

# Linear Regression Models

## P8111

Lecture 07

Jeff Goldsmith  
February 11, 2016



THE DEPARTMENT OF  
**BIostatISTICS**



Columbia University  
**MAILMAN SCHOOL  
OF PUBLIC HEALTH**

# Today's lecture

- Multiple Linear Regression
  - "Non-linear" models
  - MLR Estimation
  - LSE Properties

# Non-linear relationships

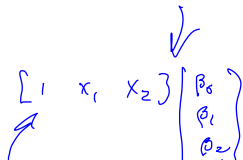
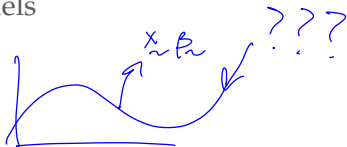
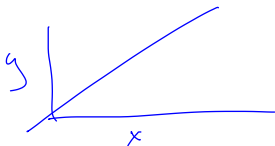
What do we mean by “linear models”?

- Linearity in the coefficients
- Conditional expectations are a linear combination of scalar values and regression coefficients

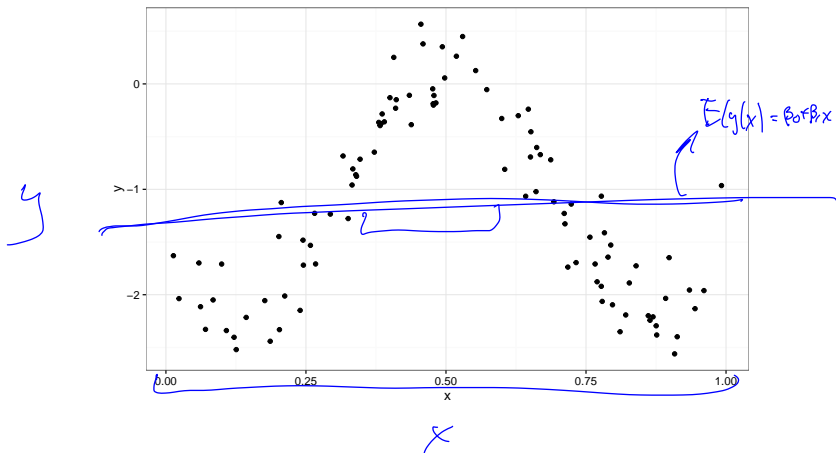
▶  $E(y|x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$  is linear;

$E(y|x) = \beta_0 + x_1^{\beta_1} + \log(\beta_2)x_2$  is not

- A non-linear relationship between  $y$  and  $x$  can still be addressed using linear models



# Non-linear relationships



# Non-linear relationships

Some ways to address this sort of thing

- Polynomials ✓
- Piece-wise linear models ✓
- Splines ✓

# Polynomial models

$$y_i = x_i \beta + \epsilon_i$$

- Model of the form

$$\underline{y_i} = \underline{\beta_0} + \underline{\beta_1 x_i} + \underline{\beta_2 x_i^2} + \dots + \underline{\beta_p x_i^p} + \epsilon_i; \epsilon_i \stackrel{iid}{\sim} (0, \sigma^2)$$

- $p$  is the polynomial order

- More polynomial terms can lead to a better approximation of  $E(y|x)$ , but also higher variability in the fit
- Conversely, smaller  $p$  can lead to inability to capture  $E(y|x)$ , but is often more stable
- Quadratic fits are pretty okay. I don't trust cubic and beyond.

# Polynomial models

$$y_i = \beta_0 + \beta_1 k_i + \beta_2 k_i^2 + \beta_3 k_i^3 + \beta_4 k_i^4 + \epsilon_i$$

