Lecture 5 Using Linear Methods for Nonlinear Prediction Models and Multiple Linear Prediction

***Example* 1.** Very often, engineers assume that a spring has a *linear* force-displacement relationship; that is, $F = kX$. This assumption relies on the assumption that the spring will not be stretched too far (i.e. an amount that is on the order of the total length of the coils, for example). In order to obtain a model for a newly designed spring that can be used over a large range of displacements, 100 springs were selected. Each spring was subjected to a randomly chosen amount of displacement, and the resulting force was recorded. A scatter plot of the results is shown at right.

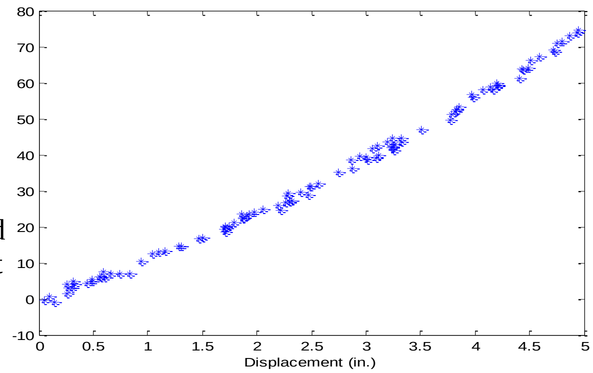


Figure 1.1 Force-displacement data for 100 springs.

(a) Consider first the following linear model: $\hat{F} = kX$. Notice that this model includes a slope parameter, but no force-intercept parameter. This is because physics dictates that for zero displacement there must be zero force. To obtain an estimate of the spring rate, k, we will require that $\hat{F}$ be an *unbiased* predictor of F; that is:

$$E(\hat{F}) = E(kX) = kE(X) \equiv E(F). \tag{1a}$$

The condition (1a) results in: $$k = E(F)/E(X) \overset{\Delta}{=} \mu_F / \mu_X . \tag{1b}$$

If we assume that we have no prior knowledge of the means, then we must estimate them from the data. We will use the sample means for this purpose. Thus, our *estimator* of k is:

$$\hat{k} = \hat{\mu}_F / \hat{\mu}_X \text{ where } \hat{\mu}_F \overset{\Delta}{=} \frac{1}{100}\sum_{j=1}^{100} F_j \text{ and } \hat{\mu}_X \overset{\Delta}{=} \frac{1}{100}\sum_{j=1}^{100} X_j .$$

From the given data, we obtain the *estimates* $\hat{\mu}_X$ =2.5075 *in*. and $\hat{\mu}_F$ =33.2589 lb_f. Hence, our *estimate* corresponding to the *estimator* (1c) is $\hat{k}$ =13.2638 lb_f/in. The model (5.17a) for this estimate of k is shown at right.

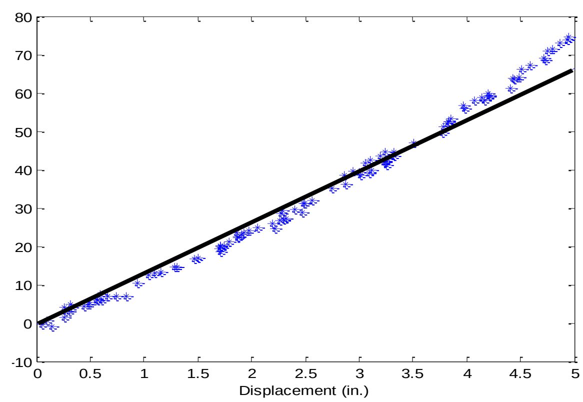


Figure 1.2 The model $\hat{F} = 13.2368X$.

Remark 1. We could have just as easily restricted our model to the region of the scatter plot that appears to suggest a linear relationship between X and F. At higher spring displacement, x, the linear model underestimates the associated force. Every spring has this property! When the spring coils are stretched further and further, the required force becomes higher and higher in a power-law fashion. The fact is, a spring cannot be stretched any further than the total length of the coils no matter how much force is applied.

(b) We now consider the quadratic model $\hat{F} = aX^2 + bX + c$. Notice that we have chosen here to include the constant, c, expecting that the model will be good enough that we will find $c \cong 0$. This additional parameter will allow us to use the more standard approach to arriving at the model parameter estimate. Define $Y \overset{\Delta}{=} X^2$. Then the model becomes

$$\hat{F} = aY + bX + c \tag{2}$$

<u>Condition 1</u>: $E(\hat{F}) = E(F)$ (i.e. $\hat{F}$ is an *unbiased* estimator for F).

