Unbalanced ANOVA

The previous examples we looked at had balanced designs–equal numbers of observations for each combination of factors. It is more typical to have unbalanced ANOVA, where there are different numbers of observations in the different factors. This could be because data are observational, and the numbers of observations were not in the researcher's control, or because of missing data (for example if a battery had been defective for reasons unrelated to the experiment, so had to be dropped from the study).

Unbalanced ANOVA

The experiment consists of measuring insulin levels in rats a certain length of time after a fixed dose of insulin was injected into their jugular or portal veins. This is a two-factor study with two vein types (jugular, portal) and three time levels (0, 30, and 60 minutes). An unusual feature of this experiment is that the rats used in the six vein and time combinations are distinct (so we don't use the rats as blocking variables).

We'll use a two-factor interaction model. (Again, we'll treat time as a factor although we could treat it as quantitative.) The sample sizes are between 3 and 12. The data is already in the long format, so doesn't need to be reshaped.

Unbalanced ANOVA

```
> rat <- read.table("rats.txt",header=T)
> rat$time <- factor(rat$time)
> rat
   vein time insulin
1     j    0      18
2     j    0      36
3     j    0      12
4     j    0      24
5     j    0      43
6     j   30      61
7     j   30     116
8     j   30      63
```

Unbalanced ANOVA

The sample sizes can found quickly using the `table()` command. The `dim()` command gives the dimensions of the data frame (number of rows and columns). The number of rows is the total sample size.

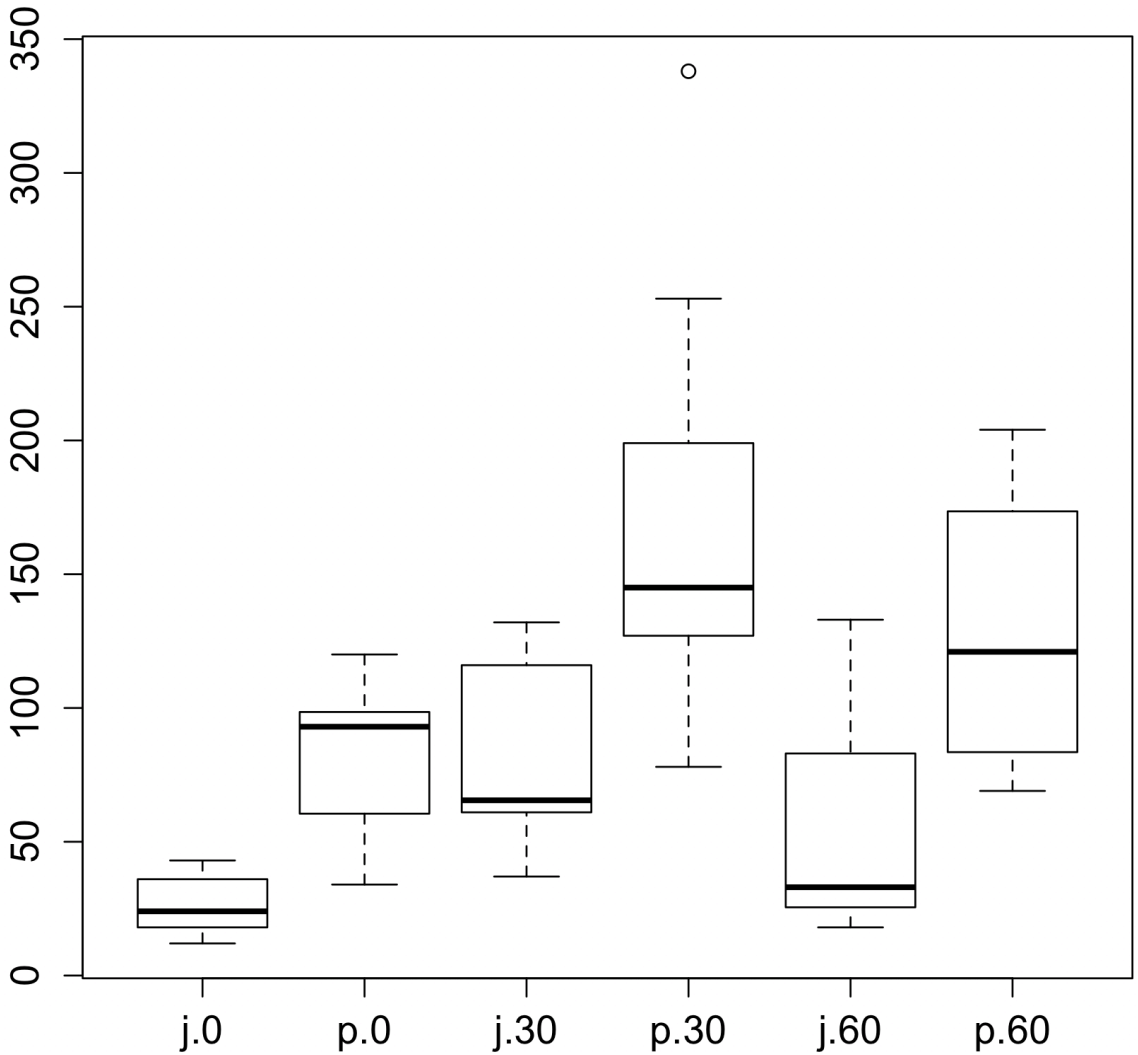
```
> table(vein,time)
     time
vein  0 30 60
   j  5  6  3
   p 12 10 12
> dim(rat)
[1] 48  3
```

Unbalanced ANOVA

Based on the boxplot, the variability in insulin appears to increase with the amount of time, so we might consider a transformation.

```
> attach(rat)
> boxplot(insulin ~ vein*time,
cex.lab=1.3,cex.axis=1.3)
```

Boxplots

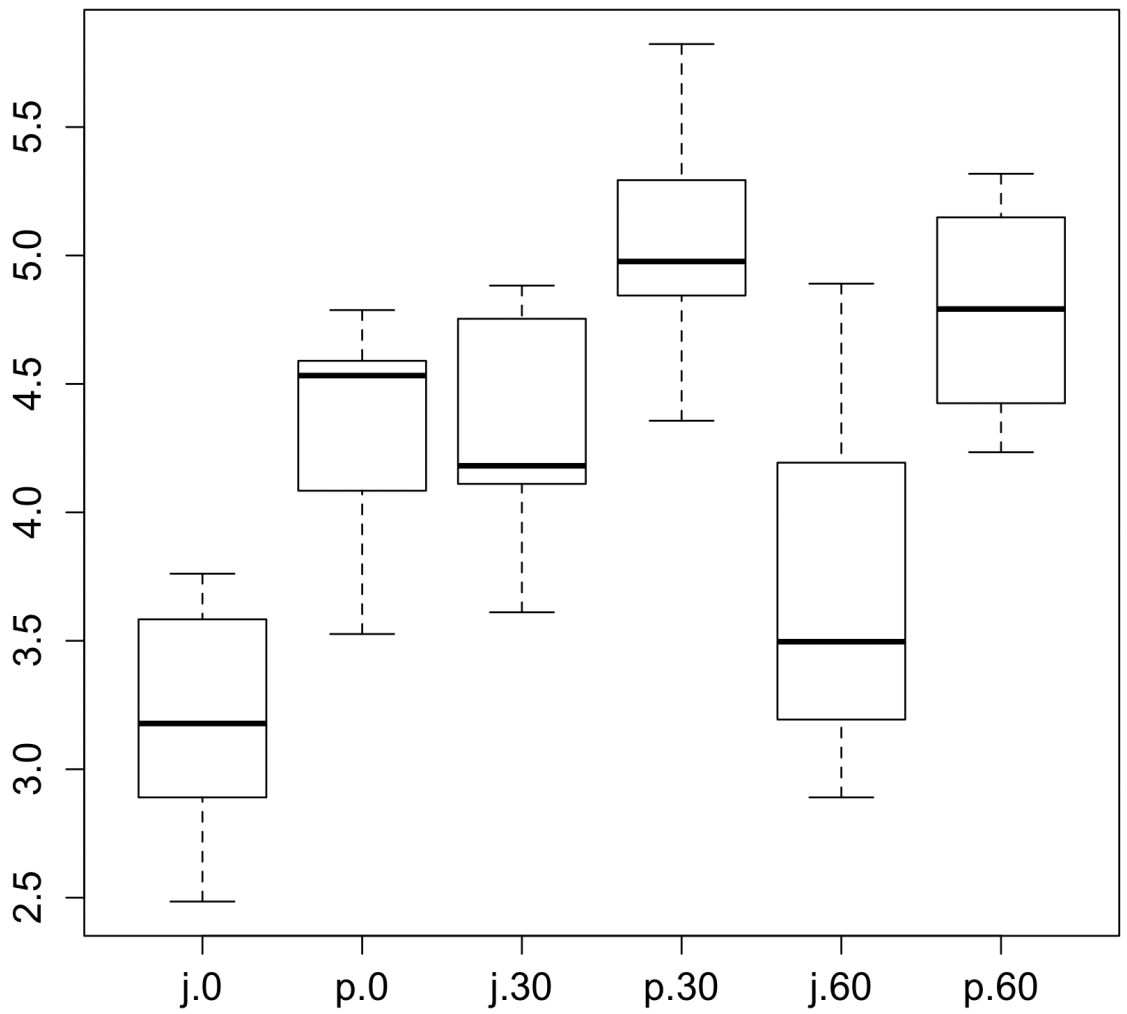


Unbalanced ANOVA

Here are the boxplots with log-transformed data, whcih looks a little better and gets rid of the outlier.

```
> boxplot(insulin ~ vein*time,
cex.lab=1.3,cex.axis=1.3)
```

Boxplots



Unbalanced ANOVA

For unbalanced ANOVA, type I versus type III sums of squares have a different meaning. Type I sums of squares are sequential, meaning that measure variation contributed by the given variable given previous variables in the model. This means that type I sums of squares are sensitive to the input order of the variables. Typically we use type III sums of squares instead.

Unbalanced ANOVA

Although some of the formulas under the hood are different for balanced versus unbalanced ANOVA, a lot of the R commands are the same. For this example, we can fit models using either the log response or original response:

```
> m1 <- lm(insulin ~ vein*time)
> library(car)
AWarning message:
package ?car? was built under R version 3.4.3
> Anova(m1,type=3)
Anova Table (Type III tests)

Response: insulin
             Sum Sq Df F value  Pr(>F)
(Intercept)   8868  1  2.7977 0.10150
vein         16897  1  5.3308 0.02571 *
time          3505  1  1.1057 0.29876
vein:time       23  1  0.0072 0.93278
Residuals   139468 44
```

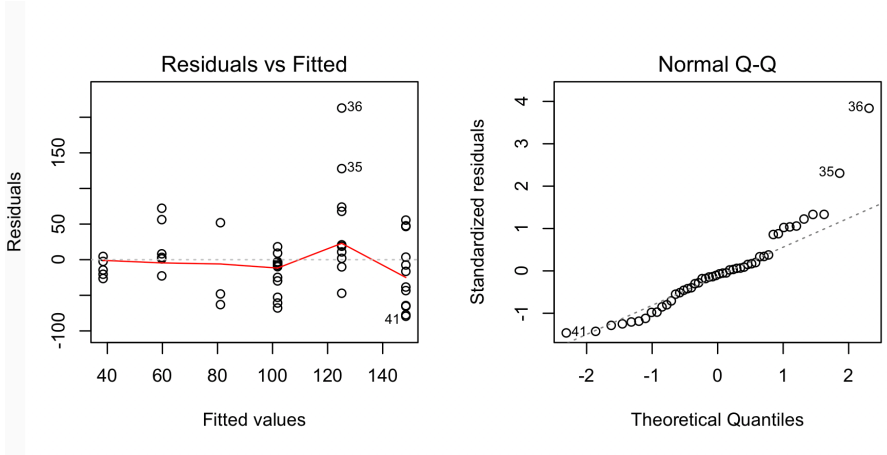
Unbalanced ANOVA

```
> Anova(m2,type=3)
Anova Table (Type III tests)

Response: log(insulin)
             Sum Sq Df  F value    Pr(>F)
(Intercept) 71.502  1 233.6279 < 2.2e-16 ***
vein         4.517  1  14.7595 0.0003884 ***
time         1.117  1   3.6505 0.0625818 .
vein:time    0.144  1   0.4698 0.4966530
Residuals   13.466 44
```

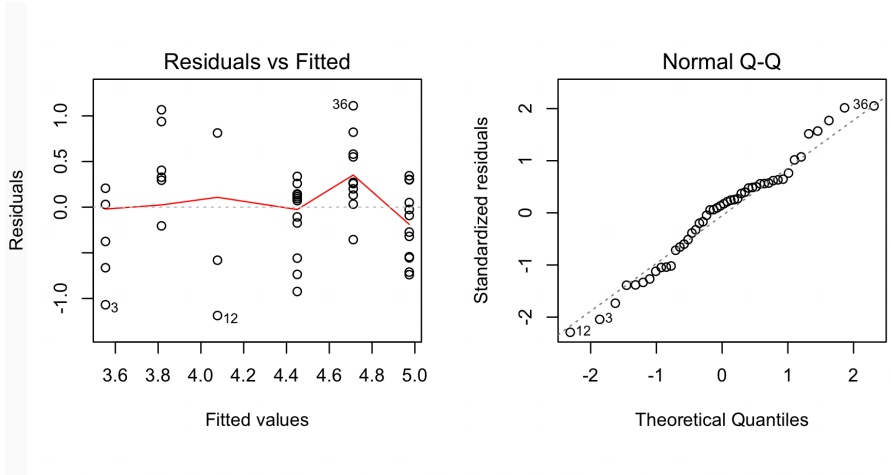
Unbalanced ANOVA

```
> m3 <- lm(log(insulin) ~ vein + time)
> Anova(m3,type=3)
Anova Table (Type III tests)

Response: log(insulin)
             Sum Sq Df  F value    Pr(>F)
(Intercept) 133.519  1 441.4671 < 2.2e-16 ***
vein          7.917  1  26.1752 6.242e-06 ***
time          2.174  1   7.1866   0.01023 *
Residuals    13.610 45
```

M1:


Residuals vs Fitted
Residuals
Fitted values
36
35
41
Normal Q-Q
Standardized residuals
Theoretical Quantiles
36
35
41

M3:


Residuals vs Fitted
Residuals
Fitted values
36
12
3
Normal Q-Q
Standardized residuals
Theoretical Quantiles
36
12
3

ANCOVA

ANCOVA (Analysis of covariance) is a linear model that allows you to compare two (or more) groups while adjusting for one or more quantitative covariates. Similarly, you might be interested in comparing the relationship between two quantitative variables while accounting for group differences.

