Exercise : Consider these three points: $\{(1,1),(2.5,8),(4,5)\}$. Find the polynomial $P(x)$ of degree 2 which passes through these points. Do this three different ways, by using

(a) the Vandermonde matrix method,

```
>> x=[1,2.5,8]';
>> y=[1,8,5]';
>> v=vander(x)
v =

    1.0000    1.0000    1.0000
    6.2500    2.5000    1.0000
   64.0000    8.0000    1.0000

>> c=v\y
c =

  -0.74459
   7.27273
  -5.52814

>> P=@(x) c(1)*x.^2+c(2)*x+c(3)
P =

@(x) c (1) * x .^ 2 + c (2) * x + c (3)

>> P(x)
ans =

   1.0000
   8.0000
   5.0000

>> y
y =

   1
   8
   5

>> diary off
```

(b) the Newton form and its triangular matrix method

```
>> A=[1,0,0;1,x(2)-x(1),0;1,x(3)-x(1),(x(3)-x(1))*(x(3)-x(2))]
A =

    1.00000    0.00000    0.00000
    1.00000    1.50000    0.00000
```

```
    1.00000    7.00000   38.50000

>> a=A\y
a =

   1.00000
   4.66667
  -0.74459

>> P=@(x) a(1)+a(2)*(x-x(1))+a(3).*(x-x(1)).*(x-x(2))
P =

@(x) a (1) + a (2) * (x - x (1)) + a (3) .* (x - x (1)) .* (x - x (2))

>> P(x)
ans =

   1.0000
   8.0000
   5.0000

>> y
y =

   1
   8
   5

>> diary off
```

In standard form:

```
>> c=[a(3),-x(1)*a(3)-x(2)*a(3)+a(2),x(1)*x(2)*a(3)-x(1)*a(2)+a(1)]
c =

  -0.74459   7.27273  -5.52814

>> diary off
```

These are exactly the same coefficients as part a.

(c) the Lagrange form

```
>> t1=@(X) y(1)*((X-x(2)).*(X-x(3)))./((x(1)-x(2)).*(x(1)-x(3)))
t1 =

@(X) y (1) * ((X - x (2)) .* (X - x (3))) ./ ((x (1) - x (2)) .* (x (1) - x (3)))

>> t2=@(X) y(2)*((X-x(1)).*(X-x(3)))./((x(2)-x(1)).*(x(2)-x(3)))
t2 =
```

```

@(X) y (2) * ((X - x (1)) .* (X - x (3))) ./ ((x (2) - x (1)) .* (x (2) - x (3)))

>> t3=@(X) y(3)*((X-x(1)).*(X-x(2)))./((x(3)-x(1)).*(x(3)-x(2)))
t3 =

@(X) y (3) * ((X - x (1)) .* (X - x (2))) ./ ((x (3) - x (1)) .* (x (3) - x (2)))

>> P=@(X) t1(X)+t2(X)+t3(X)
P =

@(X) t1 (X) + t2 (X) + t3 (X)

>> P(x)
ans =

   1
   8
   5

>> diary off
```

In standard form:

```
>> c(1)=y(1)/((x(1)-x(2)).*(x(1)-x(3)))
c =

   0.095238   0.000000   0.000000

>> c(1) += y(2)/((x(2)-x(1)).*(x(2)-x(3)))
c =

  -0.87446   0.00000   0.00000

>> c(1) += y(3)/((x(3)-x(1)).*(x(3)-x(2)))
c =

  -0.74459   0.00000   0.00000

>> diary off
>> c(2)=y(1)*(-x(2)-x(3))./((x(1)-x(2)).*(x(1)-x(3)))
c =

  -0.74459  -1.00000   0.00000

>> c(2) += y(2)*(-x(1)-x(3))./((x(2)-x(1)).*(x(2)-x(3)))
c =

  -0.74459   7.72727   0.00000

>> c(2) += y(3)*(-x(1)-x(2))./((x(3)-x(1)).*(x(3)-x(2)))
```

```
c =

  -0.74459   7.27273   0.00000

>> diary off
>> c(3)=y(1)*(x(2)*x(3))./((x(1)-x(2)).*(x(1)-x(3)))
c =

  -0.74459   7.27273   1.90476

>> c(3) += y(2)*(x(1)*x(3))./((x(2)-x(1)).*(x(2)-x(3)))
c =

  -0.74459   7.27273  -5.85281

>> c(3) += y(3)*(x(1)*x(2))./((x(3)-x(1)).*(x(3)-x(2)))
c =

  -0.74459   7.27273  -5.52814

>> diary off
```

