner(a, x)

n = length(a) - 1;

y = a(n+1);

for k=n:-1:1

y = a(k) + x \* y;

end

>> a = [-2 7 -2 0 0 3]

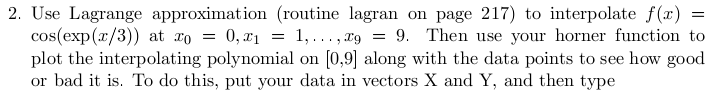
a =

-2 7 -2 0 0 3

>> horner(a, -2.3)

ans =

-221.7703



function [l,L] = lagranp(x,y)

%Input:x=[x0x1...xN], y = [y0 y1 ... yN]

%Output: l = Lagrange polynomial coefficients of degree N

% L = Lagrange coefficient polynomial

N = length(x)-1; %the degree of polynomial

l=0;

for m = 1:N + 1

P=1;

for k = 1:N + 1

if k ~= m, P = conv(P,[1 -x(k)])/(x(m)-x(k)); end

end

L(m,:) = P; %Lagrange coefficient polynomial

l=l+y(m)\*P; %Lagrange polynomial (3.1.3)

end

[C,L] = lagranp(x,y);

C = fliplr(C); % the coefficients come out backwards, so flip them around

r = 0:.01:9;

for i = 1:length(r)

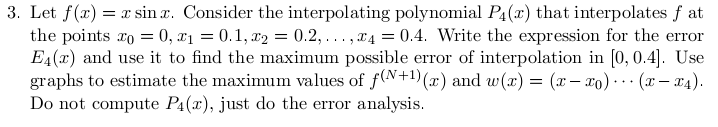
f(i) = horner(C,r(i));

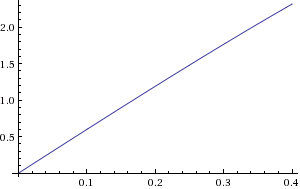
end

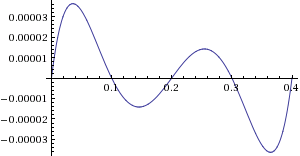
Y2 = cos(exp(r/3));

plot(x,y,'o',r,f,'k',r,Y2,'r:')

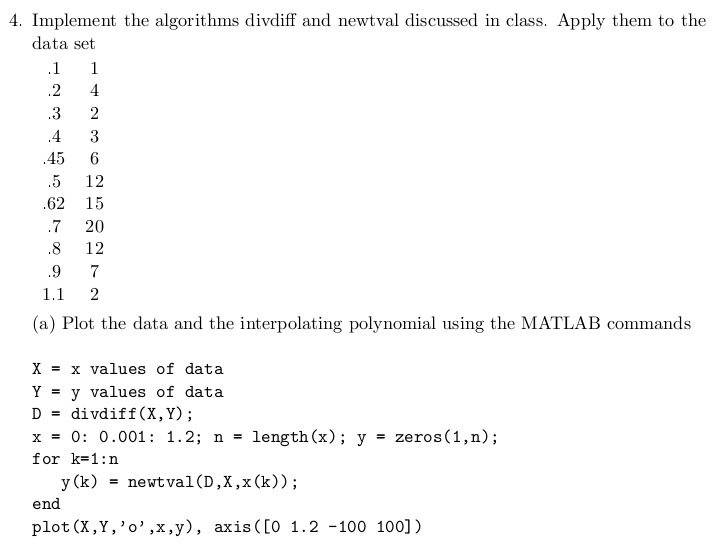








The function w(x) is symmetrical with max around x = .0356 where w(x) = .0000363143



function c = divdiff(x, y)

m = length(x);

D = zeros(m, m);

D(:,1)=y;

for j=2:m

for k=j:m

D(k, j) = (D(k,j-1) - D(k-1, j-1))/(x(k)-x(k-j+1));

end

end

c = diag(D);

function y = newtval(c, x, r)

m = length(c);

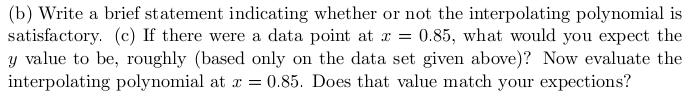
y = c(m);

for k=m-1:-1:1

y = c(k) + (r-x(k))\*y;

end





On the inner quartiles, the polynomial appears satisfactory, and any error from the “real” model could just be noise.

I would predict .85 to be a smaller in magnitude than 20 which it is: -13.3477