Failing to account for one of the variables that isn't of interesting can lead to (unintentially) misleading relationships.

ANCOVA

As a made up example, suppose you have a sample of books of different lengths, some of which are hardcover, some softcover. Maybe these are books you buy at the beginning of a semester.

title	category	length	price
Applied Linear Statistical Models	soft	1396	54.64
It	soft	1000	12.95
War & Peace	soft	1300	14.95
Wizard of Oz (5 volumes)	hard	1200	7.98

ANCOVA

As another example, suppose you recruit 20 patients for a study and wish to determine whether treatment A or treatment B is more effective. Although patients are assigned randomly to treatments, you might notice that due to the small sample size, patients receiving treatment A tend to be younger than those receiving treatment B. In this case, although randomization should have helped, you might wish to additionally take into account ages of patients in determining the effects of the two treatments.

As another example, suppose you want to test whether Toyota minivans are more expensive than Honda minivans in Albuquerque's used car market. Here you don't have a controlled experiment, and you might have limited data (for example, looking on craigslist). It is unlikely that the mileages or model years of the cars being compared will be exactly the same, so you would want to take that into account in your model.

ANCOVA

In these examples, you are more interested in comparing group means (the usually ANOVA setting). However, the model is the same if you are more interested in the relationship between a quantitative predictor and the (quantitative) response, and need to adjust for qualitative factors in the data. For example, if you want to see the effect of mileage on car price, you might want to adjust for the fact that different brands (e.g, Honda versus Toyota) might have different selling prices even when all other variables are equal.

ANCOVA

As an extreme example, suppose you are interested in the relationship of book length to book price. You might think that longer books tend to be more expensive. This might be true within categories (longer novels might be more expensive than shorter novels, longer hardback textbooks tend to be more expensive than shorter hardback textbooks). But if you ignore the category, it might be hard to compare. Are shorter hardcover textbooks less expensive than long novels?

Ignoring category differences can lead to some misleading relationships. For example if you just have long novels and short textbooks in your sample, you might conclude that length is negatively related to price.

ANCOVA

The simplest approach to ANCOVA is to adjust the intercept for each group. The model is

$$\text{response} = \text{Grand mean} + \text{group} + \text{covariate}$$

or

$$y_{ij} = \mu + \alpha_i + \beta x_{ij} + \varepsilon_{ij}$$

where $\mu_i = \mu + \alpha_i$ is the intercept for group i.
The effect of the model is that there is a separate regression line for each group, but the regression lines are assumed to be parallel. The effect of belonging to a particular group is to shift the regression line up or down. Typically, the main interest is in testing whether $\alpha_i = 0$ for each i, meaning that there

ANCOVA

As an example, we'll use the cars data and compare prices of cars with salvage versus clean titles. Does the title status matter for the price of the car? As a reminder, here is the data:

```
> x <- read.table("cars2.txt",header=T)
```

```
   year price  miles   title
1  1995  1200 150000   clean
2  2004  4500 184000 salvage
3  1995  3200    NaN   clean
4  1998  1850 152000 salvage
5  1998  3400 136000   clean
6  2004  8500  85500   clean
7  2007 12400  89000   clean
8  2002  5450 137000   clean
9  2007 18500  64000   clean
10 1996 15000 134000   clean
11 2008 13999 143934   clean
12 1997  2500    NaN salvage
13 2007  8500 129000   clean
14 2003   NaN    NaN salvage
15 1986  4500 190291   clean
16 1983  4300    NaN rebuilt
17 1976  4500 131000   clean
18 1967 10500    NaN   clean
19 2010   NaN  66471      Na
20 2008  8995  13500   clean
```

To clean up the data, since we are interested in title status of clean versus salvage, we'll remove the two observations with title status either missing or "rebuilt". We can just remove the bad rows from the dataset.

```
> x2 <- x[-c(16,19),]
```

To start, we'll consider price the response and take into account only the model year. This is partly motivated by the miles variable having a lot of missing values.

```
> mymodel <- lm(x2$price ~ x2$title + x2$year)
> summary(mymodel)
Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -256639.9   203310.9  -1.262   0.2275
x2$titlesalvage   -5897.6     2993.6  -1.970   0.0689 .
x2$year             132.8      101.8   1.304   0.2133
```

Based on the model, there is slight, but not strong evidence that salvage titles have lower prices than. The direction of the effect comes from the sign of the coefficient, which is negative. The coefficient here is -5897.6, meaning that having the salvage title reduces the price an estimated $5897.60, which seems like quite a lot. The range of prices for these used cars is $2500 to $18500, so this seems like a pretty large difference in prices. For the amount of variability in the data, however and the small sample size, it is not statistically significant at the .05 level. Note that there are only three nonmissing cases with a salvage title, so there is not much data to estimate its effect.

If you just do a t-test on prices of salvage titles versus clean titles, then you are exaggerating the effect of the salvage title by not taking into account that these are older cars. On the other hand, the effect of the year is not significant, so you could think of fitting the model with year in it. To some extent, which model you prefer depends on your interest. If you are primarily interested in estimating the effect of the salvage title, I would leave year in the model to not exaggerating the effect. If you are primarily interested in the factors that effect the price, the title status appears more important than the year.

Based on the output, we can draw two regression lines: one for salvage titles, and one for clean:

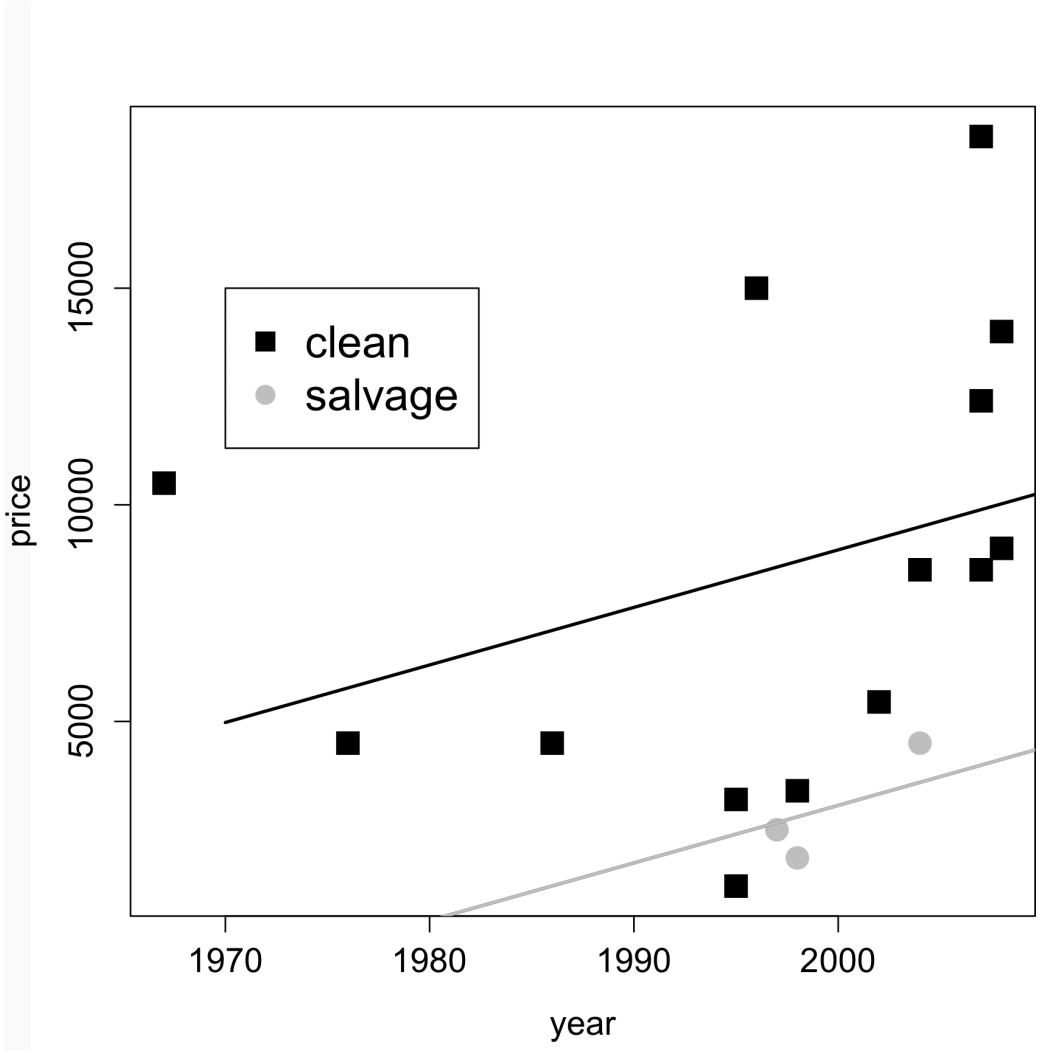
$$\text{price} = -256639.9 - 5897.6 + 132.8(\text{year}) \quad \text{for salvage}$$

$$\text{price} = -256639.9 + 132.8(\text{year}) \quad \text{for clean}$$

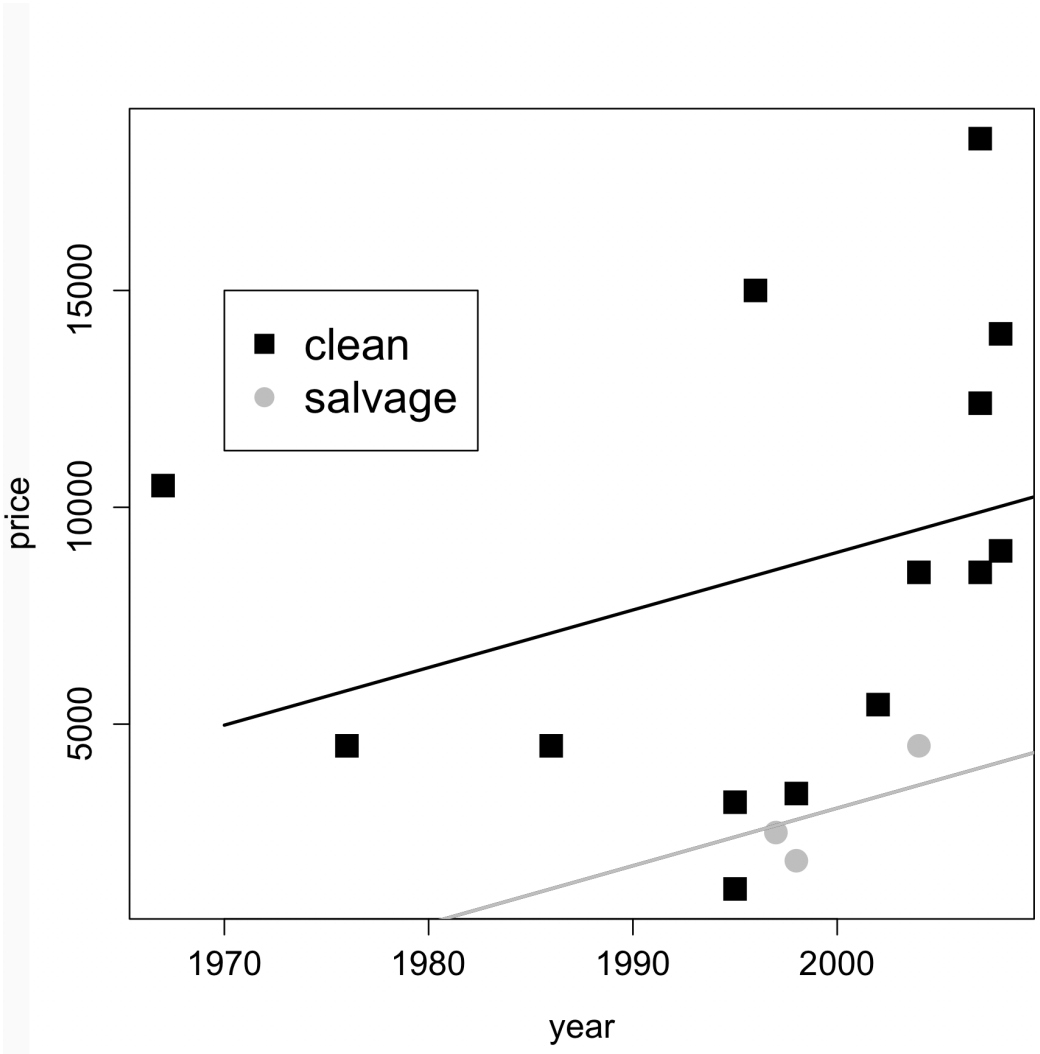
```
> summary(mymodel)
Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     -256639.9   203310.9  -1.262   0.2275
x2$titlesalvage   -5897.6     2993.6  -1.970   0.0689 .
x2$year             132.8      101.8   1.304   0.2133
```

```
> plot(x2$year[x2$title=="clean"],x2$price[x2$title=="clean"],
pch=15,cex=2,xlab="year",ylab="price",cex.lab=1.3,cex.axis=1.3)
> points(x2$year[x2$title=="salvage"],
x2$price[x2$title=="salvage"],pch=16,cex=2,col="grey")
> legend(1970,15000,legend=c("clean","salvage"),pch=c(15,16),
col=c("black","grey"),cex=1.7)
> xaxis <- 1970:2015
> price1 <- -256639.9 + 132.8*xaxis
> price2 <- -256639.9 -5897.6 + 132.8*xaxis
> points(xaxis,price1,type="l",lwd=2)
> points(xaxis,price2,type="l",lwd=2,col="grey")
```

