Supplemental 1: Find n so that degree n polynomial interpolation of $f(x) = \cos(3x)$, using equally-spaced points on $[0, 2]$, gives a maximum approximation error $|f(x) - p(x)|$ which is less than 10^{-6} on $[0, 2]$.

Then use MATLAB's polyfit and polyval and a bit of trial and error to find the actual smallest n needed to approximate $f(x) = \cos(3x)$ to within 10^{-6}.

Solution:
In class we demonstrated that given f is $n + 1$ times differentiable on $[a, b]$, $x_0, \ldots, x_n$ then there exists an $\xi \in [a, b]$ the error of an n degree polynomial interpolant,

$$|f(x) - p(x)| = f^{(n+1)}(\xi)\frac{\prod_{k=0}^{n}(x - x_k)}{(n + 1)!}$$

Finding n so that the maximum error term of $p(x)$ on the interval $[0, 2]$ is less than 10^{-6},

$$f^{(n+1)}(\xi)\frac{\prod_{k=0}^{n}(x - x_k)}{(n + 1)!} \leq 10^{-6}.$$

Note that, when differentiate $cos(3x)$, $n + 1$ times we get a trig function whose maximum value is 1 multiplied by 3^{n+1} so,

$$3^{n+1}\frac{\prod_{k=0}^{n}(x - x_k)}{(n + 1)!} \leq 10^{-6}.$$

Also note that the maximum size of each term in the product is 2 so,

$$3^{n+1}\frac{2^{n+1}}{(n + 1)!} \leq 10^{-6}.$$

Using MATLAB we get that when $n \geq 25$ then this statement is true.

Using MATLAB's polyfit and polyval approximate to approximate n by trial and error we get $n \geq 13$,
Function:

```
function [hist] = HW11_Sup(f,n)
%This function takes a function f and
% a natural number n and return the error
%in the polynomial interpolant as we add more sample points
    xx = linspace(0,2,10000);
    hist = [];

    for i = 1:n

        x = linspace(0,2,i);
```

```
        y = f(x);
        polyinter = polyfit(x,y,i);

        error = abs(f(xx)-polyval(polyinter,xx));
        hist = [hist, max(error)];

    end

end
```

Console:

```
>> hist = HW11_Sup(f,40);

>> for i = 1:20

    if (hist(i)<= 1*10^(-6))
        y = i
        break

    end
end

y =

    13
```

Text 8.9: Determine the piecewise polynomial function,

$$P(x) = \begin{cases} P_1(x) & 0 \leq x \leq 1, \\ P_2(x) & 1 \leq x \leq 2, \end{cases}$$

That is defined by the following conditions,

1. $P_1(x)$ is linear.
2. $P_2(x)$ is quadratic.
3. $P(x)$ and $P'(x)$ are continuous at $x = 1$.
4. $P(0) = 1$, $P(1) = -1$, and $P(2) = 0$.

Plot the function.

Solution:
Note that since $P(0) = 1$, $P(1) = -1$ and $P_1(x)$ is linear we know that

$$P_1(x) = -2x + 1.$$

Since $P_2(x)$ is a quadratic we know it is of the following form,

$$P_2(x) = c_2x^2 + c_1x + c_0.$$

Since $P(2) = 0$ we get the following equation,

$$0 = c_2(2)^2 + c_1(2) + c_0.$$

and since $P(x)$ and $P'(x)$ are continuous at $x = 1$ we get that

$$-1 = c_2(1)^2 + c_1(1) + c_0.$$

$$-2 = c_2(2)(1) + c_1 + (0)c_0.$$

Solving the system of equations(lazily with MATLAB) we get that

$$P_2(x) = 3x^2 - 8x + 4.$$

Finally, all together we see that,

$$P(x) = \begin{cases} -2x + 1 & 0 \le x \le 1, \\ 3x^2 - 8x + 4 & 1 \le x \le 2, \end{cases}$$

