Convenient Matrix Notation

Define:

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_k \end{bmatrix}; \quad \mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ 1 & x_{31} & x_{32} & \cdots & x_{3k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix}; \quad \mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}; \quad \varepsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

Model becomes:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \varepsilon$$

LS solution becomes:

$$\left[\mathbf{X}^T\mathbf{X}\right]\hat{\beta} = \mathbf{X}^T\mathbf{Y}$$

If $\mathbf{X}^T\mathbf{X}$ invertible:

$$\hat{\beta} = \left[\mathbf{X}^T\mathbf{X}\right]^{-1}\mathbf{X}^T\mathbf{Y}$$

- **Tip:** Always pay attention to whether the quantities are scalars, vectors, or matrices, and their dimensions

Assessing the Fit

- As in simple regression, calculate:

  fitted values: $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \hat{\beta}_2 x_{i2} + \boxed{?} + \hat{\beta}_k x_{ik} : \quad i = 1,2,\ldots,n$

  residuals: $e_i = y_i - \hat{y}_i : \quad i = 1,2,\ldots,n$

  error sum of squares: $SSE = \sum_{i=1}^{n} e_i^2$

  total sum of squares: $SST = \sum_{i=1}^{n} (y_i - \bar{y})^2$

  regression sum of squares: $SSR = \sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2$

- Still have same total sum of squares decomposition:

$$SST = SSR + SSE$$

r^2 for Multiple Regression (beware though)

- We can still look at $r^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$

- In multiple regression, r^2 is called **coefficient of multiple determination**. It still represents the proportion of variability in y that is accounted for by its linear dependence on the set of predictors.
- Mathematically, r^2 is equivalent to the square of the correlation coefficient between y_i and $\hat{y}_i$

- **Beware**: r^2 is artificially high when $n \not\gg k$ because of overfitting – use something called "adjusted r^2" instead (coming up soon)

Illustration of Overfitting with Simulated Data

- The following code generates an array of completely random data with n rows and k predictor variables and fits a regression model
- What will happen if we use k = 50 and n = 40? Why?
- With n = 40, what is the largest k for which we can still fit the model and estimate all coefficients? What will r^2 be in this case?
- What will happen if we use k = 30 and n = 40?

```
#####R code for overfitting randomly generated data##################
k=30;n=40
x<-matrix(rnorm(k*n,0,1),n,k)
y<-rnorm(n,0,1)
overfit<-data.frame(y,x)
lm2<-lm(y~.,data=overfit)
summary(lm2)
yhat<-fitted(lm2)
plot(yhat,y)  #plot of y vs. fitted values
#####
rm(x,y,overfit,k,n)
```

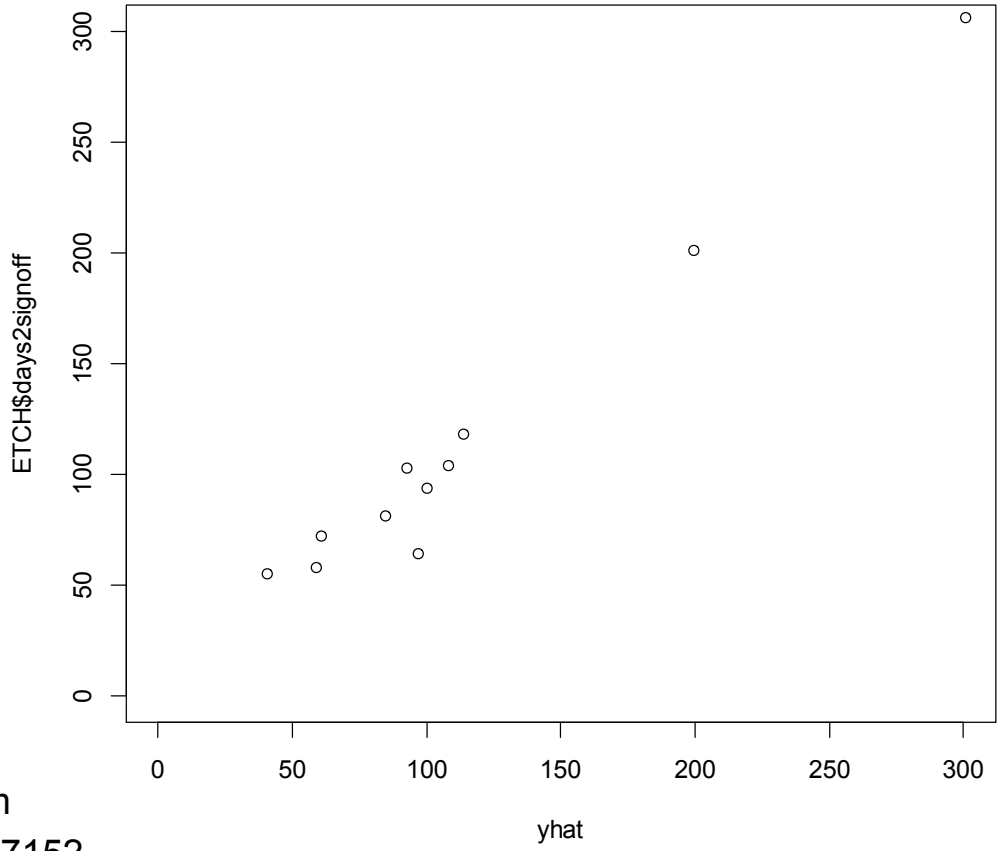
A Real Overfitting Example (sil_etch.txt)