ANCOVA


clean
salvage
price
5000
10000
15000
1970
1980
1990
2000
year

Dotted line is simple linear regression ignoring



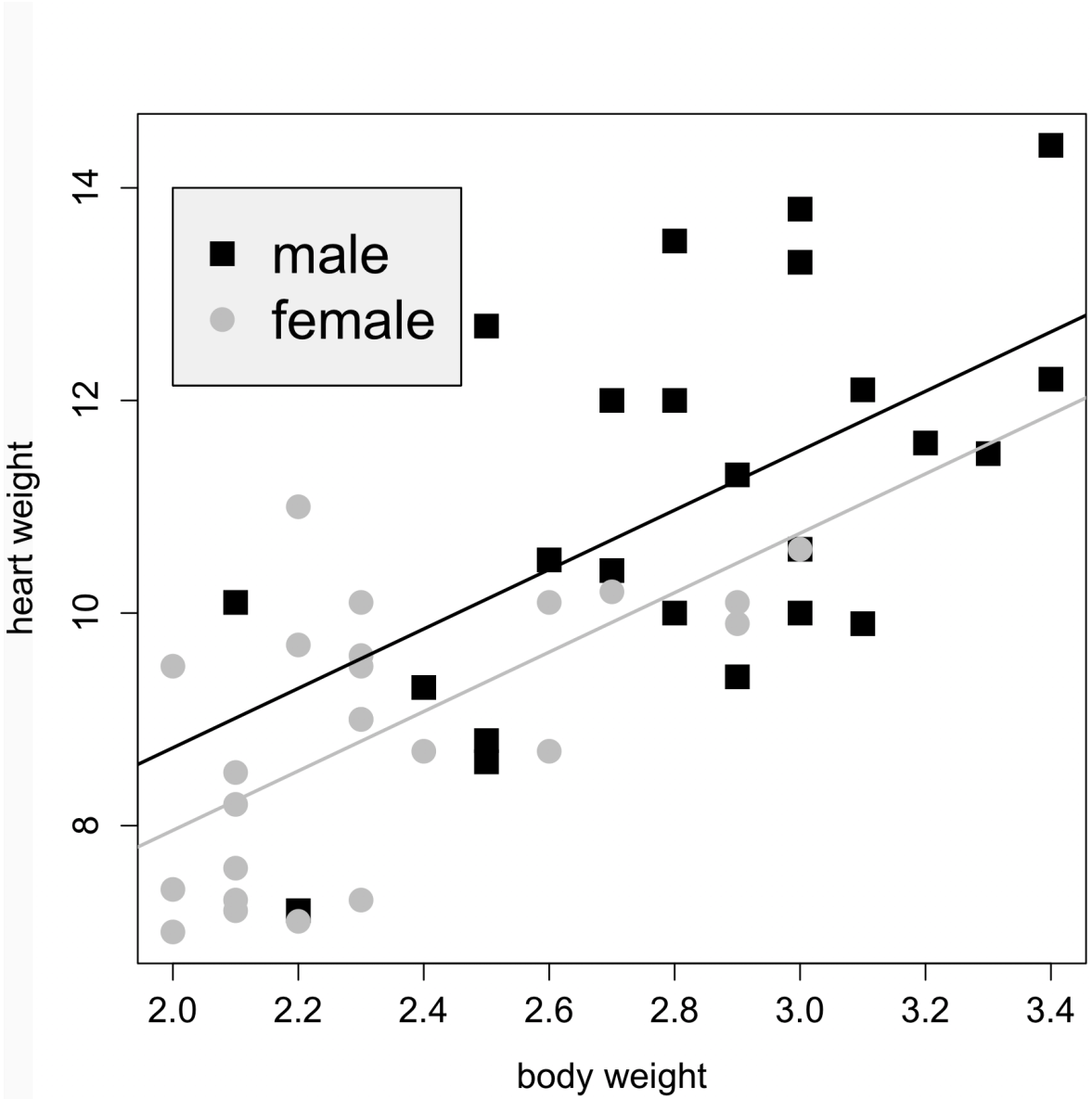
ANCOVA

As another example, we consider a data set used by Fisher (1947) on body weights and heart weights of cats given digitalis (a type of drug). The question of interest was whether female cats versus male cats had different heart weights adjusting for the fact that males are larger.

```
> x <- read.table("digitalis.txt",header=T)
> a1 <- lm(x$heart ~ x$body + x$sex)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.3663     1.3684   1.729   0.0906 .
x$body        2.7948     0.5759   4.853  1.5e-05 ***
x$sexm        0.7767     0.4641   1.674   0.1011
```

ANCOVA

Note that the output here means that heart sizes weren't significantly different for male versus female cats once body size was accounted for. In other words, differences in heart size are accounted for by differences in body size.


male
female
heart weight
body weight
8
10
12
14
2.0
2.2
2.4
2.6
2.8
3.0
3.2
3.4

Note that just doing a t-test ignoring the sex of the cats would lead to a highly significant result (just doing a t-test on heart weights for female versus male cats). This is not wrong–it is addressing a different question. The t-test is just asking whether heart weights are different, not whether heart weights are different adjusting for body weight.

ANCOVA

An interaction model for ANCOVA means that different groups can have different slopes as well as different intercepts. The model can be written this way

$$y_{ij} = \mu + \alpha_i + \beta_i x_{ij} + \varepsilon_{ij}$$

An example is the iris dataset (also from Fisher) which compares petal lengths and widths, and sepal lengths and widths for three species of iris: *setosa*, *versicolor*, and *virginica*.

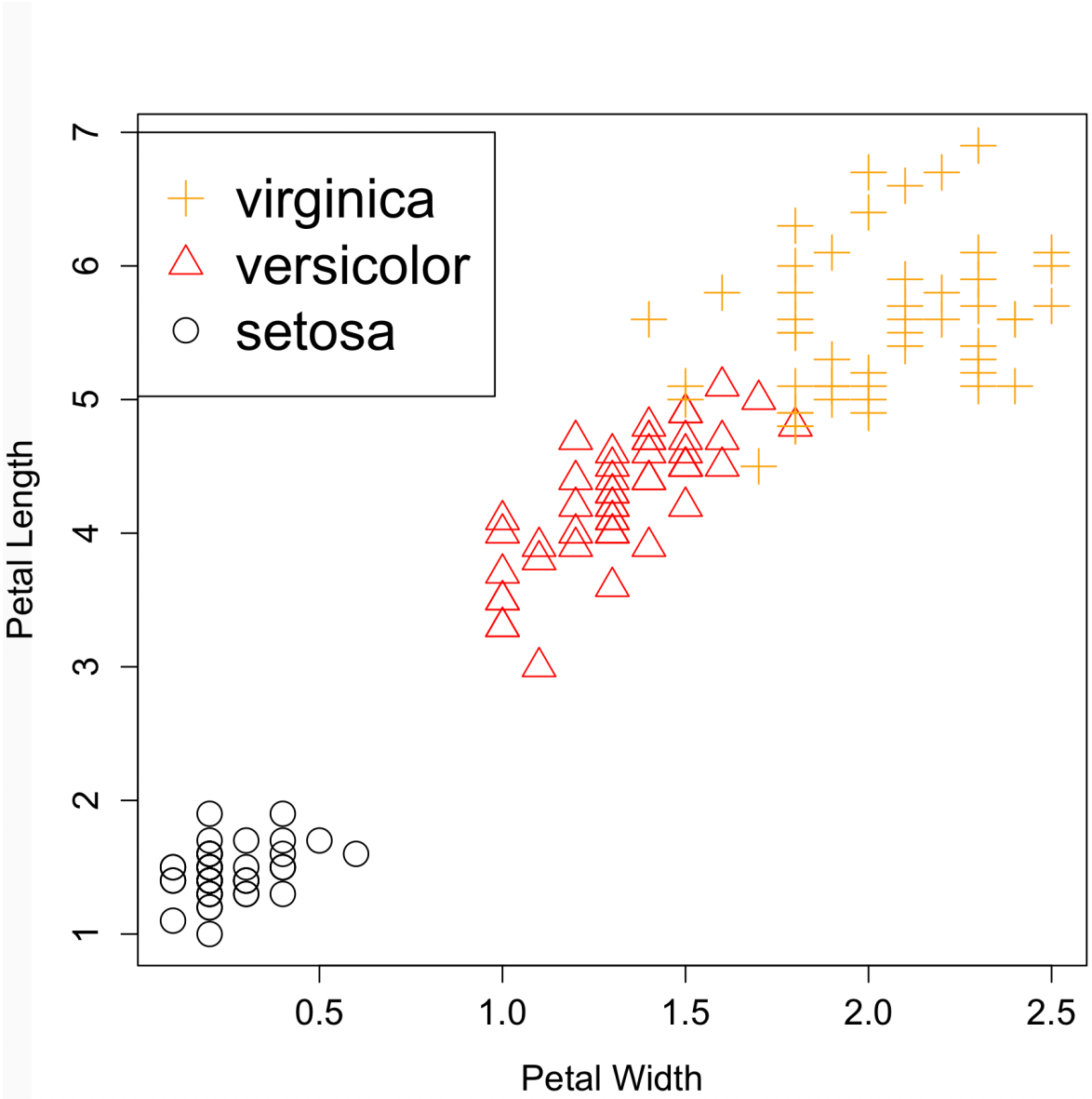
source: https://image.slidesharecdn.com/irisdataanalysiswithr-140801203600-phpapp02/95/iris-data-analysis-example-in-r-3-638.jpg?cb=1406925587

ANCOVA

```
> data(iris)
> head(iris)
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

ANCOVA

```
> plot(iris$Petal.Width,iris$Petal.Length,type="n"
,xlab="Petal Width",ylab="Petal Length",cex.lab=1.3,cex.axis
> mypch = c(rep(1,50),rep(2,50),rep(3,50))
> mycol = c(rep("black",50),rep("red",50),
rep("orange",50))
> points(iris$Petal.Width,iris$Petal.Length,
col=mycol,pch=mypch,cex=2)
> legend(0,7,legend=c("virginica","versicolor","setosa"),
pch=c(1,2,3),col=c("orange","red","black"),cex=2)
```


virginica
versicolor
setosa
Petal Length
Petal Width
1
2
3
4
5
6
7
0.5
1.0
1.5
2.0
2.5

ANCOVA

```
> m1 <- lm(iris$Petal.Length ~ iris$Petal.Width*iris$Species)
> Anova(m1,type=3)
Response: iris$Petal.Length
                              Sum Sq  Df  F value    Pr(>F)
(Intercept)                  13.4329   1 102.8050 < 2.2e-16 ***
iris$Petal.Width              0.1625   1   1.2438 0.2665889
iris$Species                  6.7474   2  25.8196 2.614e-10 ***
iris$Petal.Width:iris$Species 2.0178   2   7.7213 0.0006525 ***
Residuals                    18.8156 144
```

ANCOVA

```
Coefficients:
                                          Estimate Std. Error t valu
(Intercept)                                 1.3276     0.1309  10.13
iris$Petal.Width                            0.5465     0.4900   1.11
iris$Speciesversicolor                      0.4537     0.3737   1.21
iris$Speciesvirginica                       2.9131     0.4060   7.17
iris$Petal.Width:iris$Speciesversicolor     1.3228     0.5552   2.38
iris$Petal.Width:iris$Speciesvirginica      0.1008     0.5248   0.19

(Intercept)                                ***
iris$Petal.Width
iris$Speciesversicolor
iris$Speciesvirginica                      ***
iris$Petal.Width:iris$Speciesversicolor    *
iris$Petal.Width:iris$Speciesvirginica
```

ANCOVA

```
> m1
Call:
lm(formula = iris$Petal.Length ~ iris$Petal.Width * iris$Species)