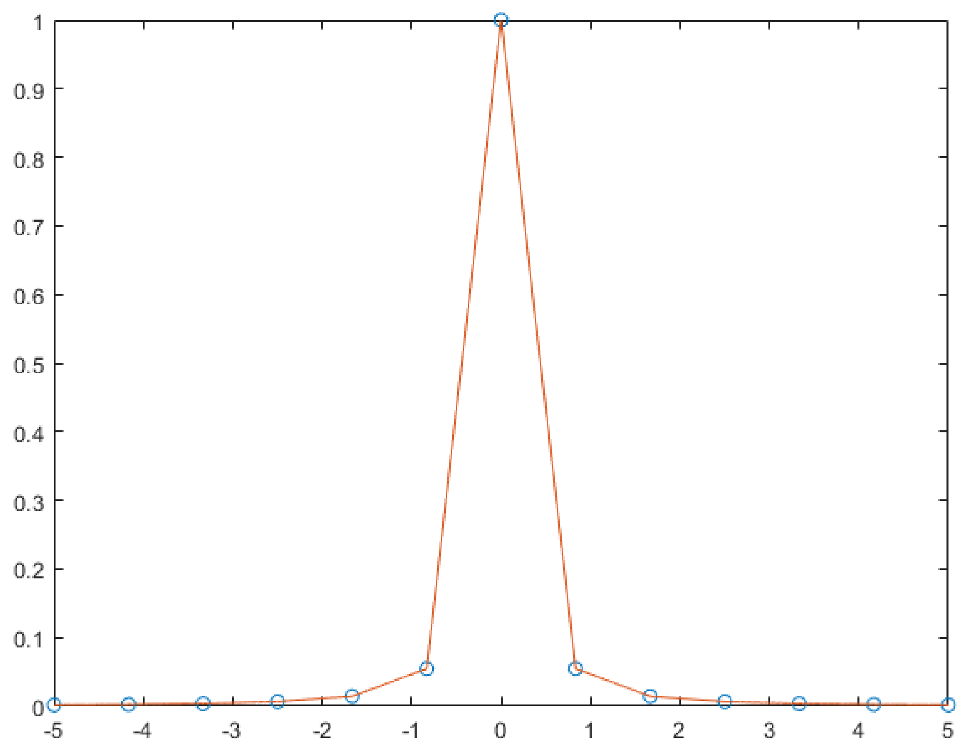
Text 8.7 (a,b): Use MATLAB to various piecewise polynomials to the Runge function, using the same nodes as in part(a) of exercise 4.

1. Find the piecewise linear interpolant of $f(x)$. (You may useMATLAB routine interp1 or write your own routine.)

   Solution:
   Console:

```
%Sample points from question 4.a
>> x = linspace(-5,5,13);

%Runge function
>> f = @(x) 1 ./ (1 + 25.*x.^2);

%set linspace to plot interpolant
>> xx = linspace(-5,5,10000);

%Calculate interpolant value on xx
>> vq = interp1(x,f(x),xx);


>> plot(x,f(x),'o',xx,vq);
```

Figure 1: Plot of linear interpolation for the Runge function with 13 sample points



2. Find the piecewise cubic Hermite interpolant of $f(x)$. (Write your own routine for this, using formulas (8.17) and (8.18).)

Solution:

Function:

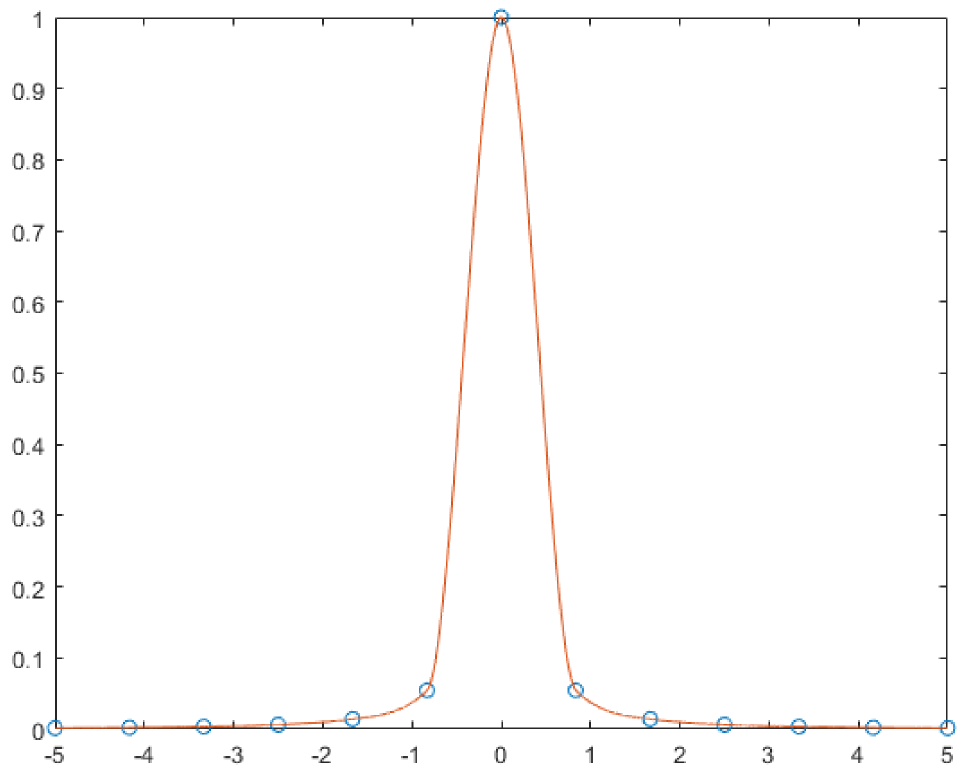
```
function px = cubicHermite(sx,sy,sdy,xl)
% This function takes sx sample points, sy the value
% at those sample points, sdy the value of the derivative
% at those sample points, and a linespace xl and returns
% the value of the piecewise cubicHermite Interpolation

%initilizing piecewise linespace
xx = [];

%Intitilizing return vector
px = [];

%Iterating through sample points
for i = 2:length(sx)

    %Creating trucated linspace xx
    for j = 1:length(xl)
        if (sx(i-1)<= xl(j) && sx(i) > xl(j))
            xx =[xx xl(j)];
            if (sx(length(sx)) == xl(j))
                xx =[xx xl(j)];
            end
            if (sx(1) == xl(j))
                xx =[xx xl(j)];
            end
        end
    end

    %Caluculating P(x) on xx linspace with proper sample interval
    h = sx(i) - sx(i-1);
    AA = (3/h^2)*(sdy(i-1)+sdy(i)) + (6/h^3)*(sy(i-1)-sy(i));

    px_i = (-sdy(i-1)/h).*((((xx - sx(i)).^2)./2) - (h^2/2)) +
    ((sdy(i)/h).*(((xx - sx(i-1)).^2)./2)) +
    AA.*((xx - sx(i-1)).^2).*(((xx - sx(i-1))./3) - h/2) + sy(i-1);

    %Storing P(x)
    px = [px px_i];

%Clearing truncated linspace
xx = [];
```

```
end

end
```