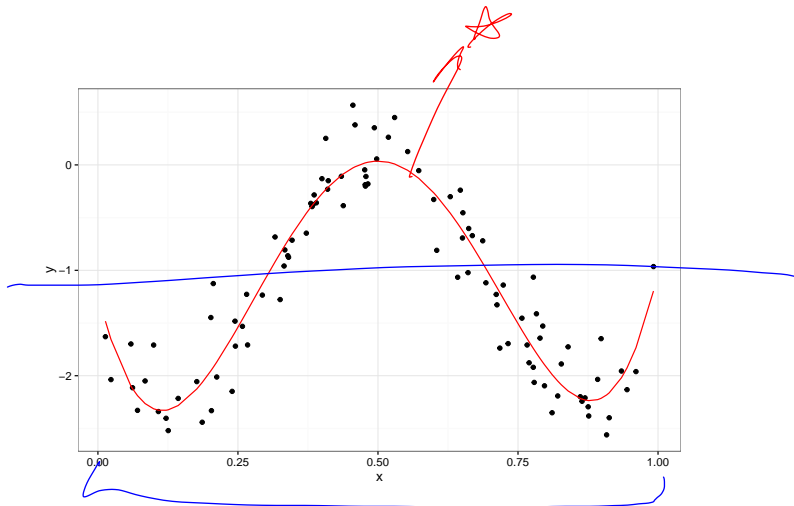
$$\hookrightarrow y = X\beta + e$$

$$\begin{bmatrix} x_1 & x_1^2 & x_1^3 & x_1^4 \\ x_2 & x_2^2 & x_2^3 & x_2^4 \\ x_3 & x_3^2 & x_3^3 & x_3^4 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & x_n^2 & x_n^3 & x_n^4 \end{bmatrix}$$

```
> data.nonlin = mutate(data.nonlin,  
+   x.pow2 = x^2, x.pow3 = x^3, x.pow4 = x^4)  
>  
> quartfit = lm(y ~ x + x.pow2 + x.pow3 + x.pow4, data = data.nonlin)  
> tidy(quartfit)
```

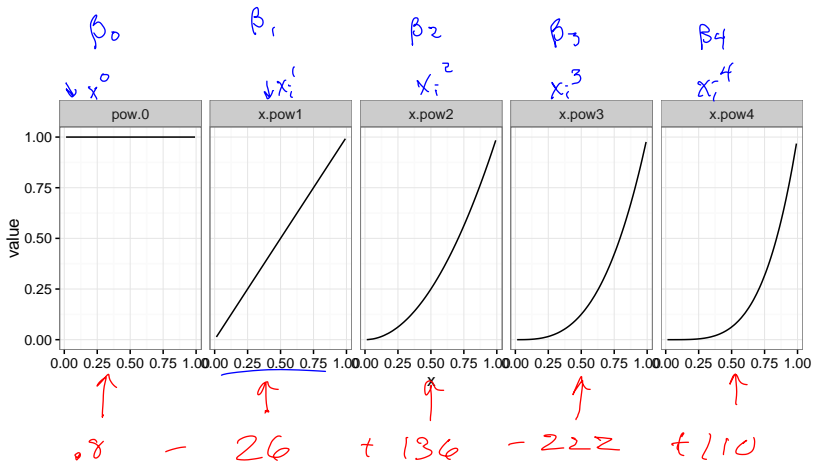
	term	estimate	std.error	statistic	p.value
1	(Intercept)	-0.8288032	0.2743086	-3.021426	3.232778e-03
2	x	-24.8618532	3.2948666	-7.545633	2.696508e-11
3	x.pow2	136.8666639	11.9951053	11.410209	1.671996e-19
4	x.pow3	-222.8346094	16.6191581	-13.408297	1.234857e-23
5	x.pow4	110.5772065	7.7358391	14.294145	2.079862e-25

# Polynomial models





# Polynomial models

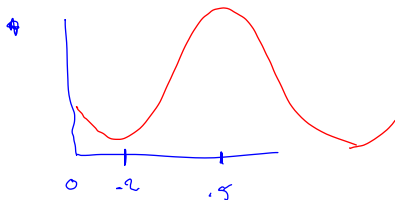


# Polynomial models

- Interpretation of  $\beta_1$ :

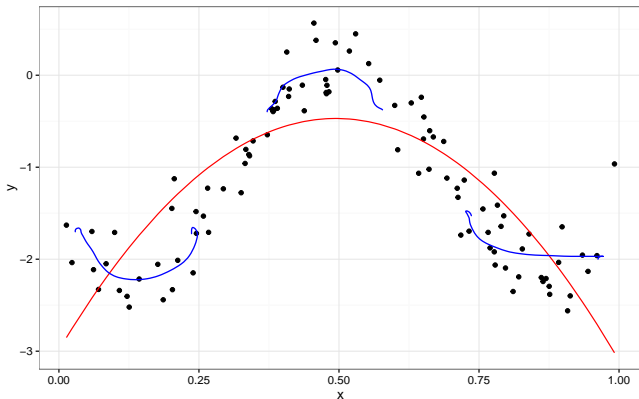
"keeping everything else fixed" ??