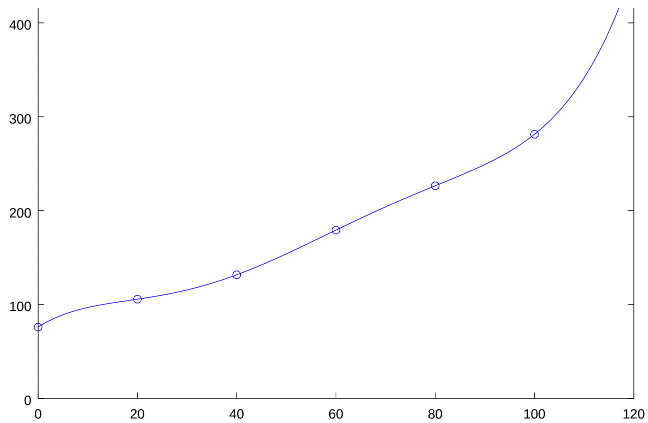
These are exactly the same coefficients as part a.

(d) True as shown above.

Exercise : 8.1

(a) The estimated value at 120 is 459.60. I feel like this estimate is way too high, a better estimate would be around 300, following a best fit line, as the data appears to be linear.

```
year=[0:20:100]';
pop=[76.0,105.7,131.7,179.3,226.5,281.4]';
plot(year,pop,'o');
v=vander(year);
c=v\pop;
x=[0:.5:120];
hold on;
plot(x,polyval(c,x));
polyval(c,120)
hold off;
```



(b) the newton form constants are independent of the number of additional points, so the zero order constant is the first constant of the first order and the first constant of the second order. Thus I can simply give the second order constants and the first two are the first order constants and the first one is the zero order constant.

```
x=[0:20:40]';
y=[76.0,105.7,131.7]';
t1=@(X) y(1)*((X-x(2)).*(X-x(3)))./((x(1)-x(2)).*(x(1)-x(3)));
t2=@(X) y(2)*((X-x(1)).*(X-x(3)))./((x(2)-x(1)).*(x(2)-x(3)));
t3=@(X) y(3)*((X-x(1)).*(X-x(2)))./((x(3)-x(1)).*(x(3)-x(2)));
L=@(X) t1(X)+t2(X)+t3(X);
#this is the Lagrange form

A=[1,0,0;1,x(2)-x(1),0;1,x(3)-x(1),(x(3)-x(1))*(x(3)-x(2))];
a=A\y
N=@(X) a(1)+a(2)*(X-x(1))+a(3).*(X-x(1)).*(X-x(2));
#this is the newton form

```

```
#to quickly verifi that they are the same
r=(rand(20000,1)-.5)*400;
l=L(r);
n=N(r);
dif=abs(l-n);
errorval=sum(dif)
```

```
>> q81b
a =

   7.6000e+01
   1.4850e+00
  -4.6250e-03

errorval =    6.0952e-09
>> diary off
```

Noting that the sum of the absolute error is extremely small, order 10^{-9}, for 20000 points we can say that the two are identical, within machine error.

Exercise : 8.2

The Lagrange form of the third order polynomial would be

$$P(x) = \sum_{i=k-3}^{k} f(i) * \prod_{j \in [k-3,k]:\, j \neq i} \frac{x - x_j}{x_i - x_j}$$

```
>> f=@(x) x.^3-2;
>> x=[0,1,2]';
>> y=f(x);
>> v=vander(x);
>> c=v\y;
>> si=[-1,1];
>> tx=(-c(2)+si*sqrt(c(2)^2-4*c(1)*c(3)))/(2*c(1))
tx =

  -0.54858   1.21525

>> diary off
```

since 1.21525 is closer to 2, it is our next x value.

