

Lecture 10: Data Fitting with MATLAB

```
% Load the data
load linefit.dat

% Restructure the data into two vectors
x=linefit(:,1);
y=linefit(:,2);

% Graph data
figure(1)
set(gca,'FontSize',20)
plot(x,y,'0:')
```

```
% Lets explore Least-Squares fitting
% with a polynomial of order n = 1 (a line)
% which should give us two coefficients
% a1 and a0 of the line fit: a1x + a0
pcoeff = polyfit(x,y,1)

% Pick points where you want to interpolate
xp=0:0.1:7;

% Evaluate the polynomial at points in xp
yp=polyval(pcoeff,xp);

% Take a look at the values at the interpolated
% values with the original data
figure(2), plot(x,y,'o',xp,yp,'g*-')

% Now lets try a polynomial of degree n = 2,
% which is parabolic
pcoeff2=polyfit(x,y,2);
yp2=polyval(pcoeff2,xp);
figure(3), plot(x,y,'o',xp,yp2,'m*-')
```

To calculate the actual least squares error the sum of the differences between the parabolic fit and the actual data must be evaluated:

$$E_2(f) = \left(\frac{1}{n} \sum_{k=1}^n |f(x_k) - y_k|^2 \right)^{1/2} \quad (1)$$

which is evaluated at the original data points

```
% Lets compute the least squares error in MATLAB  
n = length(x)  
yp3=polyval(pcoeff2,x);  
E2=sqrt( sum( ( abs(yp3-y) ) .^2 )/n )
```

```
% Interpolating the data (x,y) onto the  
% points xp with a polynomial of degree n-1  
% (given n data points an n-1 degree polynomial  
% is chosen (see theory from last two lectures)
```

```
pcoeffn=polyfit(x,y,n-1);  
ypn=polyval(pcoeffn,xp);  
figure(4), plot(x,y,'o',xp,ypn,'m')
```

What we see in figure 4 is polynomial wiggle which we discussed previously and is tricky to avoid with polynomial fits when the data is nonlinear.

```
% Interpolating the data (x,y) onto the points  
% xp with the following simple (but choppy)  
% piecewise linear fit  
yint=interp1(x,y,xp);  
figure(5), plot(x,y,'o',xp,yint,'m')
```

```
% Options within the function 'interp1'  
help interp1
```

```
% Lets compare
```

```
yint=interp1(x,y,xp);
```

```
yint2=interp1(x,y,xp,'nearest');
```

```
yint3=interp1(x,y,xp,'spline');
```

```
figure(5),
```

```
plot(x,y,'0',xp,yint,'m',xp,yint2,'k',xp,yint3,'r')
```

```
% interp1 with the spline option is equivalent
```

```
% to the spline function within MATLAB
```

```
yspline=spline(x,y,xp);
```

```
figure(6), plot(x,y,'0',xp,yspline,'r')
```

```
% Lets try to fit some nonlinear data in the
% file gaussfit.dat with a nonlinear least squares
% fit

% First look at the data
load gaussfit.dat
x2=gaussfit(:,1) ;
y2=gaussfit(:,2) ;
figure(5),
set(gca,'FontSize',20)
plot(x2,y2,'o')

% Now we will use a function called fminsearch by
% minimizing the least squares

% So we need a function that we will minimize
edit gafit.m

% Now we minimize using the function gafit(x0) with
% an initial guess of x0 = [1,1]
coeff=fminsearch('gafit',[1 1]);
a=coeff(1); b=coeff(2);
```

```
% To see the resulting interpolating function on a  
% particular set of (interpolating/extrapolating)  
% points x_new
```

```
x_new = -4:.1:4;  
y_new=a*exp(-b*x_new.^2);  
figure(6),  
set(gca,'FontSize',20)  
plot(x2,y2,'0',x_new,y_new,'b')
```