SML 201 – Week 10

John D. Storey

Spring 2016

Contents

Two Quantitative Variables

Sample Correlation

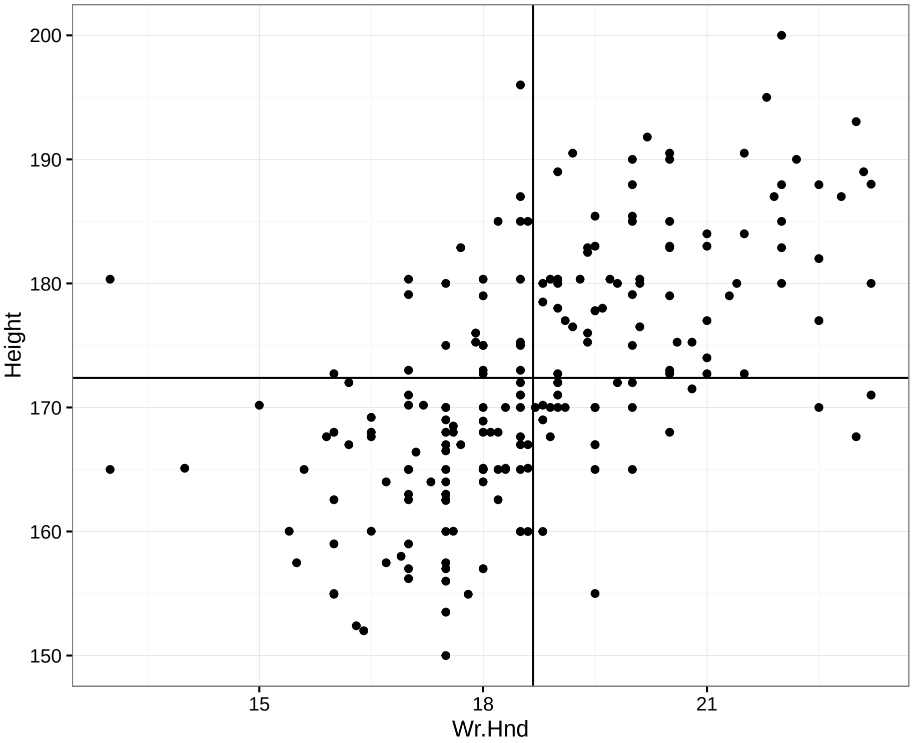
Suppose we observe n pairs of data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$. Their sample correlation is

$$\begin{aligned} r_{xy} &= \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2 \sum_{i=1}^{n}(y_i - \overline{y})^2}} & (1) \\ &= \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{(n-1)s_x s_y} & (2) \end{aligned}$$

where s_x and s_y are the sample standard deviations of each measured variable.

Hand Size Vs. Height

```
> ggplot(data = survey, mapping=aes(x=Wr.Hnd, y=Height)) +
+   geom_point() + geom_vline(xintercept=mean(survey$Wr.Hnd, na.rm=TRUE)) +
+   geom_hline(yintercept=mean(survey$Height, na.rm=TRUE))
```



HT of Correlation

```
> str(cor.test)
function (x, ...)
```

From the help file:

```
Usage

cor.test(x, ...)

## Default S3 method:
cor.test(x, y,
         alternative = c("two.sided", "less", "greater"),
         method = c("pearson", "kendall", "spearman"),
         exact = NULL, conf.level = 0.95, continuity = FALSE,
         ...)

## S3 method for class 'formula'
cor.test(formula, data, subset, na.action, ...)
```

HT of Correlation

```
> cor.test(x=survey$Wr.Hnd, y=survey$Height)

    Pearson's product-moment correlation

data:  survey$Wr.Hnd and survey$Height
t = 10.792, df = 206, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.5063486 0.6813271
sample estimates:
      cor
0.6009909
```