This condition results in: $$E(\hat{F}) = aE(Y) + bE(X) + c = E(F). \tag{3a}$$

Using our notation for means, (3a) becomes: $c = \mu_F - a\mu_Y - b\mu_X$. (3b)

Condition 2: $Cov(F - \hat{F}, X) = Cov(F - \hat{F}, Y) = 0$.

In words, we require that the prediction error be uncorrelated with both X and Y. This gives the following two equations:

$$Cov(F, X) = Cov(\hat{F}, X) = Cov(bX + aY + c, X) = bCov(X, X) + aCov(X, Y) . \quad (4a)$$

$$Cov(F, Y) = Cov(\hat{F}, Y) = Cov(bX + aY + c, Y) = bCov(Y, X) + aCov(Y, Y) . \quad (4b)$$

Using our notation for variances and covariances, (4a-b) become:

$$\begin{bmatrix} \sigma_X^2 & \sigma_{XY} \\ \sigma_{XY} & \sigma_Y^2 \end{bmatrix} \begin{bmatrix} b \\ a \end{bmatrix} = \begin{bmatrix} \sigma_{FX} \\ \sigma_{FY} \end{bmatrix} . \quad (4c)$$

Clearly, the solution for the unknown parameters a and b is then:

$$\begin{bmatrix} b \\ a \end{bmatrix} = \begin{bmatrix} \sigma_X^2 & \sigma_{XY} \\ \sigma_{XY} & \sigma_Y^2 \end{bmatrix}^{-1} \begin{bmatrix} \sigma_{FX} \\ \sigma_{FY} \end{bmatrix} . \quad (4d)$$

Computing Estimates of a, b, and c:

Let $\mathbf{D} \overset{\Delta}{=} (X, Y, F)$. Then $\boldsymbol{\mu}_\mathbf{D} \overset{\Delta}{=} (\mu_X, \mu_Y, \mu_F)$ and $\boldsymbol{\Sigma}_\mathbf{D} \overset{\Delta}{=} \begin{bmatrix} \sigma_X^2 & \sigma_{XY} & \sigma_{XF} \\ \sigma_{XY} & \sigma_Y^2 & \sigma_{YF} \\ \sigma_{XF} & \sigma_{YF} & \sigma_F^2 \end{bmatrix}$. Now, define the arrays

$\boldsymbol{\Sigma}_{(X,Y)} \overset{\Delta}{=} \begin{bmatrix} \sigma_X^2 & \sigma_{XY} \\ \sigma_{XY} & \sigma_Y^2 \end{bmatrix}$ and $\boldsymbol{\Psi}_{(X,Y)} \overset{\Delta}{=} \begin{bmatrix} \sigma_{XF} \\ \sigma_{YF} \end{bmatrix}$. Then (4d) becomes: $\begin{bmatrix} b \\ a \end{bmatrix} = \Sigma_{(X,Y)}^{-1} \boldsymbol{\Psi}_{(X,Y)}$. (5a)

From $\boldsymbol{\mu}_\mathbf{D} \overset{\Delta}{=} (\mu_X, \mu_Y, \mu_F)$, define the array $\boldsymbol{\mu}_{(X,Y)} \overset{\Delta}{=} [\mu_X \ \ \mu_Y]$. Then (3b) becomes: $c = \mu_F - \boldsymbol{\mu}_{(X,Y)} \begin{bmatrix} b \\ a \end{bmatrix}$. (5b)

Equations (5) provide a very simple procedure for estimating the model parameters from the 100×3 data array, **d**:

```
y=x.^2;
d=[x,y,f];
C=cov(d);
Cxy=C(1:2,1:2);
Sxy=C(1:2,3);
ba=Cxy^-1*Sxy;
muhatD=mean(d);
c=muhatD(3)-muhatD(1:2)*ba;
fhat=ba(2)*y+ba(1)*x+c;
%-----------------------
%To plot prediction line use specified x-values:
xo=0:.1:5;
yo=xo.^2;
fhato=ba(2)*yo+ba(1)*xo+c;
figure(2)
plot(x,f,'*')
hold on
plot(xo,fhato,'k','LineWidth',2)
xlabel('Displacement (in.)')
ylabel('Force (lb_f)')
title('Nonlinear Spring model')
```