- A manufacturer of semiconductor etching machines wants to predict the number of days until the customer signs off on a received machine and pays the manufacturer (after shipping to customer, set up, trouble-shooting, fine tuning, etc, so that the machine is confirmed to work properly). This became extremely important following the Sarbanes-Oxley Act that tightened the rules on corporate accounting following the scandals of the late 1990's
- The idea is to predict days2signoff before the machine is shipped to the customer, based on quality-related predictor variables that are recorded during manufacturing
- sil_etch.txt contains the days2signoff (the response) and nine other predictors for a set of 11 machines that were manufactured, shipped and eventually signed-off (they produce many machines, but not many of each type, and they did not want to mix machines when shipping).
- Let's fit a multiple regression model regressing days2signoff onto all nine predictors and see how well the model predicts

Fit a Multiple Regression to the ETCH Data

```
##########R code for fitting a multiple regression model to the ETCH data######
ETCH<-read.table("sil_etch.txt", header=TRUE, sep="\t")
ETCH
lm1<-lm(days2signoff~.,data=ETCH)
summary(lm1)
yhat <- predict(lm1)
plot(yhat,ETCH$days2signoff, ylim=c(0,300), xlim=c(0,300))
data.frame(ETCH,round(yhat))
```

```
> summary(lm1)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 9993.23538 3950.70892   2.529    0.240
MNC          -32.23923   13.10992  -2.459    0.246
ISDR           0.05951   21.12661   0.003    0.998
DMR          -15.76123    4.70646  -3.349    0.185
PDSrev        16.03662   15.27724   1.050    0.485
NSR          121.27142   46.62543   2.601    0.234
UPSF         -29.01261   12.14763  -2.388    0.252
ILTR          -7.91838   10.52495  -0.752    0.589
BayDay       -49.91432   23.38041  -2.135    0.279
Test        -900.59213  362.52289  -2.484    0.244

Residual standard error: 40.3 on 1 degrees of freedom
Multiple R-squared: 0.9715,     Adjusted R-squared: 0.7152
F-statistic: 3.791 on 9 and 1 DF,  p-value: 0.3801
```



Discussion Points and Questions

- How good does the fit appear to be for the ETCH data?
- Does it look like the fitted model for days2signoff has good predictive power?
- If you were the manufacturer, would you be comfortable using the model to predict days2signoff for machines you are about to ship?

Definition of r^2_{adj}

- Recall: $r^2 = 1 - \frac{SSE}{SST} = 1 - \frac{\left(\frac{SSE}{n-1}\right)}{\left(\frac{SST}{n-1}\right)}$
- Define the "mean squares" corresponding to the "sum of squares":

$$MSE = \frac{SSE}{n-(k+1)} = \text{unbiased estimate of } \sigma_\varepsilon^2$$

$$MST = \frac{SST}{n-1} = \text{unbiased estimate of } \sigma_Y^2$$

- For multiple regression, instead of r^2 you should look at "adjusted r^2":

$$r^2_{adj} = 1 - \frac{\hat{\sigma}_\varepsilon^2}{\hat{\sigma}_Y^2} = 1 - \frac{MSE}{MST} = 1 - \frac{SSE}{SST}\left[\frac{n-1}{n-k-1}\right]$$

Discussion Points and Questions

- In multiple regression, r^2_{adj} is interpreted as a better estimate (than r^2) of the percentage of variability in the response that is attributed to its linear dependence on the predictors
- But with severe overfitting, r^2_{adj} can still be misleading if the error d.f. is very small
- What are r^2_{adj} and r^2 for the GAS data? For the ETCH data? For the simulated random data with k = 30 and n = 40?
- Does r^2_{adj} for the ETCH data seem reasonable?