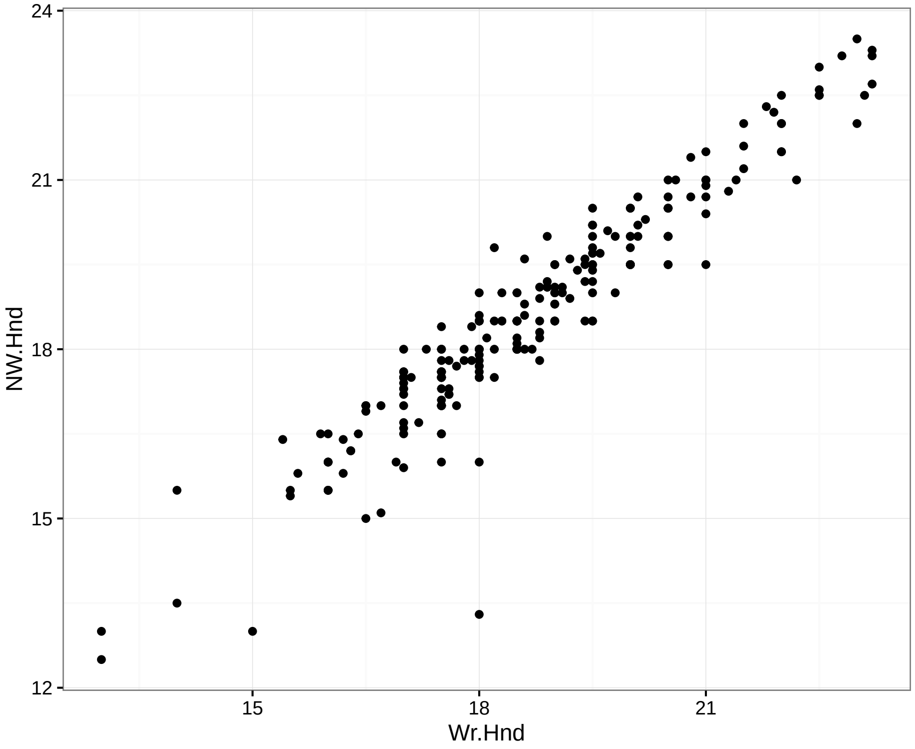
HT By Hand

Compare the following to the above output of `cor.test()`.

```
> r <- cor(survey$Wr.Hnd, survey$Height,
+      use="pairwise.complete.obs")
> df <- sum(complete.cases(survey[,c("Wr.Hnd", "Height")]))-2
> # dplyr way to get df:
> # df <- (survey %>% select(Wr.Hnd, Height) %>%
> #         na.omit() %>% nrow())-2
>
> tstat <- r/sqrt((1 - r^2)/df)
> tstat
[1] 10.79234
>
> pvalue <- 2*pt(q=-abs(tstat), df=df)
> pvalue
[1] 8.227549e-22
```

Hand Sizes

```
> ggplot(data = survey) +
+   geom_point(aes(x=Wr.Hnd, y=NW.Hnd))
```



Correlation of Hand Sizes

```
> cor.test(x=survey$Wr.Hnd, y=survey$NW.Hnd)

    Pearson's product-moment correlation

data:  survey$Wr.Hnd and survey$NW.Hnd
t = 45.712, df = 234, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.9336780 0.9597816
sample estimates:
      cor
0.9483103
```

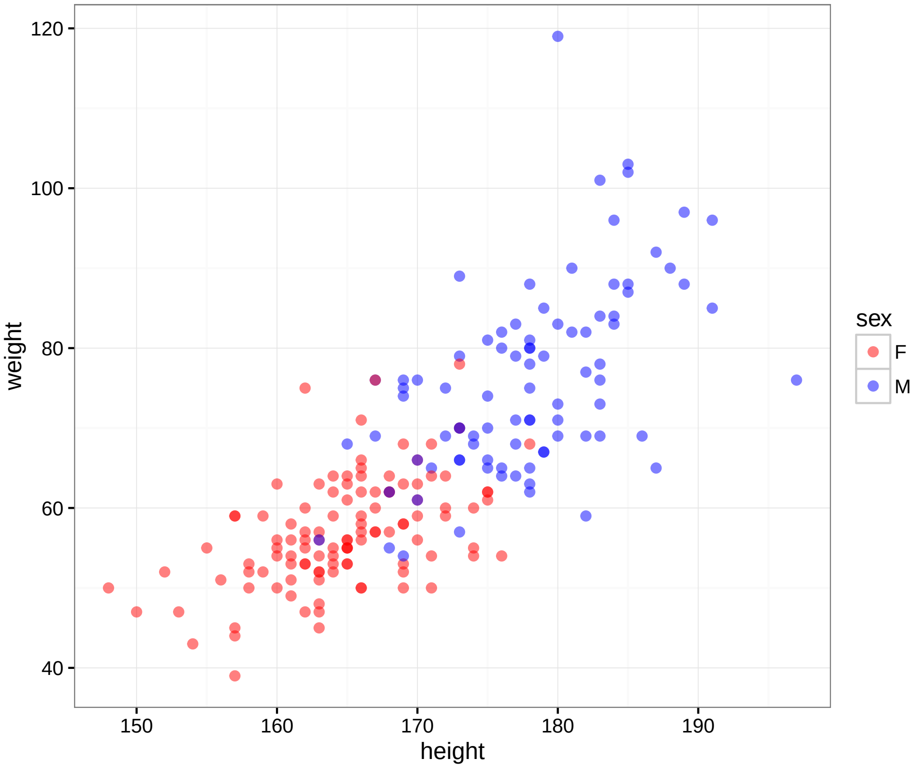
Davis Data

```
> library("car")
> data("Davis", package="car")
```

```
> htwt <- tbl_df(Davis)
> htwt[12,c(2,3)] <- htwt[12,c(3,2)]
> head(htwt)
Source: local data frame [6 x 5]

     sex weight height repwt repht
  (fctr)  (int)  (int) (int) (int)
1      M     77    182    77   180
2      F     58    161    51   159
3      F     53    161    54   158
4      M     68    177    70   175
5      F     59    157    59   155
6      M     76    170    76   165
```