---



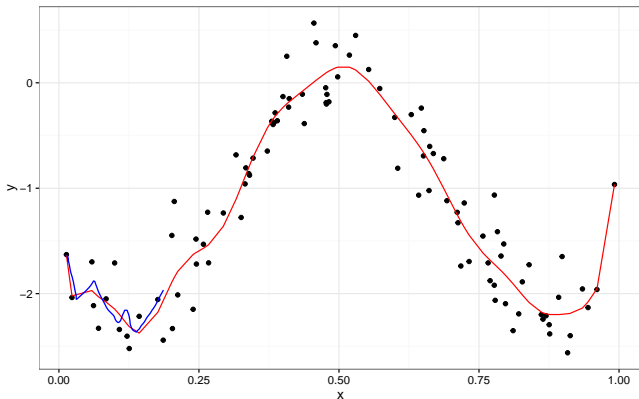
# Not enough polynomial terms

$$\beta_0 + \beta_1 x_i + \beta_2 x_i^2$$

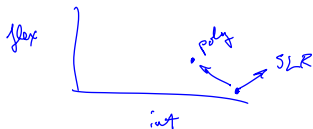


# Too many polynomial terms

$$\beta_0 + \beta_1 x_i + \dots + \beta_{20} x_i^{20}$$



# Final thoughts on polynomial models

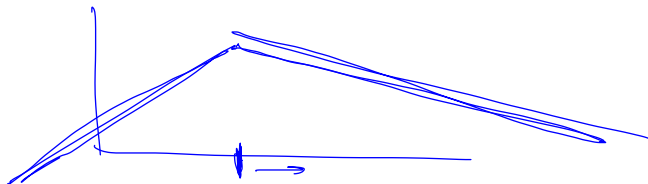


- Always include lower-order terms with higher-order terms
- You have to choose  $p$ , which isn't always easy
- Interpretation can be hard
- Raising continuous predictors for powers can lead to very large entries in your design matrix
- $x$  is almost always correlated with  $x^2$ .

# Piecewise linear models

A piecewise linear model (also called a change point model or broken stick model) contains a few linear components

- Outcome is linear over full domain, but with a different slope at different points
- Points where relationship changes are referred to as “change points” or “knots”
- Often there's one (or a few) potential change points



# Piecewise linear models

Suppose we want to estimate  $E(y|x) = f(x)$  using a piecewise linear model.

- For one knot we can write this as

$$\underline{E(y|x)} = \underline{\beta_0} + \underline{\beta_1 x} + \underline{\beta_2(x - \kappa)} \oplus$$

where  $\kappa$  is the location of the change point

# Interpretation of regression coefficients



# Estimation

- Piecewise linear models are low-dimensional (no need for penalization)
- Parameters are estimated via OLS
- The design matrix is ...

# Multiple knots

Suppose we want to estimate  $E(y|x) = f(x)$  using a piecewise linear model.

- For multiple knots we can write this as

$$E(y|x) = \beta_0 + \beta_1 x + \sum_{k=1}^K \beta_{k+1} (x - \kappa_k)_+$$

where  $\{\kappa_k\}_{k=1}^K$  are the locations of the change points

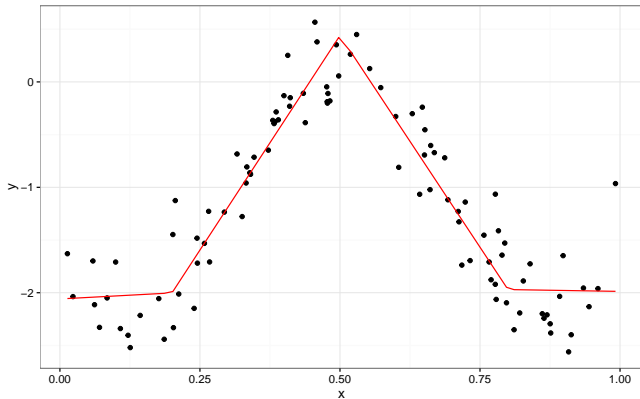
- Note that knot locations are defined before estimating regression coefficients
- Also, regression coefficients are interpreted conditional on the knots.