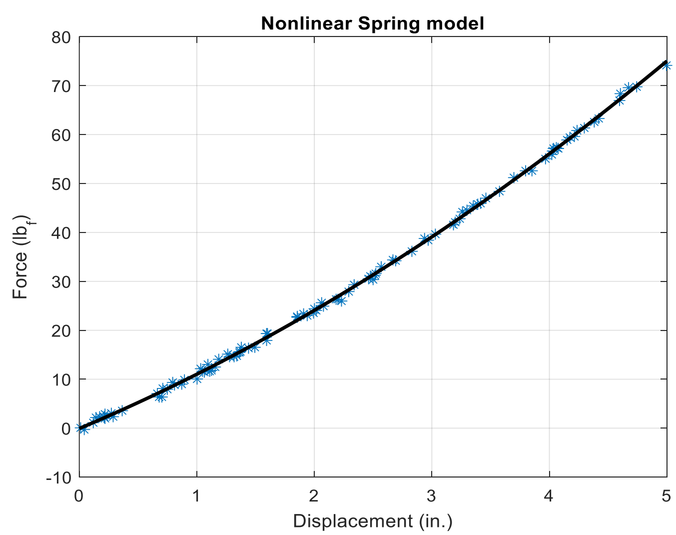


Figure 1.3 $\hat{F} = 0.98X^2 + 10.07X - 0.06$.

The model used to generate the data set was $F = X^2 + 10X + \varepsilon$. □

***Example* 2 A 2-D Prediction Model Example**

Consider the 2-D prediction model: $\begin{bmatrix} \hat{Y}_1 \\ \hat{Y}_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \Rightarrow \hat{\mathbf{Y}} = \mathbf{AX} + \mathbf{B}$.

We have

<u>Condition 1</u>: $$E(\hat{\mathbf{Y}}) = \mathbf{A}E(\mathbf{X}) + \mathbf{B} = E(\mathbf{Y}) \quad \Rightarrow \quad \mathbf{B} = \boldsymbol{\mu}_{\mathbf{Y}} - \mathbf{A}\boldsymbol{\mu}_{\mathbf{X}}. \tag{6}$$

<u>Condition 2</u>: $$Cov(\mathbf{Y} - \hat{\mathbf{Y}}, \mathbf{X}) = \mathbf{0}_{2\times 2}. \tag{7a}$$

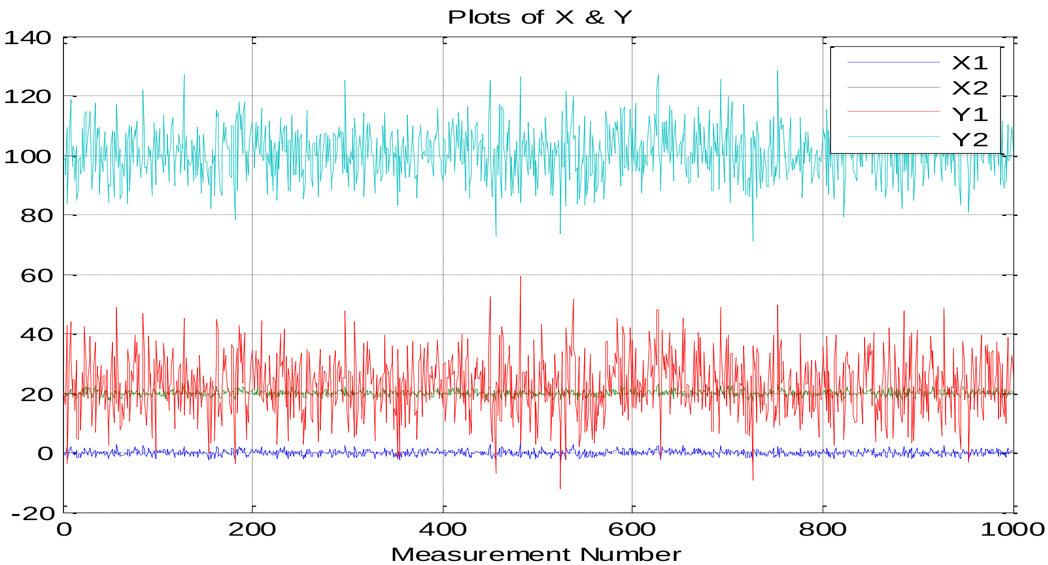
From (7a): $$Cov(\mathbf{Y}, \mathbf{X}) = Cov(\mathbf{AX} + \mathbf{B}, \mathbf{X}) = \mathbf{A}Cov(\mathbf{X}, \mathbf{X}) \quad \Rightarrow \quad \mathbf{A} = Cov(\mathbf{X}, \mathbf{X})^{-1} Cov(\mathbf{Y}, \mathbf{X}). \tag{7b}$$