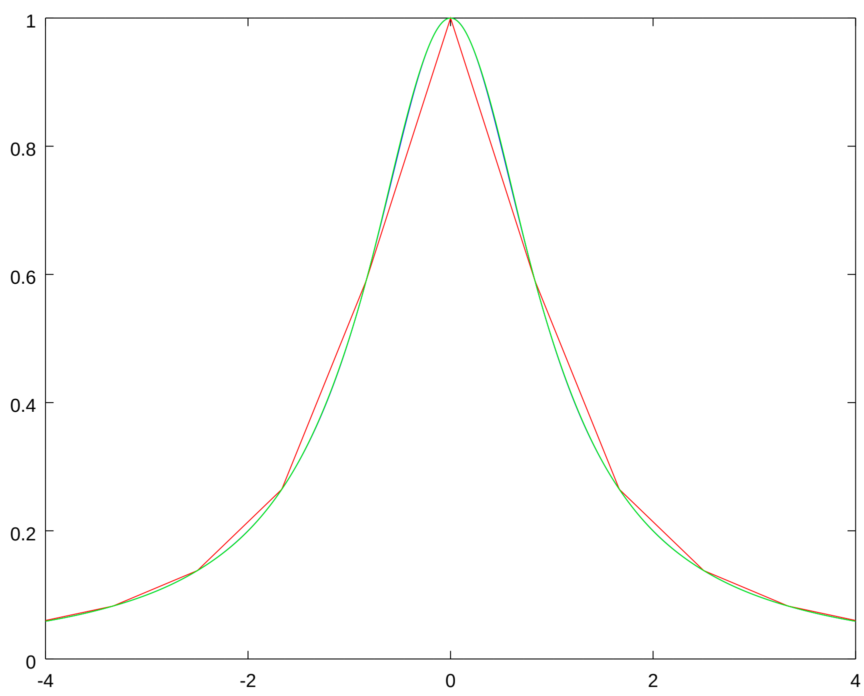
Exercise : 8.7

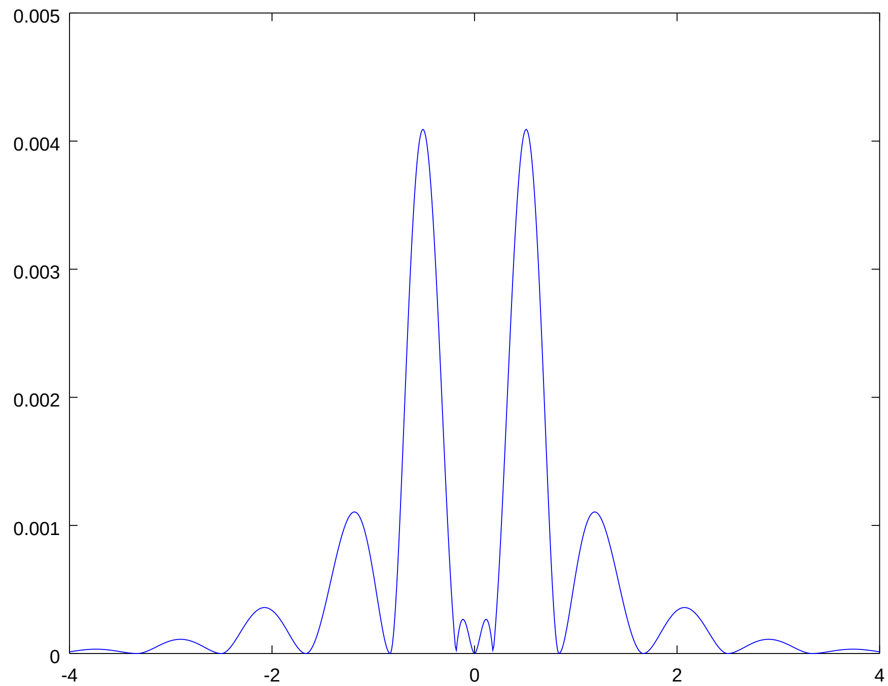
```
R=@(x) 1./(x.^2+1);
px=[-4:.01:4]';
```

```
plot(px,R(px));
x=linspace(-5,5,13);
fx=R(x);
y=interp1(x,fx,px);
hold on;
plot(px,y,'r-');
h=12/10
dxR=@(x) -2*x./(x.^2+1).^2;
fpx=dxR(x);
for i=[1:length(px)]
    y(i)=herm(x,fx,fpx,px(i));
end
plot(px,y,'g-');
hold off;
figure(2);
plot(px,abs(y-R(px)));
```

```
function yout=herm(x,y,yp,xeval)
i=2;
while(true)
    if(x(i-1)<=xeval && x(i)>xeval)
        break;
    end
    i+=1;
end
h=x(i)-x(i-1);
alph=3/h^2*(yp(i-1)+yp(i))+6/h^3*(y(i-1)-y(i));
yout=-yp(i-1)/h*((xeval-x(i))^2/2-h^2/2)+yp(i)/h*(xeval-x(i-1))^2/2;
yout+=alph*(xeval-x(i-1))^2*((xeval-x(i-1))/3-h/2)+y(i-1);
endfunction
```