# Example

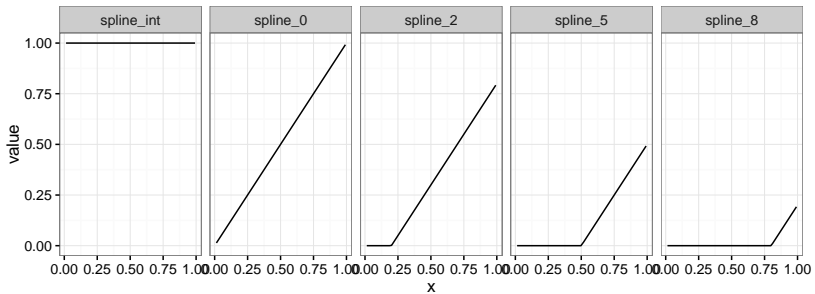
```
> data.nonlin = mutate(data.nonlin,  
+                       spline_2 = (x - .2) * (x >= .2),  
+                       spline_5 = (x - .5) * (x >= .5),  
+                       spline_8 = (x - .8) * (x >= .8))  
>  
> piecewise.fit = lm(y ~ x + spline_2 + spline_5 + spline_8, data = data.nonlin)  
> tidy(piecewise.fit)
```

	term	estimate	std.error	statistic	p.value
1	(Intercept)	-2.266553	0.2098863	-10.798953	3.280465e-18
2	x	1.655019	1.3456918	1.229865	2.217847e-01
3	spline_2	6.071173	1.6974614	3.576619	5.497482e-04
4	spline_5	-15.917475	0.8574252	-18.564273	2.283994e-33
5	spline_8	10.891211	1.1754422	9.265629	6.136562e-15

# Example



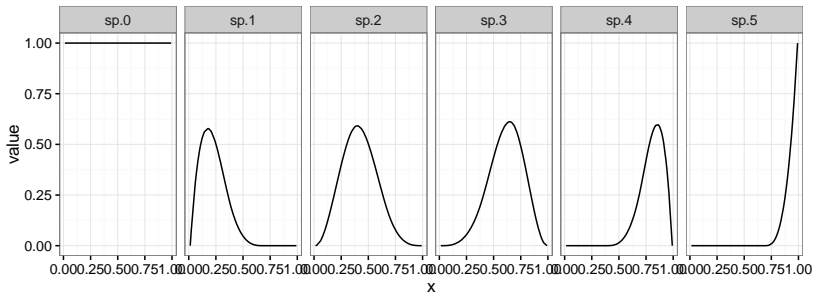
# Example



# Final thoughts on piecewise linear models

- Just like you can have too many polynomial terms, you can have too many knots
- You also have to choose where the knots go
- Interpretation is more straightforward than for polynomial models
- Can also have piecewise quadratic, piecewise cubic ...

# Spline models



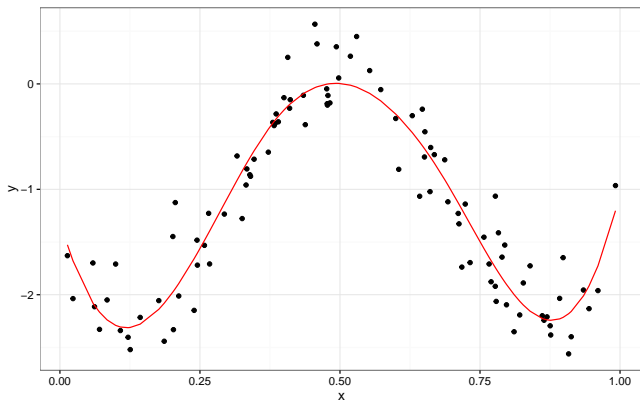
# Spline models

```
> data.nonlin = data.nonlin %>% bind_cols(., data.frame(ns(.[[ 'x' ]], df = 5))) %>%  
+   rename(sp.1 = X1, sp.2 = X2, sp.3 = X3, sp.4 = X4, sp.5 = X5)  
>  
> bspline.fit = lm(y ~ sp.1 + sp.2 + sp.3 + sp.4 + sp.5, data = data.nonlin)  
> tidy(bspline.fit)
```