Console:

```
>> f = @(x) 1 ./ (1 + 25.*x.^2)

f =

  function_handle with value:

    @(x)1./(1+25.*x.^2)


>> fd = @(x) (-50.*x)./((25.*x.^2 + 1).^2)

fd =

  function_handle with value:

    @(x)(-50.*x)./((25.*x.^2+1).^2)

>> x = linspace(-5,5,13);
>> xx = linspace(-5,5,10000);

>> px = cubicHermite(x',f(x)',fd(x)',xx');

>> plot(x,f(x),'o',xx, px);
```

Figure 2: Plot of cubicHermite interpolation for the Runge function with 13 sample points



Text 8.8: Computer libraries often use tables of function values together with piecewise linear interpolation to evaluate elementary functions such as *sinx*, because table lookup and interpolation can be faster than, say,using a Taylor series expansion.

1. In MATLAB create a x of 1000 uniformly spaced values between 0 and π . Then create a vector y with the values of the sine function at each of these points.
Console:

```
>> x = linspace(0,pi,1000);
>> y = sin(x);
```

2. Next create a vector r of 100 randomly distributed values between 0 and π. (This can be done in MATLAB by typing $r = pirand(100, 1)$;.) Estimate $sin(r)$ as follows:

For each value $r(j)$,find the two consecutive x entries, $x(i)$ and $x(i + 1)$ that satisfy $x(i) \leq r(j) \leq x(i + 1)$. Having identified the sub-interval that contains $r(j)$, use linear interpolation to estimate $sin(r(j))$. Compare your results with those returned by MATLAB when you type $sin(r)$. Find the maximum absolute error and the maximum relative error in your results.

Console:

```
r = pi* rand(100,1);
>> X = HW8_8(x,r); %Scans through r and pulls the sample intervals

>> X = unique(X); % HW8_8 function produces duplicates
>> X = sort(X); % interp1 requires a sorted list of sample points
>> Y = sin(X);

>> vq = interp1(X,Y,x'); %finding linear interpolation over linspace

>> estimate = interp1(X,Y,r); %Finding estimated values of sin(r(j))


%Plot of our linear interpolation + estimates of sin(r(j))
%+ sample points x

>> plot(X,Y,'o',r,estimate,'x',x',vq,':.');

% Error calculation/ finding max errors of our interpolation

>> error = abs(estimate - sin(r));
>> max(error)

ans =

     1.211551538315980e-06



>> error_relative = abs((estimate - sin(r))./sin(r));

>> max(error_relative)

ans =

     1.235656166677914e-06
```

Function HW88:

```
function [I] = HW8_8(x,r)
%This function takes in a linspace, x and a vector r,
%and returns a vector of valus in x that surround r, ie
%x(i)<r(j)<x(i+1)




n = length(x);
m = length(r);
I = [];
for i = 1:m
    for j = 1:n
        if (x(j)>= r(i))
            I = [I ; x(j-1);x(j)];
            break
        end
    end
end


end
```