Coefficients:
                                (Intercept)
                                     1.3276
                           iris$Petal.Width
                                     0.5465
                  iris$Speciesversicolor
                                     0.4537
                   iris$Speciesvirginica
                                     2.9131
iris$Petal.Width:iris$Speciesversicolor
                                     1.3228
 iris$Petal.Width:iris$Speciesvirginica
                                     0.1008
```

ANCOVA

To get the regression lines, we have one per species, each with a different slope. *Setosa* is the baseline, so for this species we have

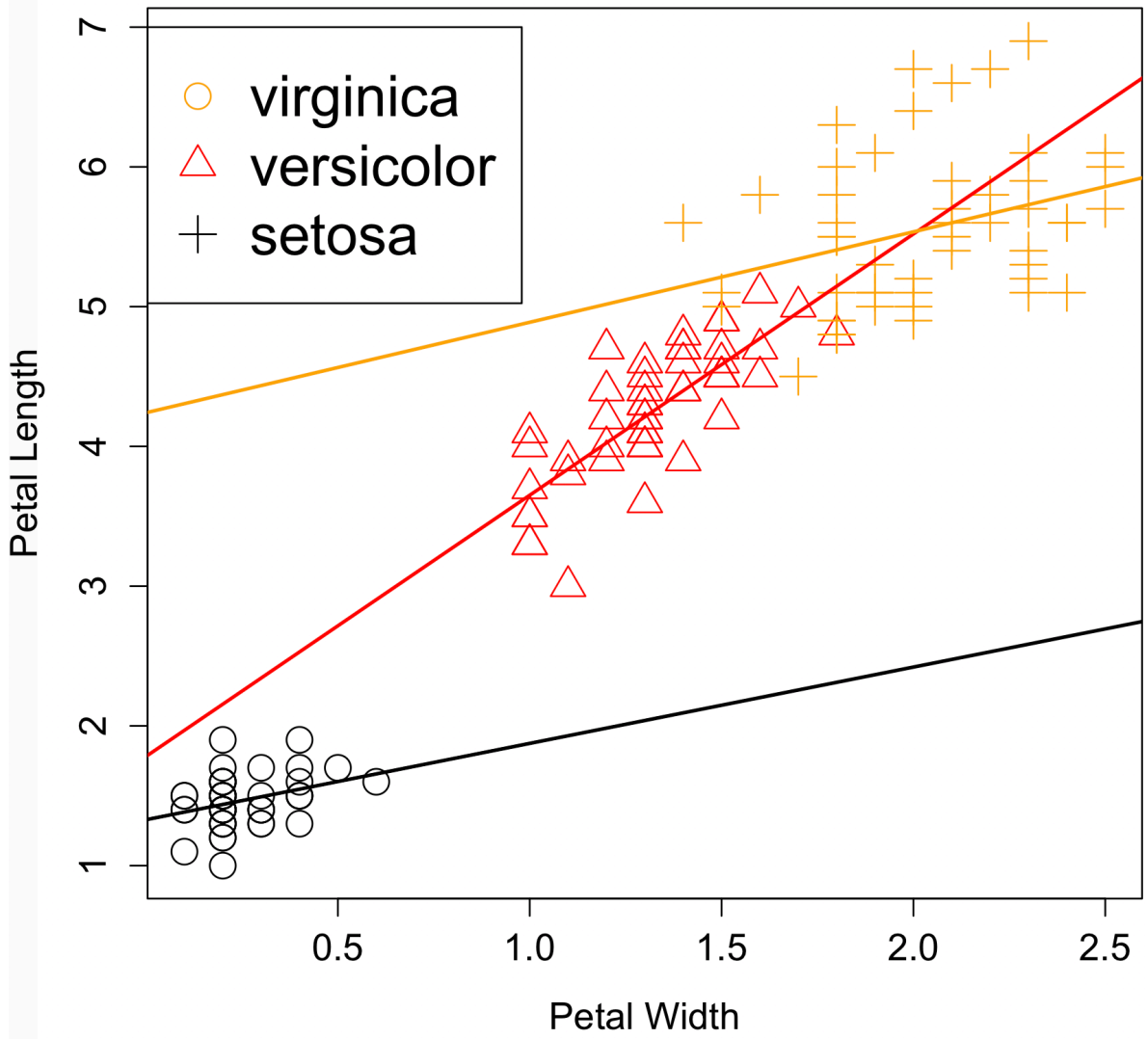
$$\text{Petal length} = 1.3276 + 0.5465 \times \text{Petal width}$$

For *versicolor*

$$\begin{aligned}\text{Petal length} &= 1.3276 + 0.4537 + (0.5465 + 1.3228) \times \text{Petal width} \\ &= 1.7813 + 1.8693 \times \text{Petal width}\end{aligned}$$

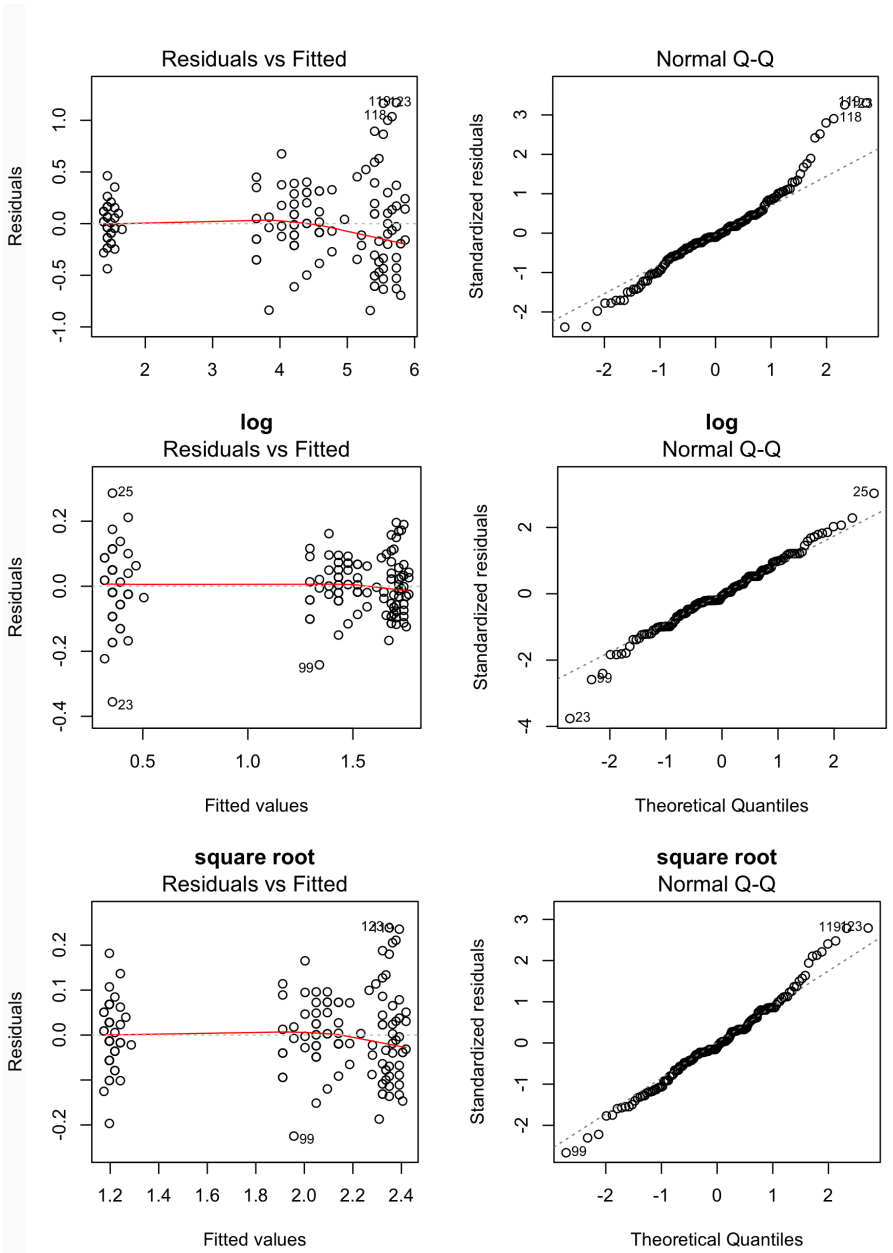
For *virginica*,

$$\begin{aligned}\text{Petal length} &= 1.3276 + 2.9131 + (0.5465 + 0.1008) \times \text{Petal width} \\ &= 4.2407 + 0.6473 \times \text{Petal width}\end{aligned}$$


virginica
versicolor
setosa
Petal Length
Petal Width
1
2
3
4
5
6
7
0.5
1.0
1.5
2.0
2.5

ANCOVA

To check model assumptions, you can again look at residual plots and plots of residuals. I tried a few transformations of the petal lengths: logarithmic and square root. Based on residual plots, the original data results in funnel-shaped residuals against fitted values. A logarithmic transformation of the response tends to overcorrect, leading to a funnel in the opposite direction, while a square-root transformation makes the residual plots look more reasonable.

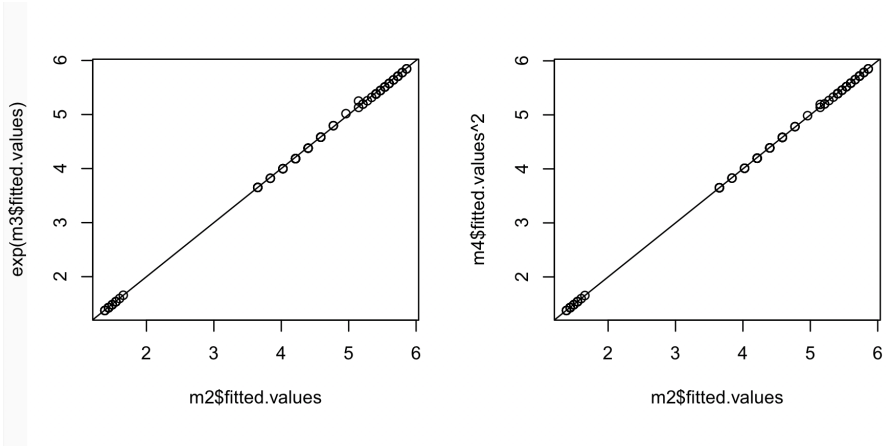

Residuals vs Fitted
Residuals
Normal Q-Q
Standardized residuals
log
Residuals vs Fitted
Residuals
Fitted values
log
Normal Q-Q
Standardized residuals
Theoretical Quantiles
square root
Residuals vs Fitted
Residuals
Fitted values
square root
Normal Q-Q
Standardized residuals
Theoretical Quantiles

ANCOVA

I won't go into the details here, but there isn't much difference in the models using the different transformations–for all of them the interaction is highly significant. Since there isn't much difference, you might just use the original data without a transformation since that is easier to interpret.

You could also compare fitted values from the different models. Here `m2` uses untransformed data, `m3` uses log-transformed response, and `m4` uses the square root transformation. To compare fitted values, you would have to transform predicted values back to the original scale

```
> plot(m2$fitted.values,exp(m3$fitted.values))
> plot(m2$fitted.values,m4$fitted.values^2)
```


exp(m3$fitted.values)
m2$fitted.values
2
3
4
5
6
m4$fitted.values^2
m2$fitted.values

Polynomial regression

Instead of transforming responses, another possibility is to either transform predictor variables or add polynomial functions of predictor variables. This is especially done when the relationship between response and predictors appears to be curvilinear, such as when the response is either maximized or minimized by intermediate values of a response variable. In this case the response might be considered a polynomial (e.g., quadratic, cubic, etc.) function of the predictor(s).

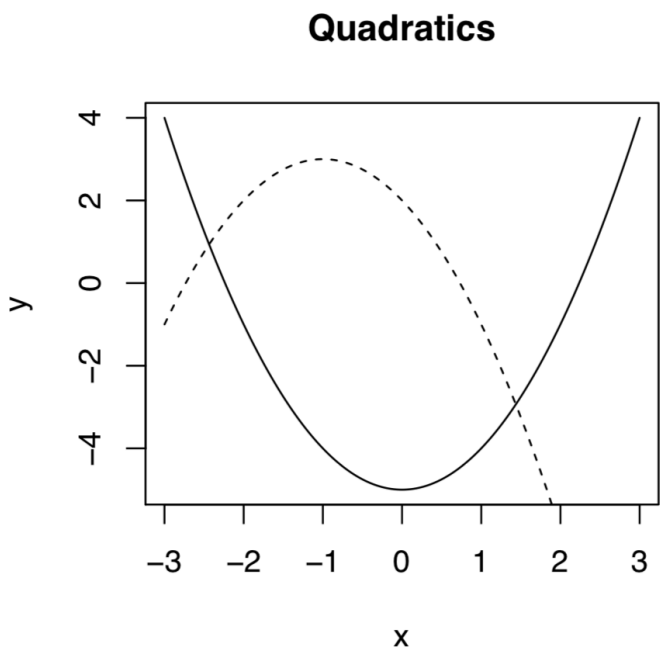
Polynomial regression

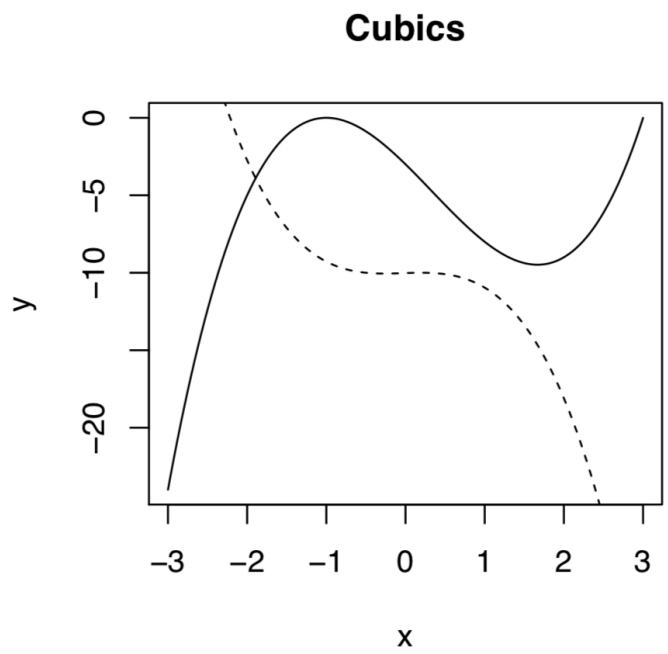
If there is only one predictor variable, the model can be written as

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_p x_i^p + \varepsilon$$

Often, only $p = 2$ or $p = 3$ is used. Here are some examples in R:

```
x <- seq(-3,3,0.01);
y21 <- x^2-5;
y22 <- -(x+1)^2+3;
y31 <- (x+1)^2*(x-3);
y32 <- -(x-.2)^2*(x+.5)-10;
plot(  x, y21, type="l", main="Quadratics", ylab="y")
points(x, y22, type="l", lt=2)
plot(  x, y31, type="l", main="Cubics", ylab="y")
points(x, y32, type="l", lt=2)
```


Quadratics
y
4
2
0
−2
−4
−3
−2
−1
0
1
2
3
x


Cubics
y
0
−5
−10
−20
−3
−2
−1
0
1
2
3
x

Polynomial regression

Note that a polynomial relationship might be useful even if the maximum and minimum points are not within the range of the predictor (for example on the left hand graph if only $x > 0$ is observed), simply because it allows a nonlinear relationship. Also note that linear regression is a special case of polynomial regression. For example, in the model

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \varepsilon_i$$

if $\beta_2 = 0$, then the model reduces to simple linear regression. Testing the null hypothesis $H_0 : \beta_2 = 0$ in this case could be used to decide whether the relationship is linear versus quadratic.

Polynomial regression

The equation

$$y = \beta_0 + \beta_1 x + \beta_2 x^2$$

is the equation of a parabola. Another way of expressing this (using completing the square)

$$y = \beta_0 + \beta_2 \left(\frac{\beta_1}{\beta_2} x + x^2 \right)$$

$$y = \beta_0 + \beta_2 \left(x + \frac{\beta_1}{2\beta_2} \right)^2 - \frac{\beta_1^2}{2\beta_2}$$

$$y = \beta_0 - \frac{\beta_1^2}{2\beta_2} + \beta_2 \left(x + \frac{\beta_1}{2\beta_2} \right)^2$$

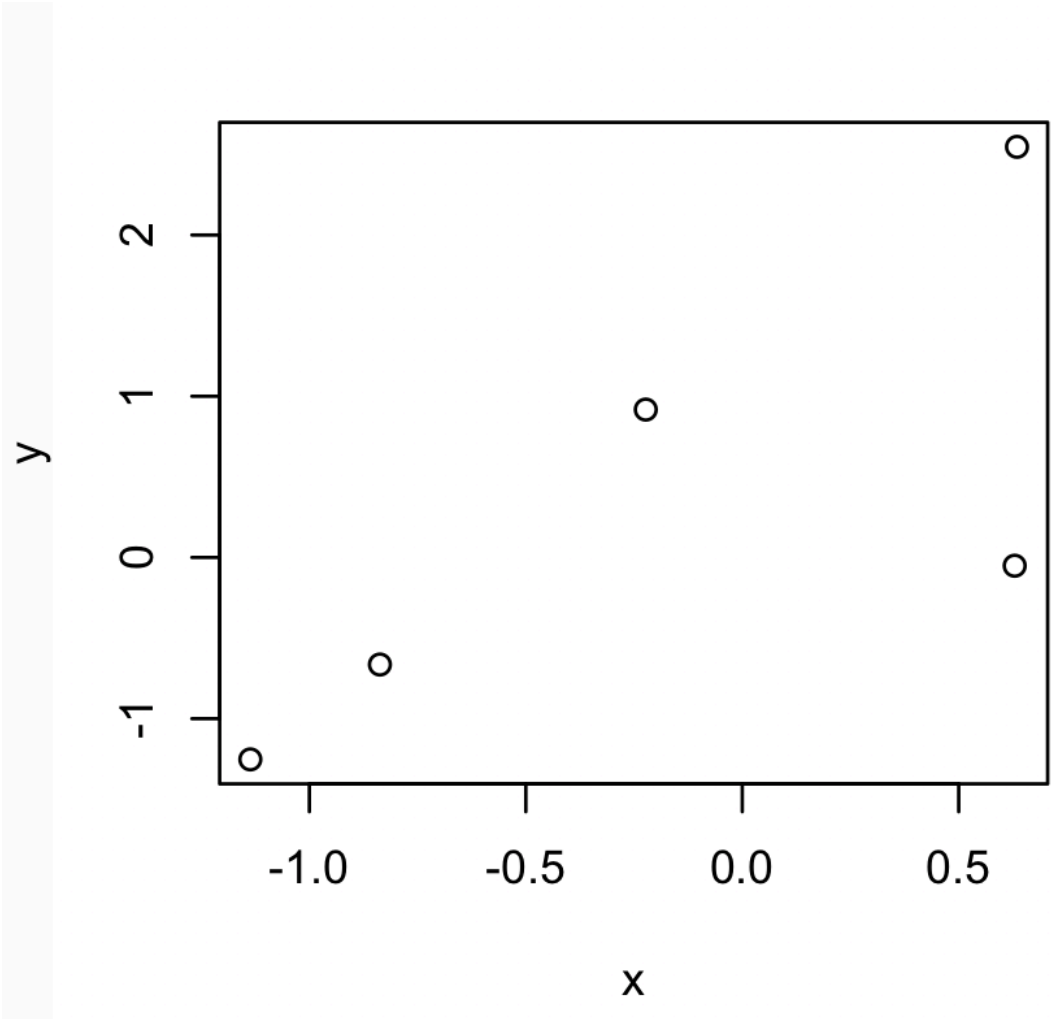
This means that the parabola is centered at $-\frac{\beta_1}{2\beta_2}$. If β_k is estimated by b_k, then the estimated regression curve is centered at $-b_1/(2b_2)$.

Polynomial regression

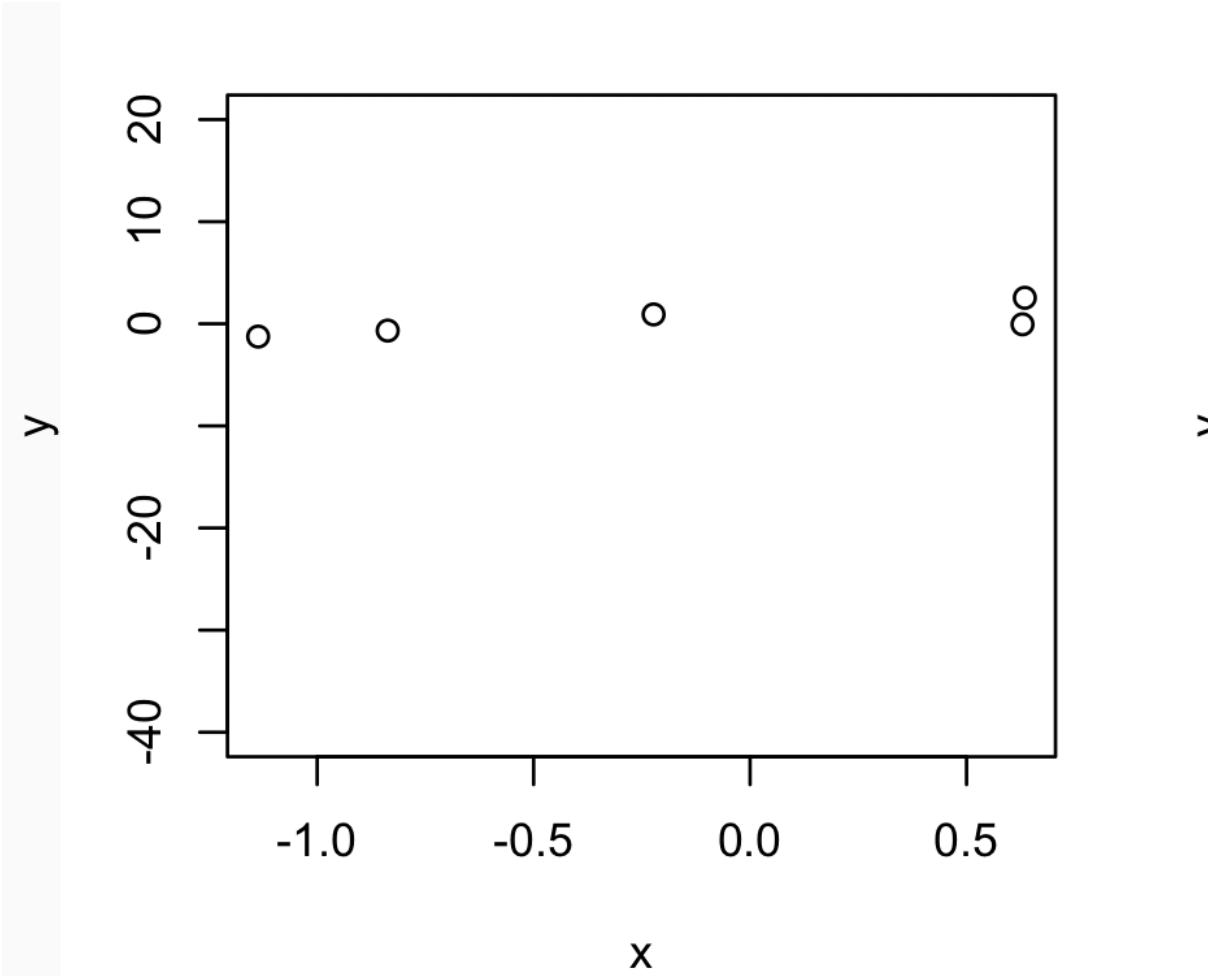
Polynomial regression can be fit by using new predictor variables based on powers of the original predictor `x`. Here is a toy example using powers up to x^4 with only 5 observations:

```
> x <- rnorm(5)
> y <- x+runif(5)
> x2 <- x^2
> x3 <- x^3
> x4 <- x^4
> a <- lm(y ~ x + x2 + x3 + x4)
> summary(a)
> x
[1]  0.6292986  0.6346305 -0.2228644 -1.1363222 -0.8370428
> y
[1] -0.05138843  2.54694578  0.91717318 -1.25256085 -0.66412
```

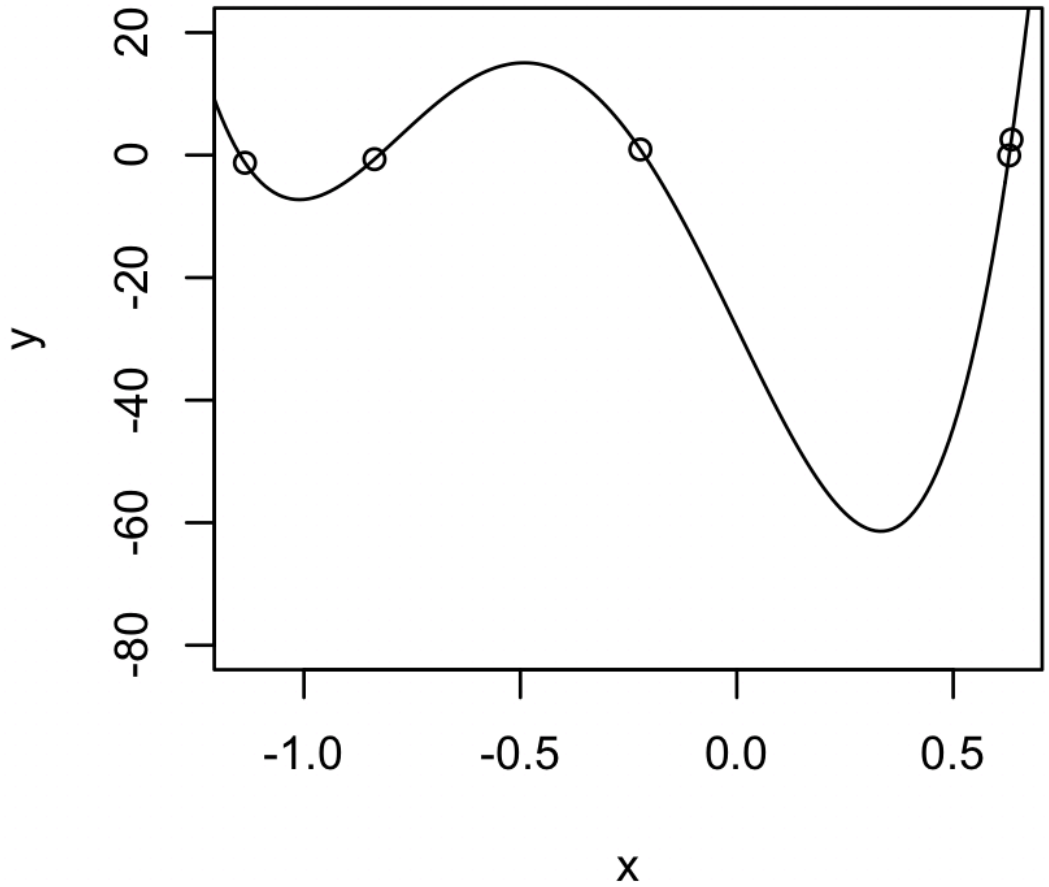
Polynomial regression



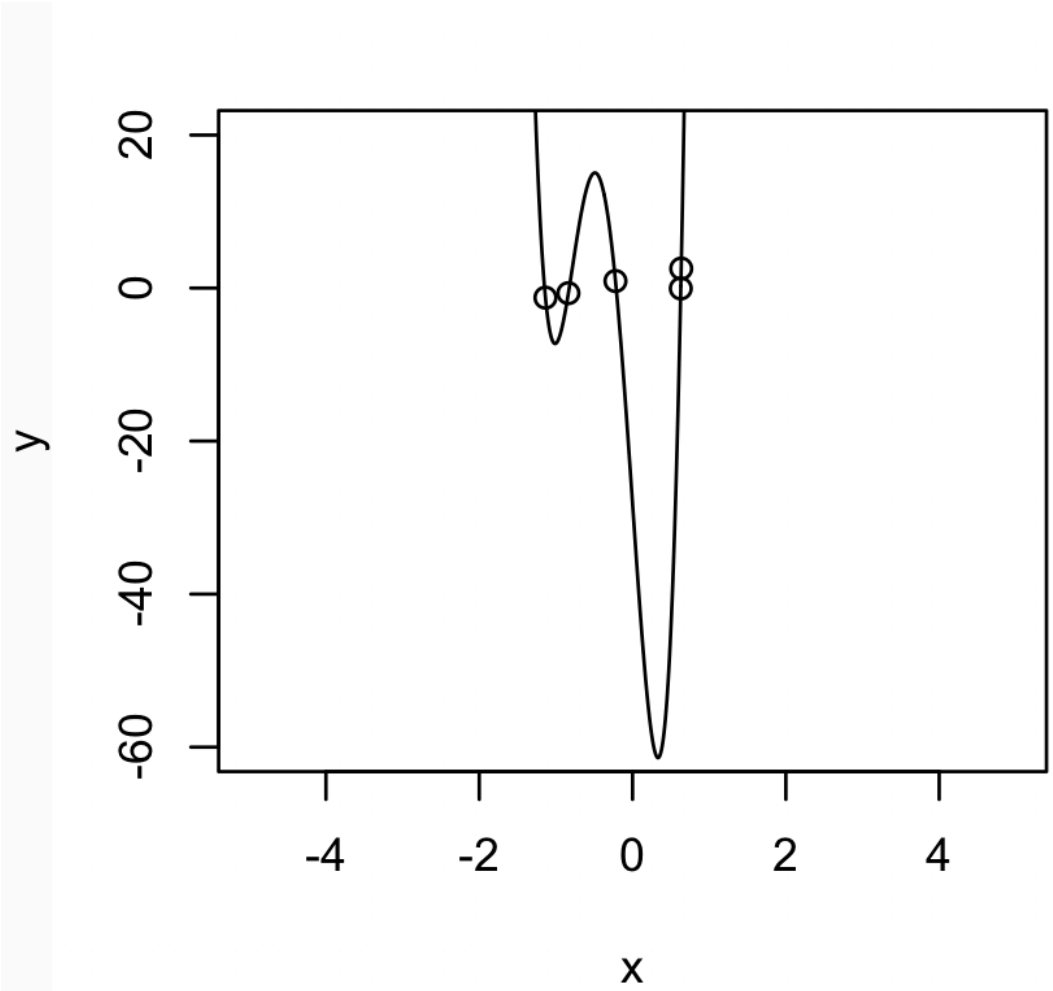
Polynomial regression



Polynomial regression



Polynomial regression