Statistical Inference on the Coefficients

- A regression fit can seem **practically significant** (high r^2) without being **statistically significant**, and vice-versa.
- Three common tests of whether individual parameters or groups of parameters differ from zero are:
  - F-test for testing whether at least one of the k parameters differs from zero
  - t-tests and CIs for testing whether an individual parameter differs from zero (if so, the predictor has a statistically significant effect on the response)
  - Partial sum of squares F-test for testing whether at least one of a specified group of parameters differs from zero

Overall F-test on All k Coefficients

All of the statistical inference assumes a "true" model:

observations: $Y_i = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_k x_{ik} + \varepsilon_i : \quad i = 1, \ldots, n$

random errors: $\varepsilon_i \sim N(0,\sigma^2)$ and all i.i.d.

"true" parameters: $\beta_0, \beta_1, \ldots, \beta_k$

To test: $H_0: \beta_1 = \ldots = \beta_k = 0$

$H_1:$ at least one $\beta \neq 0$

Use test statistic $F = \dfrac{MSR}{MSE}$ where $MSR = \dfrac{SSR}{k}$

Null distribution: $F \sim F_{k,n-(k+1)}$

Reject H_0 if $F > f_{k,n-k-1,\alpha}$

F-test for the GAS data

```
##########R code for F-test with gas mileage data and r^2#####
GAS<-read.csv("gas_mileage.csv",header=TRUE)
n<-30
k<-11
lm1<-lm(Mpg~.,data=GAS)
summary(lm1)  #The F-test produced by the summary() command is the overall F-test
a <- anova(lm1); a  #This shows SSE, MSE, and other things
#The following does the same F-test manually
SSR <- sum(a[[2]][1:11])
SSE <- a[[2]][12]
MSR <- SSR/k
MSE <- SSE/(n-k-1)
F <- MSR/MSE
pf(F,k,n-k-1, lower.tail=FALSE) #P-value for F test
```

```
> summary(lm1)

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     17.339838  30.355375   0.571   0.5749
Displacement    -0.075588   0.056347  -1.341   0.1964
Hpower          -0.069163   0.087791  -0.788   0.4411
Torque           0.115117   0.088113   1.306   0.2078
Comp_ratio       1.494737   3.101464   0.482   0.6357
Rear_axle_ratio  5.843495   3.148438   1.856   0.0799 .
Carb_barrels     0.317583   1.288967   0.246   0.8082
No._speeds      -3.205390   3.109185  -1.031   0.3162
Length           0.180811   0.130301   1.388   0.1822
Width           -0.397945   0.323456  -1.230   0.2344
Weight          -0.005115   0.005896  -0.868   0.3971
Trans._type      0.638483   3.021680   0.211   0.8350
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.227 on 18 degrees of freedom
  (2 observations deleted due to missingness)
Multiple R-squared: 0.8355,     Adjusted R-squared: 0.7349
F-statistic:  8.31 on 11 and 18 DF,  p-value: 5.231e-05
```

Discussion Points and Questions

- Is the multiple regression fit to the GAS data statistically significant?
- In general, does strong statistical significance imply a strong predictability?

Practical Versus Statistical Significance

triscan_5dx.txt contains quite a few observations of two variables related to measurement of solder paste volume in printed circuit board assembly. The response is FiveDX, which are volume measurements for a set of solder bricks using a machine based on X-ray technology. The predictor variable is Triscan, which are volume measurements of the same set of solder bricks using a machine based on laser scanning. The Triscan measurements are known to be quite accurate, but these measurements can only be obtained prior to placing the chips on the board. The FiveDX measurements can be obtained even after the chips are placed, but their accuracy is in question. The goal is to assess the accuracy of the FiveDX measurements by comparing it to the Triscan measurements. What is the conclusion?

```
#######R code#####
X<-read.table("triscan_5dx.txt",header=TRUE,sep="\t")
X[1:20,]
anova(lm(FiveDX~Triscan,data=X))
##
plot(X$Triscan,X$FiveDX); rm(X)
```