I will present my results as a graph, however the results for the Hermite polynomial are so close that one would need a zoom to make out the difference. Thus I also give a difference plot between the Hermite polynomial and the function.


1
0.8
0.6
0.4
0.2
0
-4
-2
0
2
4



Exercise : 8.8

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

```
r=rand(100,1)*pi;
ry=sin(r);
Rest=r;
for i=[1:length(r)]
    Rest(i)=rest(x,y,r(i));
end
abserr=max(abs(Rest-ry))
relerr=max(abs((Rest-ry)./ry))
```

```
function yest=rest(x,y,r)
i=2;
while(true)
    if(x(i-1)<=r && r<x(i))
        break;
    end
    i+=1;
end
m=(y(i-1)-y(i))/(x(i-1)-x(i));
yest=m*(r-x(i))+y(i);
endfunction
```

```
>> q88
abserr =    1.2278e-06
relerr =    1.2361e-06
>> q88
abserr =    1.2091e-06
relerr =    1.2326e-06
>> q88
abserr =    1.2149e-06
relerr =    1.2361e-06
>> q88
abserr =    1.2095e-06
relerr =    1.2361e-06
>> q88
abserr =    1.2048e-06
relerr =    1.2362e-06
>> diary off
```

Exercise : 7.14

```
x=linspace(0,1,50)';
y=cos(4*x);
v=vander(x',10);
#note we want v*c=y, however this is imposible, as c does not have enough variables
c=v\y;
c2=(v'*v)\(v'*y);
format long e
```

```
c
c2
err=abs(c-c2)
```

```
>> q714
c =

  -5.95484314049769e-01
   2.07303158488186e+00
   8.48610575052772e-03
  -5.93544628891181e+00
   1.82203937403852e-01
   1.06028268880602e+01
   1.17638892210016e-02
  -8.00106220766967e+00
   3.67105073956209e-05
   9.99999815646296e-01

c2 =

  -5.95531446839227e-01
   2.07324609938819e+00
   8.07783930992307e-03
  -5.93502522532791e+00
   1.81949130193354e-01
   1.06029183399554e+01
   1.17451885163276e-02
  -8.00106024972344e+00
   3.66287854458158e-05
   9.99999816219448e-01

err =

   4.71327894575602e-05
   2.14514506335473e-04
   4.08266440604644e-04
   4.21063583897485e-04
   2.54807210497449e-04
   9.14518952388477e-05
   1.87007046739979e-05
   1.95794623181200e-06
   8.17219498050814e-08
   5.73151748284317e-10

>> diary off
```

The errors for the low order constants have more error then the high order ones.