The Matlab code is:

```
% PROGRAM NAME: TwoDmodeldemo.m
A = [ 10 ,1 ; 5, 5];
B = [3 ; 1];
n = 1000;
SigmaXX_true = [1 0 ; 0 1];
MuX_true = [0 ; 20];
X = mvnrnd(MuX_true , SigmaXX_true,n);
SigmaEE_true = eye(2);
MuE_true = [0 ; 0];
E = mvnrnd(MuE_true , SigmaEE_true,n);
Y = zeros(n,2);
for k = 1:n
    Y(k,:) = A*X(k,:)' + B + E(k,:)';
end
nvec = 1:n; nvec = nvec';
figure(1)
XY = [X , Y];%(X,Y) nx4 data array
plot(nvec,XY)
legend('X1','X2','Y1','Y2')
xlabel('Measurement Number')
title('Plots of X & Y')
grid
pause
%=================================
CXY = cov(XY)
pause
SigmaYX = CXY(3:4,1:2)
SigmaYX_true = A*SigmaXX
SigmaXX = CXY(1:2,1:2)
SigmaXX_true
pause
Ahat = SigmaYX*SigmaXX^-1
A
pause
MuXY = mean(XY);
MuX = MuXY(1:2)';
MuY = MuXY(3:4)'
MuY_true = A*MuX_true + B
pause
Bhat = MuY - Ahat*MuX
B
%==================================
% Compute Yhat:
Yhat = zeros(n,2);
for k = 1:n
    Yhat(k,:) = Ahat*X(k,:)' + Bhat;
end
figure(2)
plot(nvec,Y,'k',nvec,Yhat,'r')
title('Prediction Performance')
```

```
grid
%===================================
% Estimate SigmaEE:
Ehat = Y - Yhat;
SigmaEE = cov(Ehat)
SigmaEE_true
```

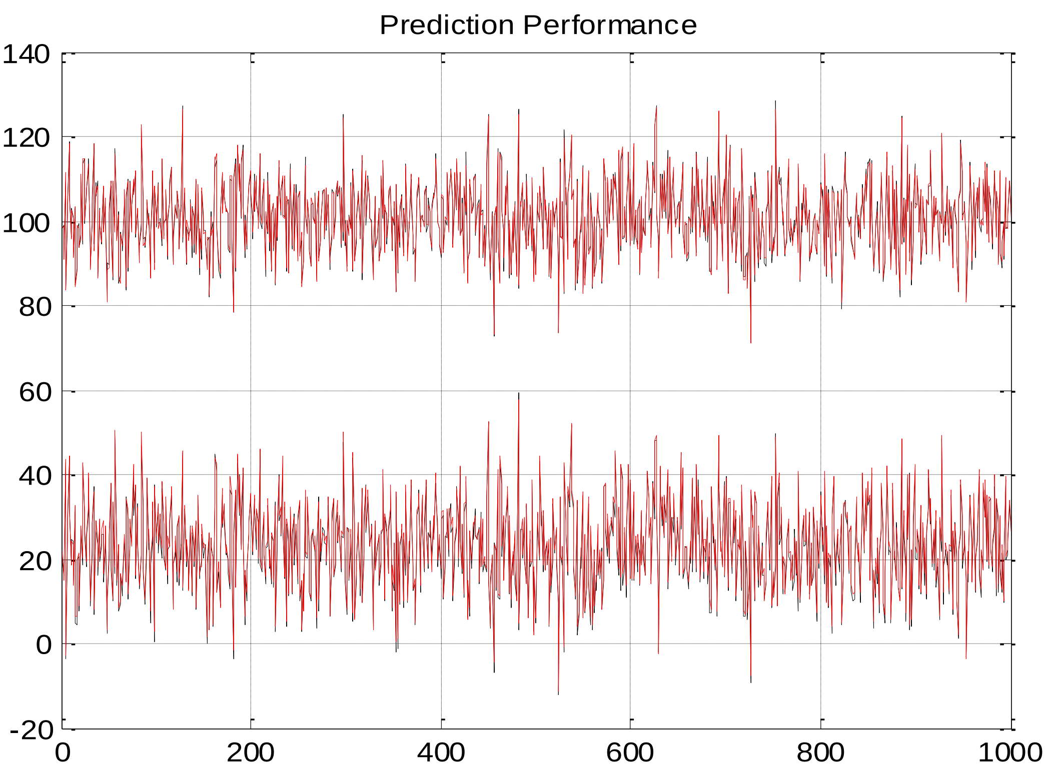
The data are plotted below.



Ahat = [10.08 0.96 ; 4.98 4.98] & A =[10 1 ; 5 5]

Bhat = [3.7329 1.4105] & B = [3 1]

The predictions are overlaid against the data in the plot below.



□