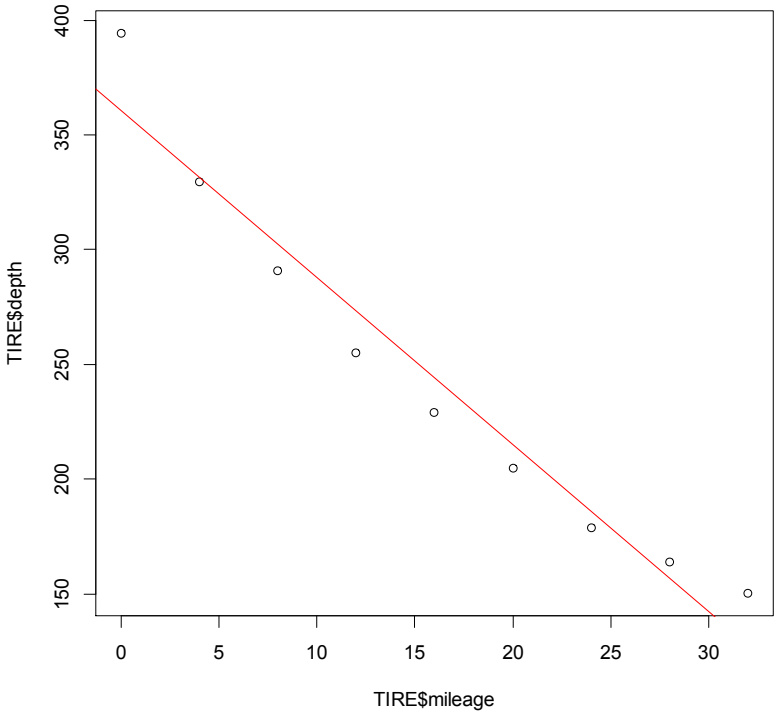
Discussion Points and Questions

- If the F-test rejects H_0, an appropriate next step might be to determine which of the predictor variables (e.g., all of them, just a few, etc) have significant effects on the response
- Why might it be of interest to determine which predictors have significant effects?
- How would you formalize this as an hypothesis test?
- We can sometimes (but there is a big pitfall, discussed later) use a *t*-test on individual coefficients to determine which β_j's $\neq 0$

t-tests for the Tire Wear Data

```
#######R code for t-tests and CIs on tire wear data ####
TIRE<-read.table("tire_wear.txt",header=TRUE,sep="\t")
TIRE
plot(TIRE$mileage, TIRE$depth)
abline(lm(depth~mileage, data=TIRE), col="red") #plot of simple lin. regression
lm1<-lm(depth~poly(mileage,2, raw=TRUE), data=TIRE)
summary(lm1)
confint(lm1,level=.95)

##The following fits the same quadratic model
lm1<-lm(depth ~ mileage + I(mileage^2), data=TIRE)

##can calculate t-percentile via
qt(.975, 6)
```

```
> summary(lm1)
Coefficients:
                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                     386.26485    4.79996   80.47 2.48e-10 ***
poly(mileage, 2, raw = TRUE)1 -12.77238    0.69948  -18.26 1.74e-06 ***
poly(mileage, 2, raw = TRUE)2   0.17162    0.02103    8.16 0.000182 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.906 on 6 degrees of freedom
Multiple R-squared: 0.9961,     Adjusted R-squared: 0.9948
F-statistic: 762.8 on 2 and 6 DF,  p-value: 6.011e-08

> confint(lm1,level=.95)
                                  2.5 %      97.5 %
(Intercept)                     374.5197613 398.0099357
poly(mileage, 2, raw = TRUE)1 -14.4839431 -11.0608134
poly(mileage, 2, raw = TRUE)2   0.1201549   0.2230796
```



Some Points and Pitfalls

- Usually begin with the overall F-test:
  - If H_0 not rejected, consider other predictors, nonlinear regression, or conclude there is no predictability and stop
  - If H_0 rejected, follow up by determining important predictors using t-tests on individual predictors (problematic with multicollinear predictors), partial F-tests on groups of predictors, or automated methods like stepwise or best subsets
- **Pitfall:** Beware interpreting individual t-tests when predictors are multicollinear, which is almost always. P-values will be misleadingly high. The reason is that the t-test of whether $\beta_j \neq 0$ is essentially testing whether including/excluding the individual predictor x_j in the model significantly changes the SSE. E.g., the t-test for β_1 compares the following two models:

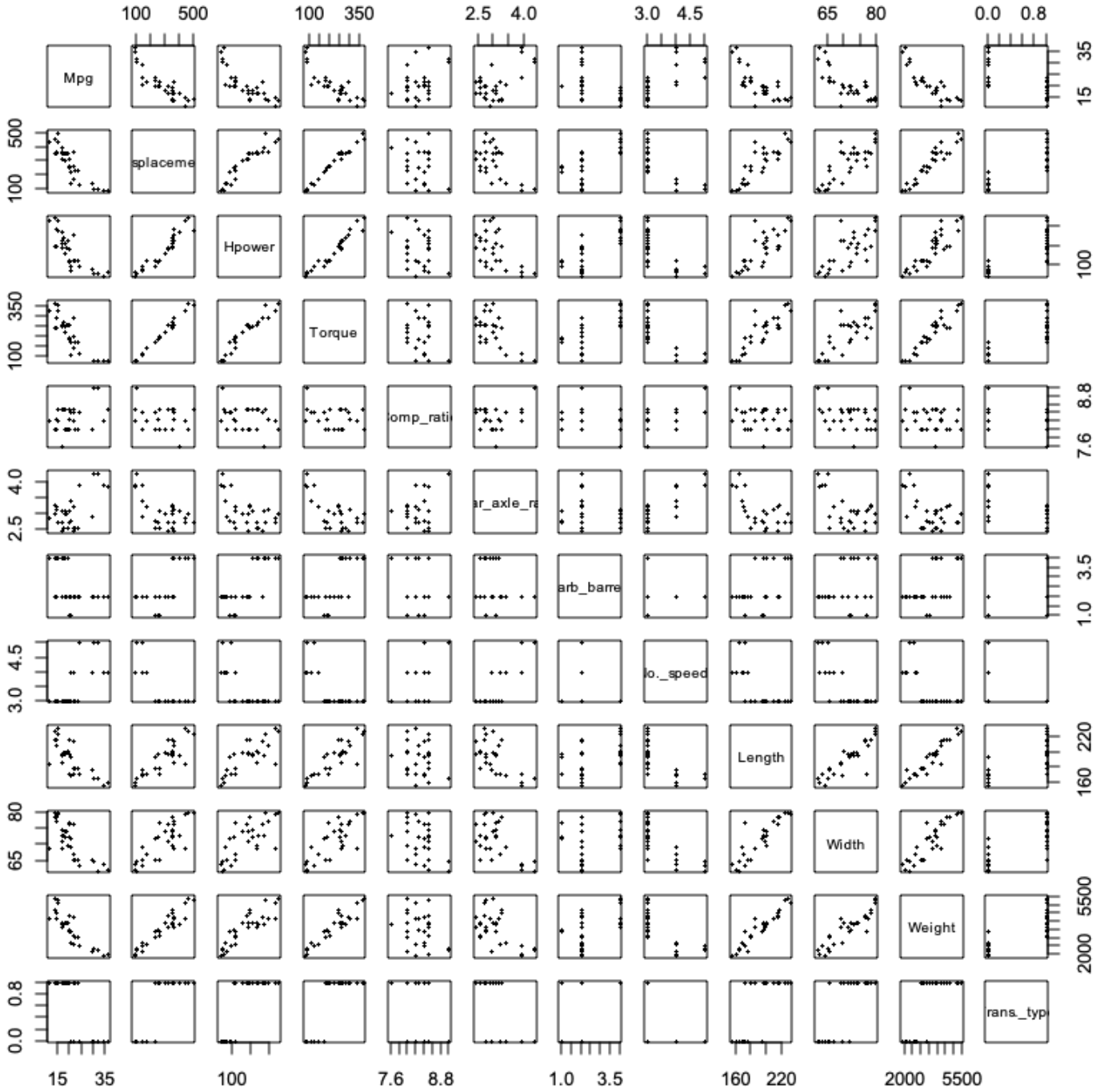
$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \varepsilon \quad (\text{with } x_1), \text{ vs}$$

$$Y = \beta_0 + \qquad \beta_2 x_2 + \ldots + \beta_k x_k + \varepsilon \quad (\text{without } x_1)$$

Individual t-tests for the GAS Data Illustrating the Pitfall

```
###########refit the gas mileage data regression with all predictors included######
GAS<-read.csv("gas_mileage.csv",header=TRUE)
pairs(GAS, cex = 0.5, pch = 16)  #matrix scatterplot
##fit a linear regression with all 11 predictors
lm1<-lm(Mpg~.,data=GAS)
summary(lm1)

##repeat with only Rear_axle_ratio and weight
lm1<-lm(Mpg~ Rear_axle_ratio + Weight,data=GAS)
summary(lm1)
```

```
> summary(lm1)
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     17.339838  30.355375   0.571   0.5749
Displacement    -0.075588   0.056347  -1.341   0.1964
Hpower          -0.069163   0.087791  -0.788   0.4411
Torque           0.115117   0.088113   1.306   0.2078
Comp_ratio       1.494737   3.101464   0.482   0.6357
Rear_axle_ratio  5.843495   3.148438   1.856   0.0799 .
Carb_barrels     0.317583   1.288967   0.246   0.8082
No._speeds      -3.205390   3.109185  -1.031   0.3162
Length           0.180811   0.130301   1.388   0.1822
Width           -0.397945   0.323456  -1.230   0.2344
Weight          -0.005115   0.005896  -0.868   0.3971
Trans._type      0.638483   3.021680   0.211   0.8350

Residual standard error: 3.227 on 18 degrees of freedom
  (2 observations deleted due to missingness)
Multiple R-squared: 0.8355,     Adjusted R-squared: 0.7349
F-statistic:  8.31 on 11 and 18 DF,  p-value: 5.231e-05
```