Figure 3: Plot of linear interpolation with x sample points

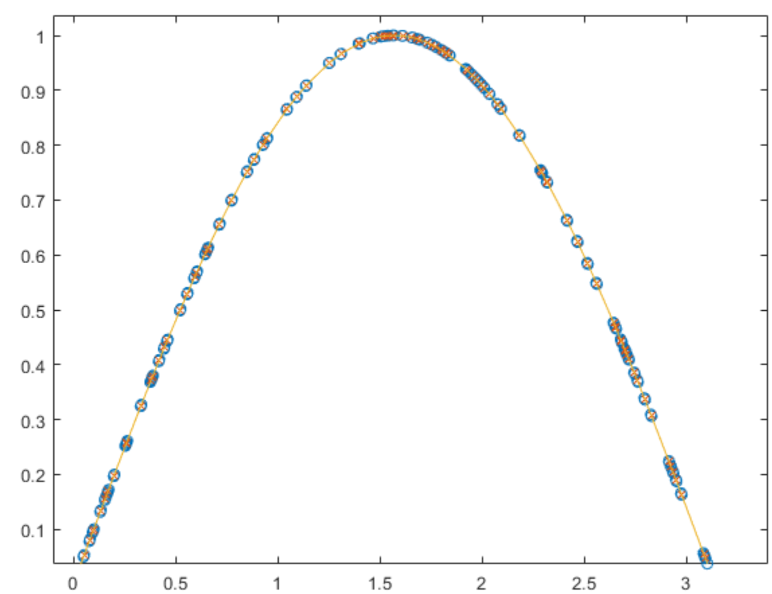
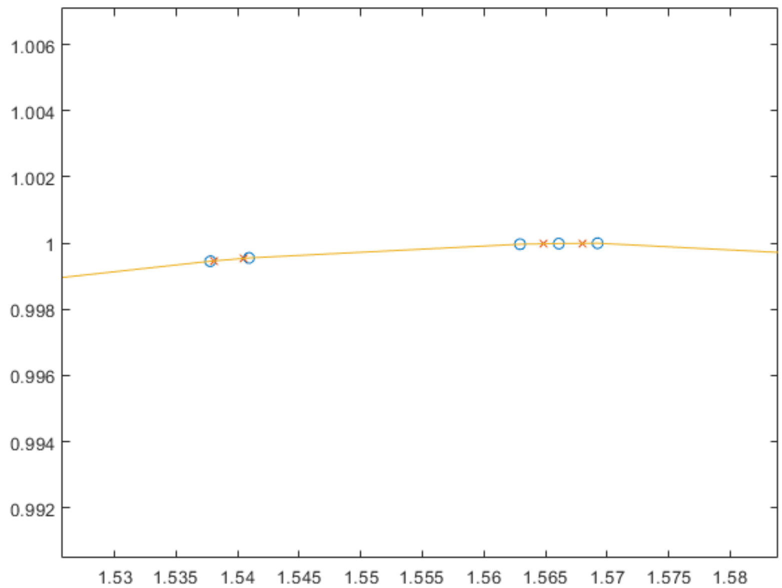


Figure 4: Linear interpolation zoomed in, O are sample points, x are $sin(r(j))$ approximation



Supplemental 2: At the bottom of page 198 is an inequality that describes the error from the piecewise-linear interpolant $\ell(x)$ for $f(x)$ on $[a, b]$. Suppose we have equally spaced points $a = x_0 < x_1 < \cdots < x_n = b$ with spacing $h = x_i - x_{i-1}$. Then:

$$|f(x) - \ell(x)| \leq \frac{Mh^2}{8}$$

for all $x \in [a, b]$. In this inequality we are assuming $f''(x)$ exists and is bounded by the number M, so that $|f''(x)| \leq M$ for all $x \in [a, b]$. Use this inequality to find n so that $|f(x) - l(x)| \leq 10^{-6}$ for $x \in [0, 2]$ if $f(x) = \cos(3x)$.

Solution:
Note that with evenly spaced points the value for $h = \frac{b-a}{n}$. Therefore we seek to solve the inequality,

$$\frac{M(\frac{2}{n})^2}{8} \leq 10^{-6}.$$

Note that the second derivative of $f(x) = cos(3x)$, is $f''(x) = -9cos(3x)$. Since $|cos(3x)| \leq$

1 we know that $M = 9$. Solving the inequality for n,

$$\begin{aligned}
\frac{9(\frac{2}{n})^2}{8} &\leq 10^{-6}, \\
\frac{36}{8n^2} &\leq 10^{-6}, \\
\sqrt{\frac{36 * 10^6}{8}} &\leq n, \\
2122 &\leq n.
\end{aligned}$$