	term	estimate	std.error	statistic	p.value
1	(Intercept)	-1.9529246	0.1420332	-13.749775	3.152280e-24
2	sp.1	2.5173253	0.1629991	15.443796	1.573047e-27
3	sp.2	1.9125629	0.2212551	8.644151	1.398240e-13
4	sp.3	-0.3654431	0.1575141	-2.320066	2.250136e-02
5	sp.4	-0.4146350	0.3533596	-1.173408	2.435969e-01
6	sp.5	0.4320127	0.1742734	2.478937	1.495979e-02



# Spline models



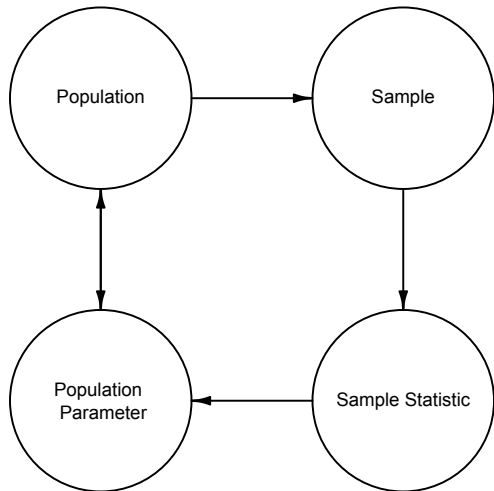
# Final thoughts on spline models

- Splines are constructed as numerically-stable versions of piecewise polynomials
- Cubic B-splines are popular (default of `splines::bs()`)
- Still have to choose knot location and number of knots
- Interpretation is roughly equivalent to that of polynomials

# Bringing it all together

- MLR covers a lot of stuff
- Models can be easy or very complex
- All depends on your design matrix ...

# Circle of Life



# Multiple linear regression

- Let

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1p} \\ \vdots & \vdots & & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_p \end{bmatrix}, \quad \boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

- Then we can write the model in a more compact form:

$$\mathbf{y}_{n \times 1} = \mathbf{X}_{n \times (p+1)} \boldsymbol{\beta}_{(p+1) \times 1} + \boldsymbol{\epsilon}_{n \times 1}$$

- $\mathbf{X}$  is called the *design matrix*

# Matrix notation

$$y = X\beta + \epsilon$$

- $\epsilon$  is a random vector rather than a random variable
- $E(\epsilon) = 0$  and  $Var(\epsilon) = \sigma^2 I$
- Note that *Var* is potentially confusing; in the present context it means the “variance-covariance matrix”

# Mean, variance and covariance of a random vector

- Let  $\mathbf{y}^T = [y_1, \dots, y_n]$  be an  $n$ -component random vector. Then its mean and variance are defined as

$$E(\mathbf{y})^T = [E(y_1), \dots, E(y_n)]$$

$$\text{Var}(\mathbf{y}) = E[(\mathbf{y} - E\mathbf{y})(\mathbf{y} - E\mathbf{y})^T] = E(\mathbf{y}\mathbf{y}^T) - (E\mathbf{y})(E\mathbf{y})^T$$

- Let  $\mathbf{y}$  and  $\mathbf{z}$  be an  $n$ -component and an  $m$ -component random vector respectively. Then their covariance is an  $n \times m$  matrix defined by

$$\text{Cov}(\mathbf{y}, \mathbf{z}) = E[(\mathbf{y} - E\mathbf{y})(\mathbf{z} - E\mathbf{z})^T]$$