Mpg
splaceme
Hpower
Torque
omp_rati
r_axle_ra
arb_barre
lo._speed
Length
Width
Weight
rans._typ

the analogous results with only two predictors

```
> summary(lm1)

Call:
lm(formula = Mpg ~ Rear_axle_ratio + Weight, data = GAS)

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     31.7594958  5.8348313   5.443 7.41e-06 ***
Rear_axle_ratio  2.2141129  1.3146877   1.684    0.103
Weight          -0.0051025  0.0007106  -7.181 6.63e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.151 on 29 degrees of freedom
Multiple R-squared: 0.7674,     Adjusted R-squared: 0.7514
F-statistic: 47.84 on 2 and 29 DF,  p-value: 6.547e-10
```

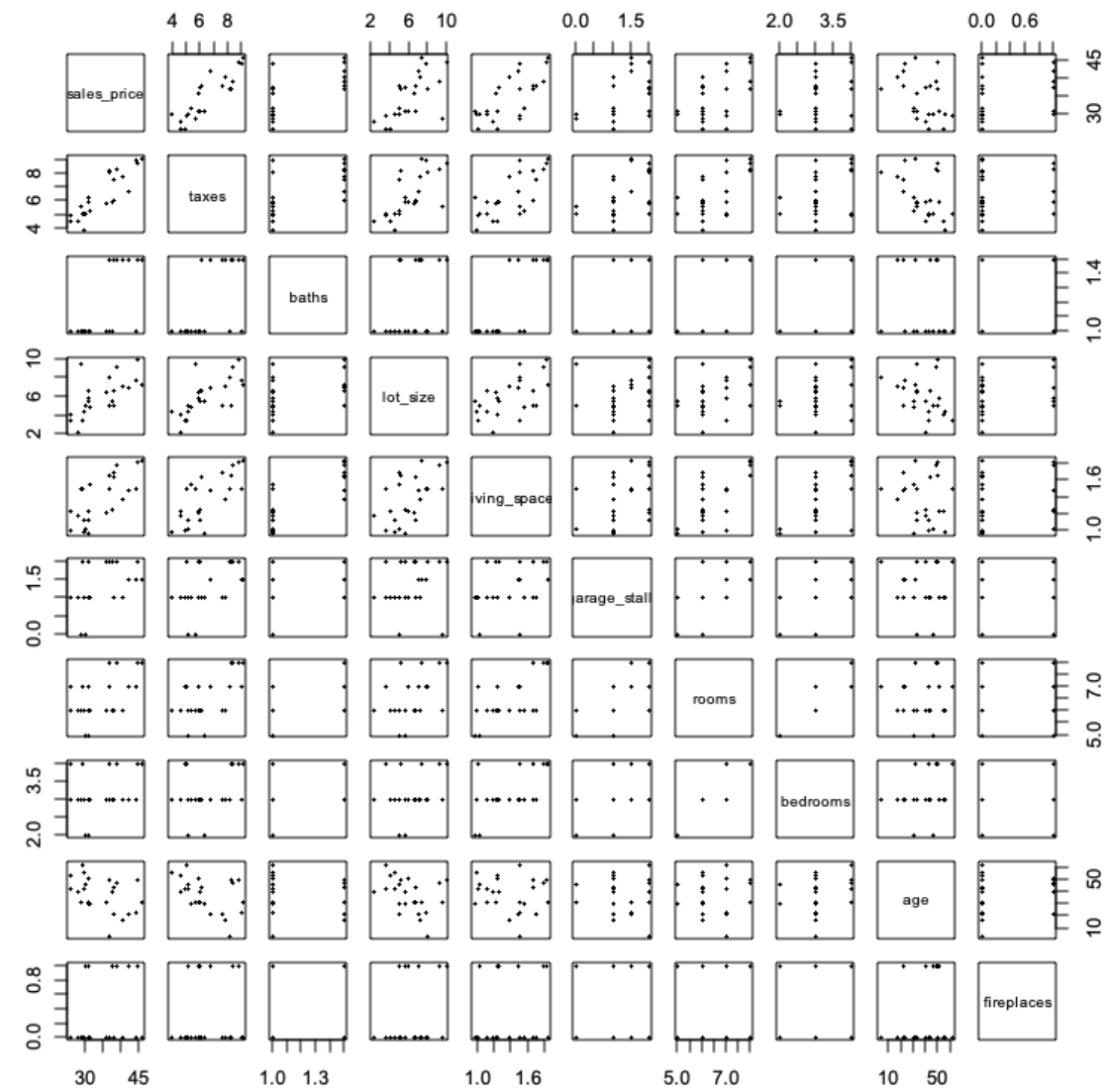
Discussion Points and Questions

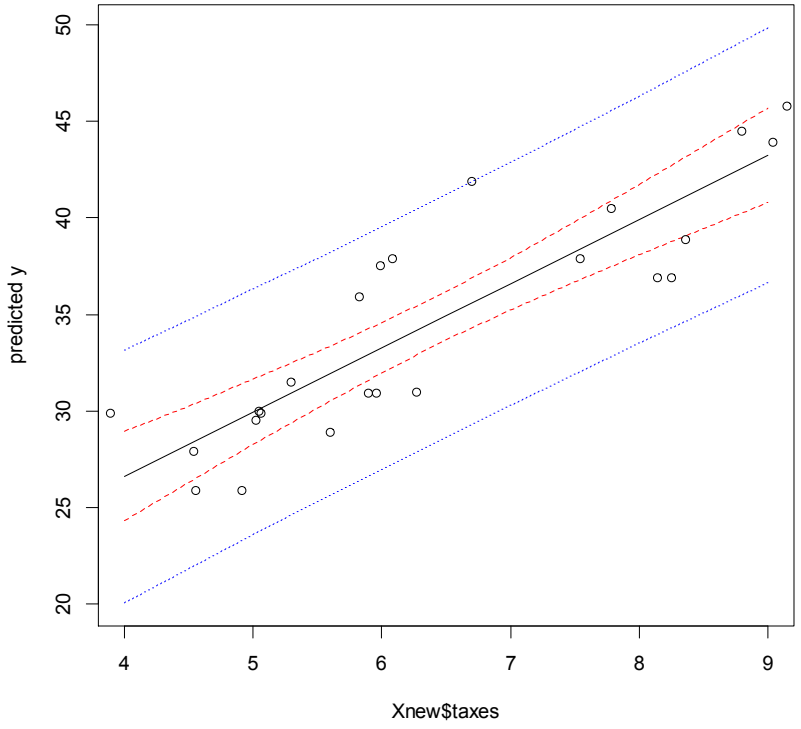
- Why are the coefficients not statistically significant when we include all 11 predictor variables?
- Why does Weight become much more significant when we fit the model with only Weight and Rear_axle_ratio included?

Example: Predicting Property Value – Illustration of PI on Y^* vs. CI on μ^*

- property_value.txt contains home sales prices and nine other characteristics (taxes, lot size, living space, age, etc) for a sample of 24 houses. The objective is to predict the sales price as a function of the other characteristics
- The following R code illustrates PIs and CIs for the simpler case of having only a single predictor taxes.

```
###R code for illustrating CIs and PIs with property value data with one predictor####
PROP<-read.table("property_value.txt",sep="\t",header=TRUE)
pairs(PROP, cex=0.5, pch=16)  #matrix scatterplot
PROP[1:10,]
Xnew <- data.frame(taxes = seq(4, 9, 0.5))
plim <- predict(lm(sales_price ~ taxes,data=PROP), newdata=Xnew, interval="prediction")
clim <- predict(lm(sales_price ~ taxes,data=PROP), newdata=Xnew,
      interval="confidence")
 matplot(Xnew$taxes,cbind(clim, plim[,-1]), lty=c(1,2,2,3,3),
      col=c("black","red","red","blue","blue"),type="l", ylab="predicted y")
points(PROP$taxes,PROP$sales_price)
```


sales_price
taxes
baths
lot_size
ving_space
arage_stall
rooms
bedrooms
age
fireplaces