```
> a

Call:
lm(formula = y ~ x + x2 + x3 + x4)

Coefficients:
(Intercept)            x           x2           x3           x4
    -0.3325       8.9617      -2.9693     -47.6713     -35.8821

> plot(x,y,ylim=c(-60,20))
> xaxis <- seq(-2,2,.01)
> yhat <-  -0.3325 *8.9617*xaxis-2.9693*xaxis^2-
47.6713*xaxis^3-35.8821*xaxis^4
> points(xaxis,yhat,type="l")
```

Polynomial regression

One thing to notice about this is that with n observations, a polynomial regression with $p = n - 1$ predictors can **exactly** fit the data. This is not really a good thing. By doing this it tends to make some extreme predictions for potential data values that weren't observed. Normally having higher powers makes it more desirable to have larger sample sizes to avoid **overfitting**. The idea of overfitting is that the model fits the particular observations but is unlikely to generalize to a new data set collected from the same population.

Although extrapolation beyond the range of the data can be dangerous in linear regression, the situation is even worse in polynomial regression since it can lead to such extreme predictions.

Another issue in polynomial regression is that measurement scale (e.g., Celsius versus Fahrenheit), now can affect the results (p-vlaues, predicted values, etc.).

Polynomial regression

Another problem with using n observations to fit n parameters ($n-1$ coefficients plus the intercept) is that it doesn't allow any extra information to estimate uncertainty. As a result, the standard errors and p-values cannot be given.

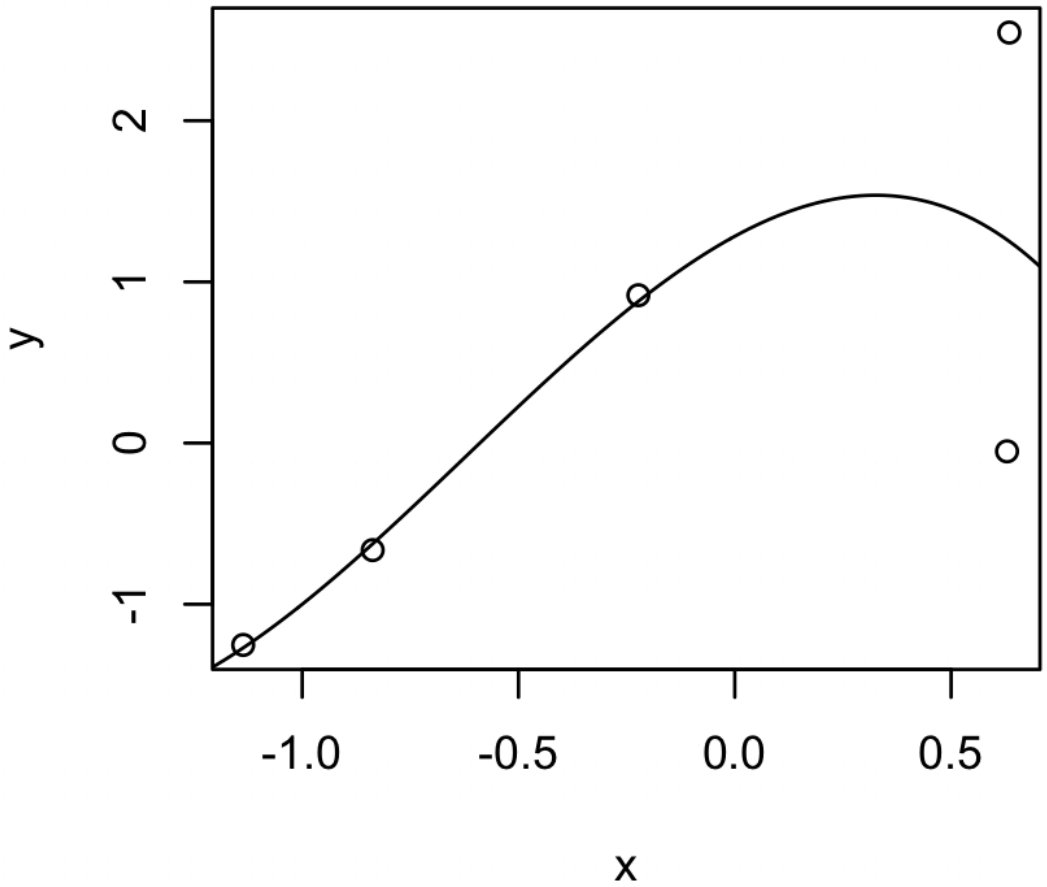
```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -28.190          NA      NA       NA
x           -145.540          NA      NA       NA
x2            -1.357          NA      NA       NA
x3           343.781          NA      NA       NA
x4           220.553          NA      NA       NA

Residual standard error: NaN on 0 degrees of freedom
Multiple R-squared:      1,Adjusted R-squared:    NaN
F-statistic:   NaN on 4 and 0 DF,  p-value: NA
```

Polynomial regression

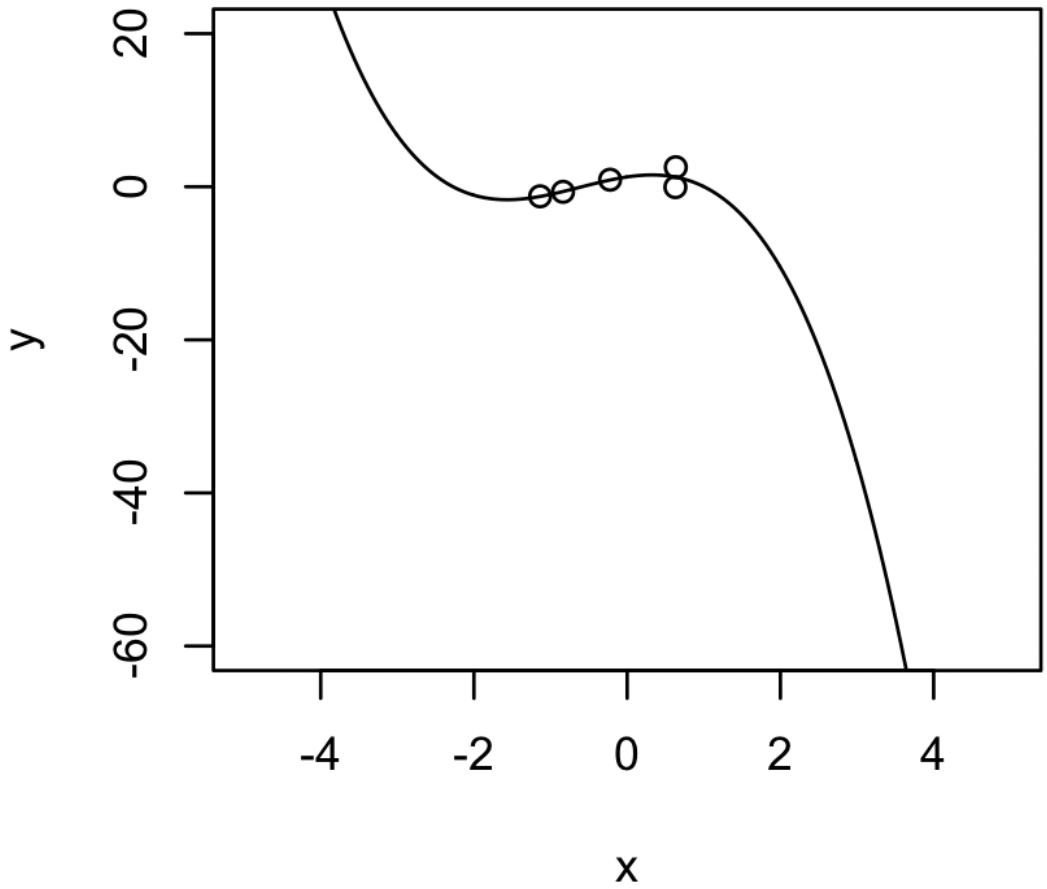
Amazingly, by having just one less parameter, you can suddenly get standard errors and p-values for all parameters. Note that none of the p-values indicates singificance even though the curve essentially goes through three of the five data points. With a small ratio of sample size to parameters, it is difficult to find significance.

```
> a2 <- lm(y ~ x + x2 + x3)
> summary(a2)
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.2830     2.5999   0.493    0.708
x             1.4618     4.7427   0.308    0.810
x2           -1.7765     7.4583  -0.238    0.851
x3           -0.9551     8.7390  -0.109    0.931
```

Polynomial regression: cubic fit



Polynomial regression: cubic fit



Polynomial regression: cubic fit

We also see that interpolation seems more reasonable in the cubic model compared to the quartic, but that extrapolation (beyond the range of the data) will lead to some very extreme predictions.

Polynomial regression: cubic fit

Although a simple linear regression is also not significant, the p-values have gone down and the standard errors are much smaller.