# Basics on random vectors

Let  $A$  be a  $t \times n$  non-random matrix and  $B$  be a  $p \times m$  non-random matrix. Then

$$E(A\mathbf{y}) = AE\mathbf{y}$$

$$\text{Var}(A\mathbf{y}) = A\text{Var}(\mathbf{y})A^T$$

$$\text{Cov}(A\mathbf{y}, B\mathbf{z}) = A\text{Cov}(\mathbf{y}, \mathbf{z})B^T$$



# Vector differentiation

- For two vectors  $\mathbf{a}$  and  $\mathbf{b}$  and a matrix  $\mathbf{C}$ , the following rules hold:
  - $\frac{d}{d\mathbf{a}}(\mathbf{a}^T \mathbf{b}) = \mathbf{b}$
  - $\frac{d}{d\mathbf{a}}(\mathbf{a}^T \mathbf{C} \mathbf{a}) = (\mathbf{C} + \mathbf{C}^T) \mathbf{a}$
  - In the special case when the matrix  $\mathbf{C}$  is symmetric (i.e.  $\mathbf{C} = \mathbf{C}^T$ ), we have  $\frac{d}{d\mathbf{a}}(\mathbf{a}^T \mathbf{C} \mathbf{a}) = 2\mathbf{C} \mathbf{a}$

# Least squares

As in simple linear regression, we want to find the  $\beta$  that minimizes the residual sum of squares.

$$RSS(\beta) = \sum_i \epsilon_i^2 =$$

# Least squares

# Unbiasedness of LSEs

$$E(\hat{\beta}) =$$

## Variance of LSEs

$$\text{Var}(\hat{\beta}) =$$

$$\text{Var}(c\hat{\beta}) =$$

# Sampling distribution of $\hat{\beta}$

If our usual assumptions are satisfied and  $\epsilon \sim N [0, \sigma^2 I]$  then

$$\hat{\beta} \sim N \left[ \beta, \sigma^2 (\mathbf{X}^T \mathbf{X})^{-1} \right].$$

- This will be used later for inference.
- Even without Normal errors, asymptotic Normality of LSEs is possible under reasonable assumptions.

# Definitions

- *Fitted values:*  $\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{H}\mathbf{y}$
- *Residuals / estimated errors:*  $\hat{\boldsymbol{\epsilon}} = \mathbf{y} - \hat{\mathbf{y}}$
- *Residual sum of squares:*  $\sum_{i=1}^n \hat{\epsilon}_i^2 = \hat{\boldsymbol{\epsilon}}^T \hat{\boldsymbol{\epsilon}}$
- *Residual variance:*  $\hat{\sigma}^2 = \frac{RSS}{n-p-1}$
- *Degrees of freedom:*  $n - p - 1$

## $R^2$ and sums of squares

- Regression sum of squares  $SS_{reg} = \sum(\hat{y}_i - \bar{y})^2$
- Residual sum of squares  $SS_{res} = \sum(y_i - \hat{y}_i)^2$
- Total sum of squares  $SS_{tot} = \sum(y_i - \bar{y})^2$
- Coefficient of determination

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2}$$



# Hat matrix

Some properties of the hat matrix:

- It is a projection matrix:  $\mathbf{H}\mathbf{H} = \mathbf{H}$
- It is symmetric:  $\mathbf{H}^T = \mathbf{H}$
- The residuals are  $\hat{\mathbf{e}} = (\mathbf{I} - \mathbf{H})\mathbf{y}$
- The inner product of  $(\mathbf{I} - \mathbf{H})\mathbf{y}$  and  $\mathbf{H}\mathbf{y}$  is zero (predicted values and residuals are uncorrelated).

# Projection space interpretation

The hat matrix projects  $\mathbf{y}$  onto the column space of  $\mathbf{X}$ .  
Alternatively, minimizing the  $RSS(\beta)$  is equivalent to minimizing the Euclidean distance between  $\mathbf{y}$  and the column space of  $\mathbf{X}$ .

# Today's big ideas

- Non-linear models; least squares estimates and properties; definitions, hat matrix and vector space interpretation

- 
- Suggested reading: Faraway Ch 2.2 - 2.7; ISLR 3.2