predicted y
Xnew$taxes

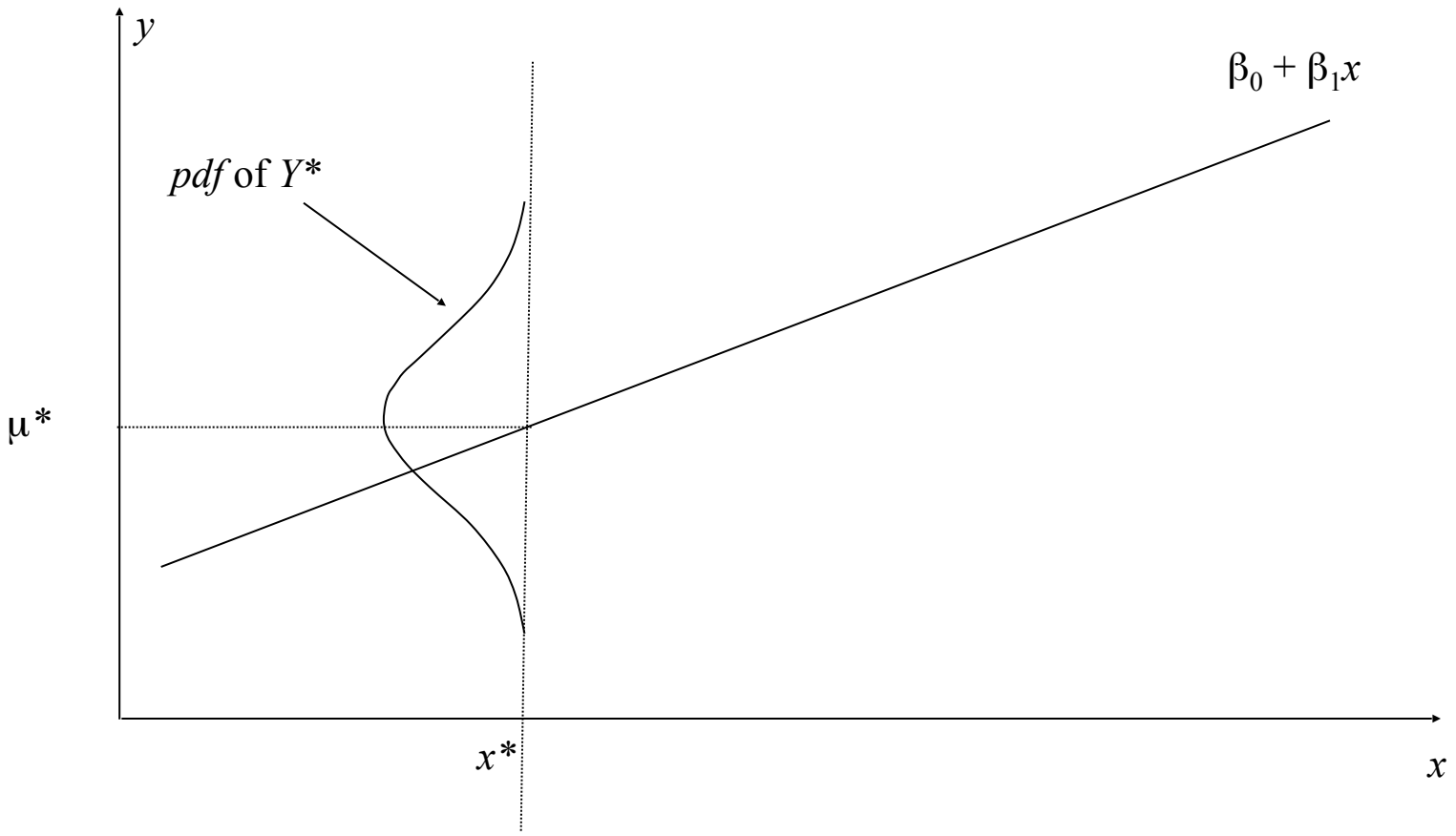
Discussion Points and Questions

- Which is the PI and which is the CI in the previous figure?
- What is the interpretation of the PI?
- What is the interpretation of the CI?
- If someone is putting their house up for sale and wants to know the high end of the range for which it might sell, would the response PI or CI be more relevant?
- What is the relationship between the CI on μ^* versus a CI on one of the coefficients?
- How are the response CI and PI calculated?

The Statistical View of Y^*

For fixed $\mathbf{x}^*$: $Y^* = \beta_0 + \beta_1 x_1^* + \ldots + \beta_k x_k^* + \varepsilon$

$= \mu^* + \varepsilon \quad \sim N(\mu^*, \sigma^2)$



Calculating PIs and CIs in R

```
##########R code for illustrating CIs and PIs with property value data##############
PROP<-read.table("property_value.txt",sep="\t",header=TRUE)
pairs(PROP[,1:3], cex=0.5, pch=16)  #matrix scatterplot
lm1<-lm(sales_price~.,data=PROP[,1:3])
summary(lm1)
Xnew<-data.frame(taxes=7,baths=1.5)
predict(lm1, newdata=Xnew, se.fit = T, level=0.95, interval = "confidence")
predict(lm1, newdata=Xnew, se.fit = T, level=0.95, interval = "prediction")

###manual calculations of some of the same thing###
s<-sqrt(anova(lm1)[[2]][3]/21)  #this is s, the sqrt of the MSE
X<-as.matrix(cbind(1,PROP[,2:3]))
x<-matrix(c(1,7,1.5),3,1)
V<-solve(t(X)%*%X)
SE<-s*sqrt(t(x)%*%V%*%x)  #this is SE of mu*
```