```
> a3 <- lm(y ~ x)
> summary(a3)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.5536     0.5283   1.048    0.372
x             1.3643     0.7009   1.946    0.147

Residual standard error: 1.145 on 3 degrees of freedom
Multiple R-squared:  0.5581,Adjusted R-squared:  0.4108
F-statistic: 3.788 on 1 and 3 DF,  p-value: 0.1468
```

It is also possible to have two or more predictors, each of which could be fit with quadratic, cubic or higher order terms. With more predictors, you could easily end up with huge numbers of parameters to estimate, which will require more data. Usually we want as few parameters as possible, and for polynomial regression, we usually want to just use quadratic or maybe cubic powers if possible.

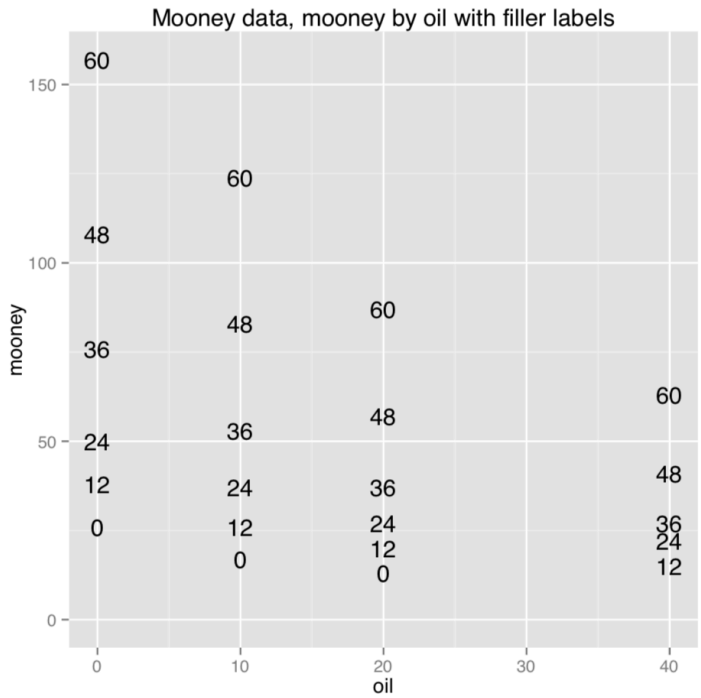
With two predictors, each of which could be quadratic, we can have the model

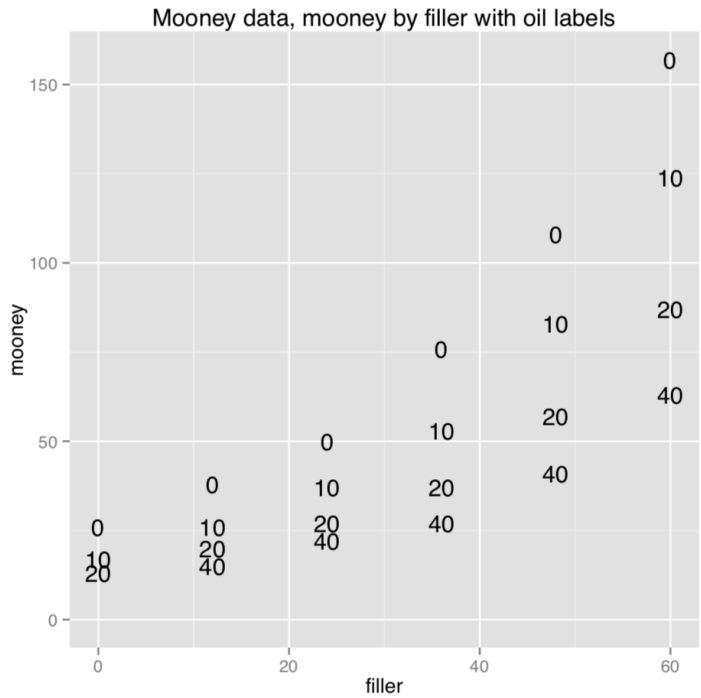
$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{1i}^2 + \beta_4 x_{2i}^2 + \beta_4 x_{1i} x_{2i} + \varepsilon_i$$

This model includes an interaction, which is still quadratic since the total power of $x_1 x_2$ is 2.

As an example, the data below give the Mooney viscosity at 100 degrees Celsius (y) as a function of the filler level (x_1) and the naphthenic oil (x_2) level for an experiment involving filled and plasticized elastomer compounds.

Polynomial regression: cubic fit





The graphic plots two variables—such as Mooney viscocity against oil, and instead of using a plotting character for each point, replaces it with the value of the third variable. This is a clever way to get three dimensional information into an apparently two-dimensional graph, and mostly works if you have a small number of values in the third variable.

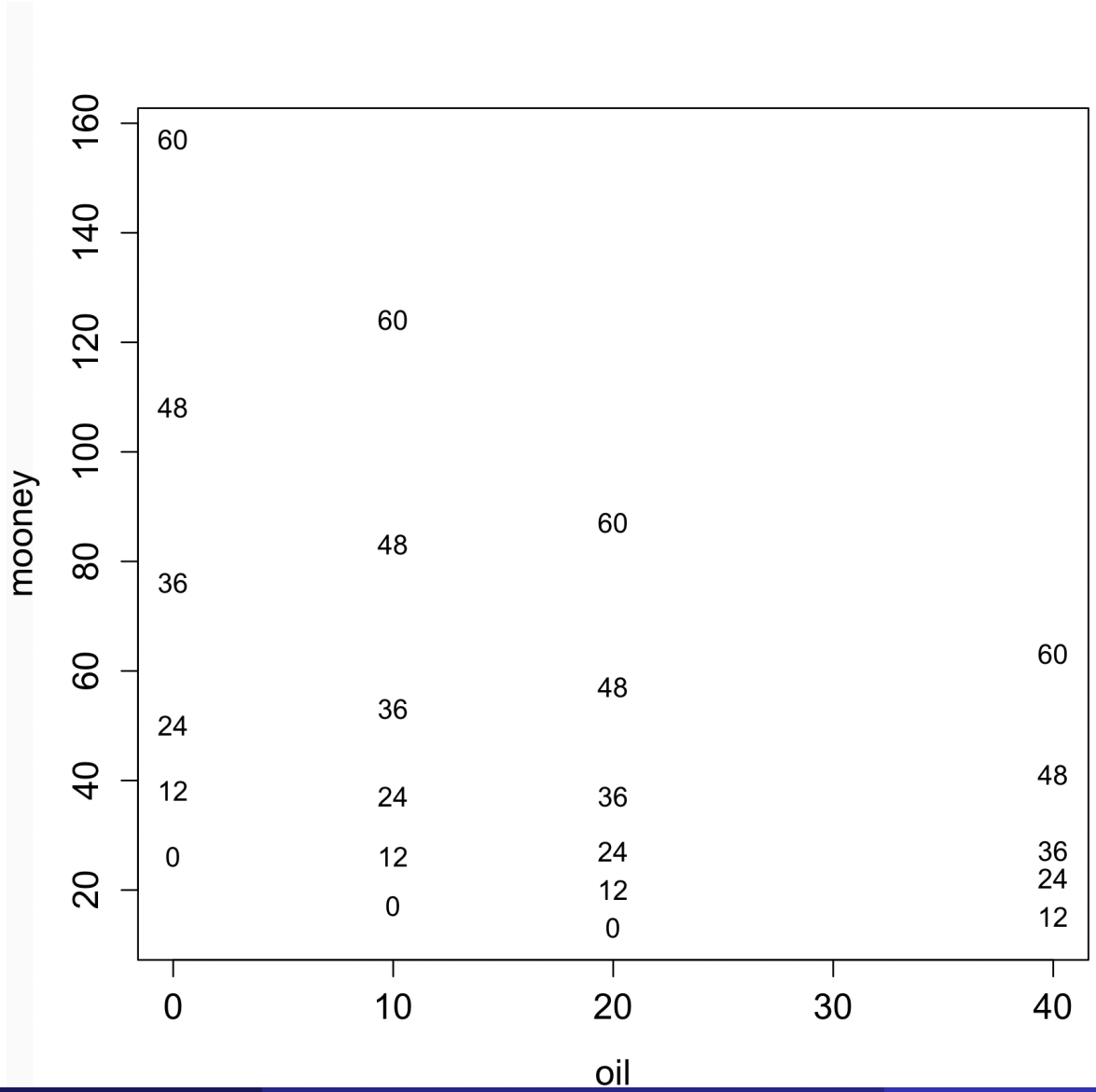
The plots can be generated using ggplot2() using

```
library(ggplot2)
p <- ggplot(mooney, aes(x = oil, y = mooney, label = filler))
p <- p + geom_text()
p <- p + scale_y_continuous(limits = c(0,
max(mooney$mooney, na.rm=TRUE)))
p <- p + labs(title="Mooney data, mooney by oil with
filler labels")
print(p)
## Warning: Removed 1 rows containing missing values (geom text).
library(ggplot2)
p <- ggplot(mooney, aes(x = filler, y = mooney, label = oil))
p <- p + geom_text()
```

Plots like these can also be made in base graphics by plotting an empty plot and then using the `text()` command, which is usually used to annotate graphs:

```
> attach(x)
> plot(oil,mooney,type="n",cex.lab=1.3,cex.axis=1.3)
> text(oil,mooney,as.character(filler))
```

Polynomial regression: cubic fit



From the plots, the relationship between viscosity and both variables appears to be curvilinear. This suggests adding quadratic terms for both variables.

```
> oil2 <- oil^2
> filler2 <- filler^2
> oil.m <- lm(mooney ~ oil+filler+oil2+filler2+oil*filler)
> summary(oil.m)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 27.144582   2.616779  10.373 9.02e-09 ***
oil         -1.271442   0.213533  -5.954 1.57e-05 ***
filler       0.436984   0.152658   2.862   0.0108 *
oil2     0.033611   0.004663   7.208 1.46e-06 ***
filler2  0.027323   0.002410  11.339 2.38e-09 ***
oil:filler  -0.038659   0.003187 -12.131 8.52e-10 ***
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

Multiple R-squared:  0.9917,Adjusted R-squared:  0.9892
F-statistic: 405.2 on 5 and 17 DF,  p-value: < 2.2e-16
```

From the output, the regression equation is

$$\begin{aligned}\text{Mooney} = 27.144 &- 1.271 \times \text{oil} + 0.437 \times \text{filler} \\ &+ 0.034 \times \text{oil}^2 + 0.027 \times \text{filler}^2 - 0.0387 \times \text{oil} \times \text{filler}\end{aligned}$$

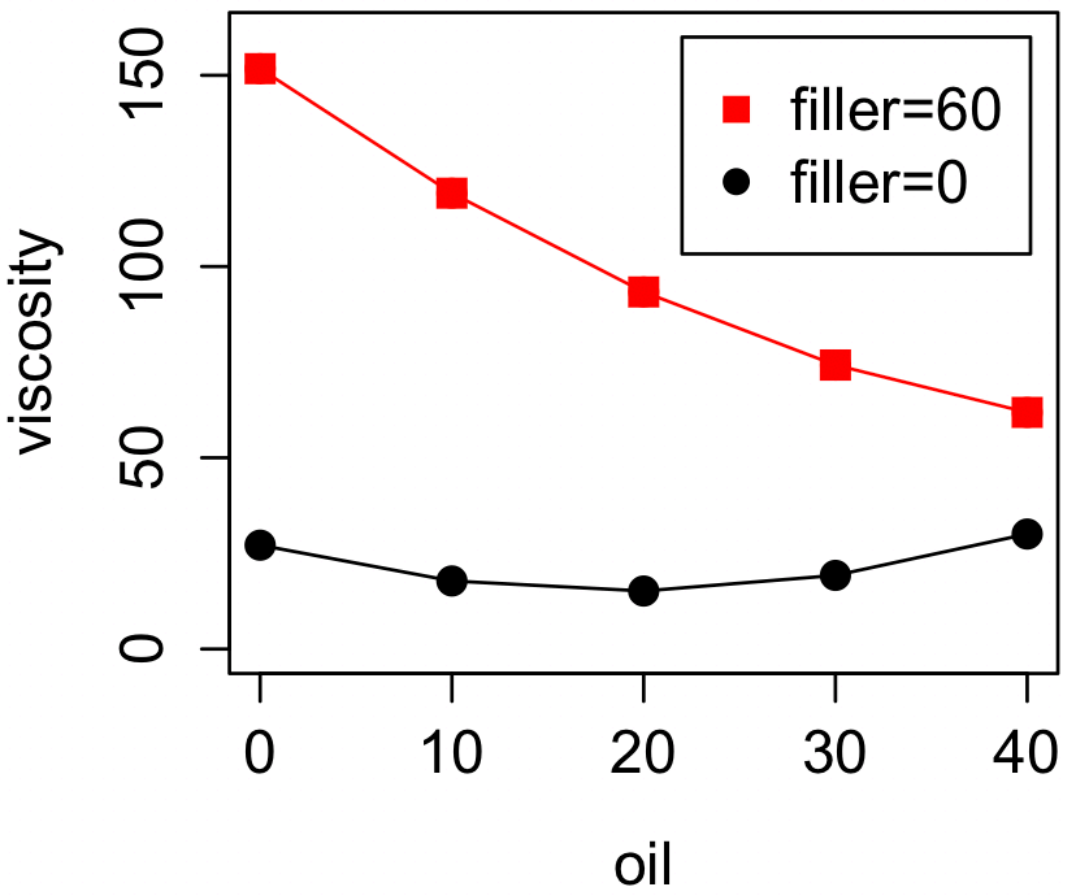
To see how the interaction works, let's predict some values. Instead of copying out the regression equation—which also would tend to lead to some roundoff error—we'll use some built in R functions. In particular, the function `predict` let's you input some fake data, and get predicted values from the model. To set up the fake data, you have to put it into a data frame with the same variable names as the predictor variables in the model.

```
> newoil <- c(0,10,20,30,40)
> newfiller <- c(0,0,0,0,0,60,60,60,60,60)
> newoil <- c(newoil,newoil)
> newoil2 <- newoil^2
> newfiller2 <- newfiller^2
> mydata <- data.frame(cbind(newoil,newfiller,
newoil2,newfiller2))
> names(mydata) <- c("oil","filler","oil2","filler2")
> a <- predict(oil.m,mydata)
> a
       1         2         3         4         5         6
 27.14458  17.79121  15.15996  19.25082  30.06379 151.72512 119.1
       9        10
 74.24519  61.86277
```

```
> plot(mydata$oil[1:5],a[1:5],pch=16,cex=1.5,cex.lab=1.3,cex.axis=
> points(mydata$oil[1:5],a[1:5],type="l")
> points(mydata$oil[1:5],a[6:10],type="l",col="red")
> points(mydata$oil[1:5],a[6:10],pch=16,cex=1.5,
col="red")
> legend(22,160,legend=c("filler=60","filler=0"),
col=c("red","black"),pch=c(16,15),cex=1.3)
```

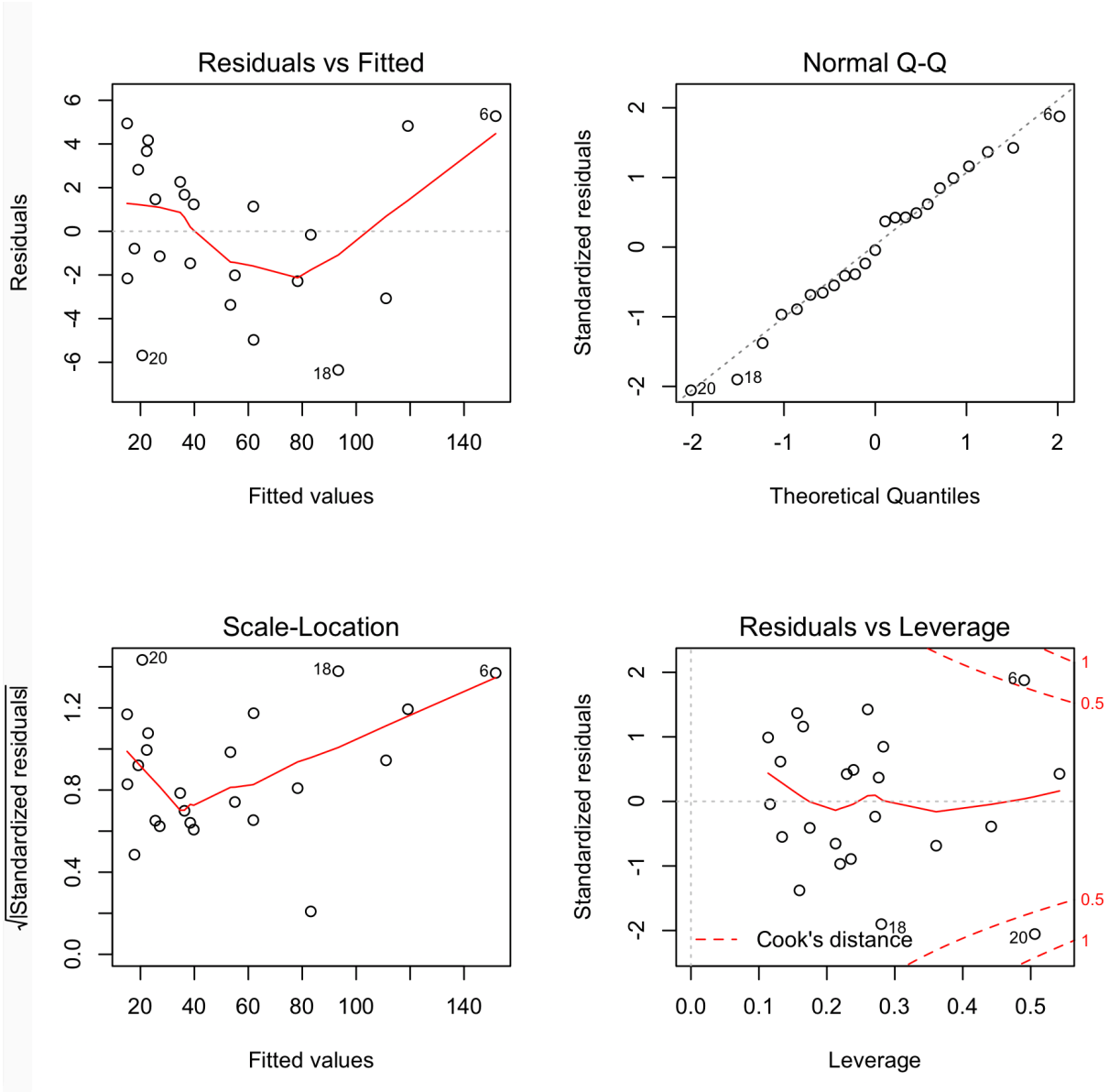
If you forget to match the variable names, here is the error you get (I often forget to match the names):

```
>  mydata <- data.frame(cbind(newoil,newfiller,newoil2,newfiller2)
> a <- predict(oil.m,mydata)
Warning message:
'newdata' had 10 rows but variables found have 24 rows
```


viscosity
0
50
100
150
0
10
20
30
40
oil
filler=60
filler=0

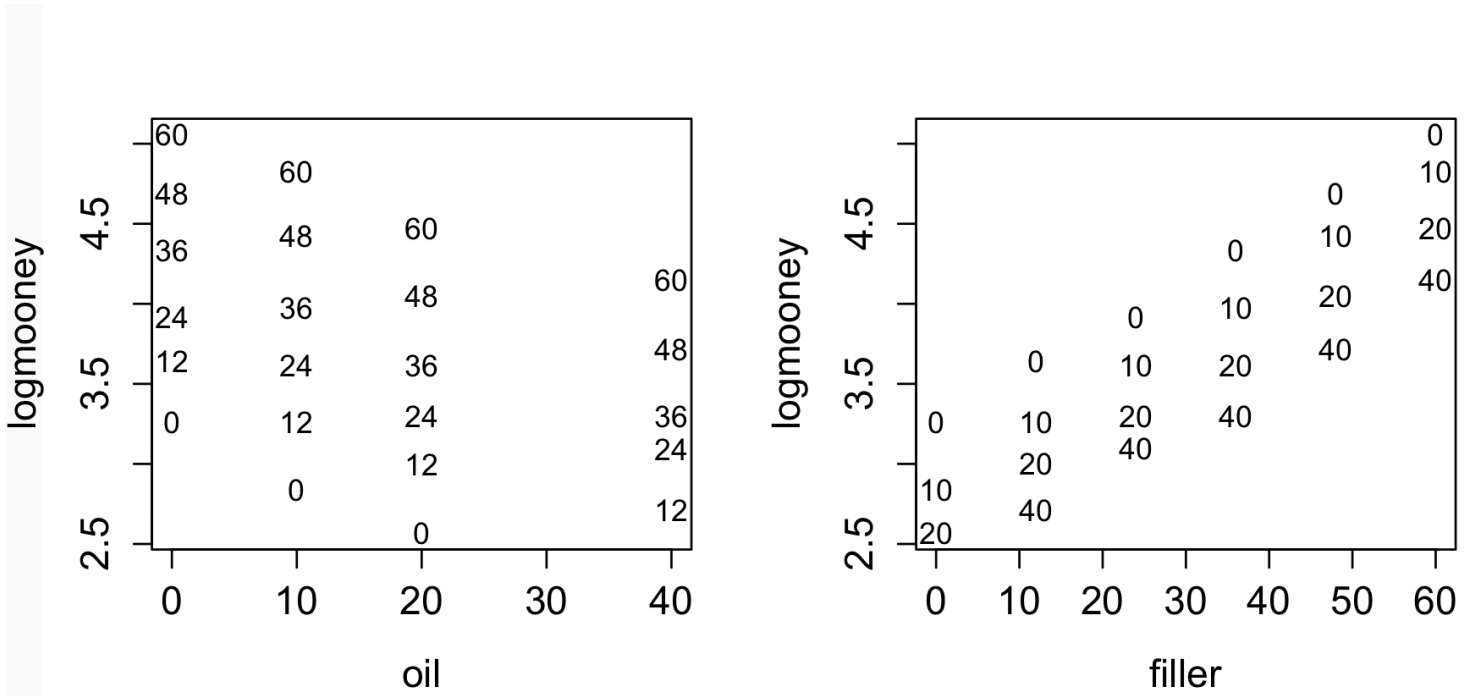
To comment on the output, all second-order terms, including interactions, are highly significant. Also, the R^2 values are extremely high, suggesting that not much else (for example cubic terms) would explain more of the response. The direction of the effects is hard to interpret because the signs change. For example, the effect of oil decreases viscosity in the first order term, but increases for the second order term, and the interaction is also negative, suggesting that as oil level increases, increasing the filler will decrease viscosity more, and vice versa (as filler increases, increasing oil decreases viscosity more.

The plot helps illustrate the idea of the interaction. The relationship between viscosity and oil is quadratic for both levels of filler (I only plotted the two extreme values for the filler), but this quadratic relationship depends on the level of the filler. Similarly, one could make a plot of viscosity versus filler, and find that the quadratic relationship depends on the oil value.


Residuals vs Fitted
Residuals
Fitted values
Normal Q-Q
Standardized residuals
Theoretical Quantiles
Scale-Location
√|Standardized residuals|
Fitted values
Residuals vs Leverage
Standardized residuals
Leverage
Cook's distance

The residual plots look ok. There are potentially a couple of influential observations (points 6 and 20), but this does not seem bad.

Another possibility is to use the log of the Mooney viscosity. In this case, the log viscosity still seems to be quadratically related to oil, but linearly related to filler.


logmooney
2.5
3.5
4.5
0
10
20
30
40
oil
60
48
36
24
12
0
filler
50
60

```
> oil.m2 <- lm(log(mooney) ~ oil + filler + oil2 +
filler2 + oil*filler)
> summary(oil.m2)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.236e+00  3.557e-02  90.970  < 2e-16 ***
oil         -3.921e-02  2.903e-03 -13.507 1.61e-10 ***
filler       2.860e-02  2.075e-03  13.781 1.18e-10 ***
oil2         4.227e-04  6.339e-05   6.668 3.96e-06 ***
filler2      4.657e-05  3.276e-05   1.421    0.173
oil:filler  -4.231e-05  4.332e-05  -0.977    0.342

Multiple R-squared:  0.9954,Adjusted R-squared:  0.9941
```

Here the interaction term isn't significant so we can remove it and refit the model. The quadratic term for filler is also not signifcant (after the interaction is removed), so we can remove that too.

```
> oil.m3 <- lm(log(mooney) ~ oil + filler + oil2 + filler2 )
> oil.m3
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.251e+00  3.202e-02 101.537  < 2e-16 ***
oil         -4.033e-02  2.664e-03 -15.136 1.11e-11 ***
filler       2.838e-02  2.061e-03  13.773 5.32e-11 ***
oil2         4.146e-04  6.277e-05   6.605 3.34e-06 ***
filler2      3.997e-05  3.201e-05   1.248    0.228
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.230e+00  2.734e-02 118.139  < 2e-16 ***
oil         -4.024e-02  2.702e-03 -14.890 6.26e-12 ***
filler       3.086e-02  5.716e-04  53.986  < 2e-16 ***
oil2         4.097e-04  6.356e-05   6.446 3.53e-06 ***
---
Multiple R-squared:  0.9947,Adjusted R-squared:  0.9939
```

Both the full quadratic model and the model with log-transformed responses fit the data very well in terms of R^2 and adjusted R^2. There are pros and cons for the two models. Pros for the log-viscosity model are that there are fewer parameters and that it doesn't have an interaction term, making it easier to interpret.

A pro for the quadratic model is that uses the original measurement scale, which again makes it easier to interpret in another sense, especially if you are using it to make predictions.

To get confidence intervals for effects (for regression coefficients), you can use the `confint()` function. You can either get intervals just for specific variables in the model, or for a list of them.

```
> confint(oil.m,"oil")
        2.5 %     97.5 %
oil -1.721958 -0.8209274
> confint(oil.m,c("oil","filler"))
            2.5 %     97.5 %
oil    -1.7219576 -0.8209274
filler  0.1149033  0.7590651
```

How to get all coefficients automatically?

```
> names(oil.m)
 [1] "coefficients"  "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"        "qr"            "df.residual"
 [9] "na.action"     "xlevels"       "call"          "terms"
[13] "model"
> names(oil.m$effects)
 [1] "(Intercept)" "oil"         "filler"      "oil2"        "fil
 [6] "oil:filler"  ""            ""            ""            ""
[11] ""            ""            ""            ""            ""
[16] ""            ""            ""            ""            ""
[21] ""            ""            ""
```

How to get all coefficients automatically?

```
> confint(oil.m,names(oil.m$effects)[1:6])
> confint(oil.m,names(oil.m$effects)[1:6])
                   2.5 %      97.5 %
(Intercept) 21.62366037 32.66550266
oil         -1.72195764 -0.82092736
filler       0.11490332  0.75906511
oil2         0.02377265  0.04344848
filler2      0.02223872  0.03240655
oil:filler  -0.04538226 -0.03193570
> confint(oil.m,names(oil.m$effects)[1:6],level=.90)
                     5 %        95 %
(Intercept) 22.59241512 31.69674790
oil         -1.64290585 -0.89997915
filler       0.17141878  0.70254965
oil2         0.02549891  0.04172222
filler2      0.02313079  0.03151448
oil:filler  -0.04420253 -0.03311544
```

To see the effect of removing parameters on confidence intervals, we'll compare the widths of the CIs for models with log-response and all quadratic terms, interaction removed, and interaction and filler-squared removed, just on the oil parameter. Here `m4` has the smallest number of parameters, and `m2` has the most. The CI gets slightly wider as we increase the number of parameters.

```
> confint(oil.m4,"oil")[2] - confint(oil.m4,"oil")[1]
[1] 0.01131161
> confint(oil.m3,"oil")[2] - confint(oil.m3,"oil")[1]
[1] 0.0111952
> confint(oil.m2,"oil")[2] - confint(oil.m2,"oil")[1]
[1] 0.01224919
```