Height and Weight

```
> ggplot(htwt) +
+   geom_point(aes(x=height, y=weight, color=sex), size=2, alpha=0.5) +
+   scale_color_manual(values=c("red", "blue"))
```



Correlation Test

```
> cor.test(x=htwt$height, y=htwt$weight)

    Pearson's product-moment correlation

data:  htwt$height and htwt$weight
t = 17.04, df = 198, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.7080838 0.8218898
sample estimates:
      cor
0.7710743
```

Correlation Test with Outlier

Recall we had to fix an error in the data, which we noticed as an outlier in the scatterplot. Here is the effect of the outlier:

```
> cor.test(x=Davis$height, y=Davis$weight)

    Pearson's product-moment correlation

data:  Davis$height and Davis$weight
t = 2.7179, df = 198, p-value = 0.007152
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.05228435 0.31997151
sample estimates:
      cor
0.1896496
```

Correlation Test with Outlier

Let's use the Spearman rank-based correlation:

```
> cor.test(x=Davis$height, y=Davis$weight, method="spearman")
Warning in cor.test.default(x = Davis$height, y = Davis$weight,
method = "spearman"): Cannot compute exact p-value with ties

    Spearman's rank correlation rho

data:  Davis$height and Davis$weight
S = 308750, p-value < 2.2e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.7684305
```

Correlation Among Females

```
> htwt %>% filter(sex=="F") %>%
+   cor.test(~ height + weight, data = .)

    Pearson's product-moment correlation

data:  height and weight
t = 6.2801, df = 110, p-value = 6.922e-09
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.3627531 0.6384268
sample estimates:
      cor
0.5137293
```

Correlation Among Males

```
> htwt %>% filter(sex=="M") %>%
+   cor.test(~ height + weight, data = .)

    Pearson's product-moment correlation

data:  height and weight
t = 5.9388, df = 86, p-value = 5.922e-08
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.3718488 0.6727460
sample estimates:
      cor
0.5392906
```

Why are the stratified correlations lower?

Least Squares Linear Regression

Rationale

- It is often the case that we would like to build a model that explains the variation of one variable in terms of other variables.
- **Least squares linear regression** is one of the simplest and most useful modeling systems for doing so.
- It is simple to fit, it satisfies some optimality criteria, and it is straightforward to check assumptions on the data so that statistical inference can be performed.

Setup

- Let's start with least squares linear regression of just two variables.
- Suppose that we have observed n pairs of data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.
- **Least squares linear regression** models variation of the **response variable** y in terms of the **explanatory variable** x in the form of $\beta_0 + \beta_1 x$, where β_0 and β_1 are chosen to satisfy a least squares optimization.

Line that Minimizes the Squared Error

The least squares regression line is formed from the value of β_0 and β_1 that minimize:

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \beta_1 x_i\right)^2 .$$

For a given set of data, there is a unique solution to this minimization as long as there are at least two unique values among $x_1, x_2, \ldots, x_n$.

Let $\hat{\beta}_0$ and $\hat{\beta}_1$ be the values that minimize this sum of squares.

Least Squares Solution

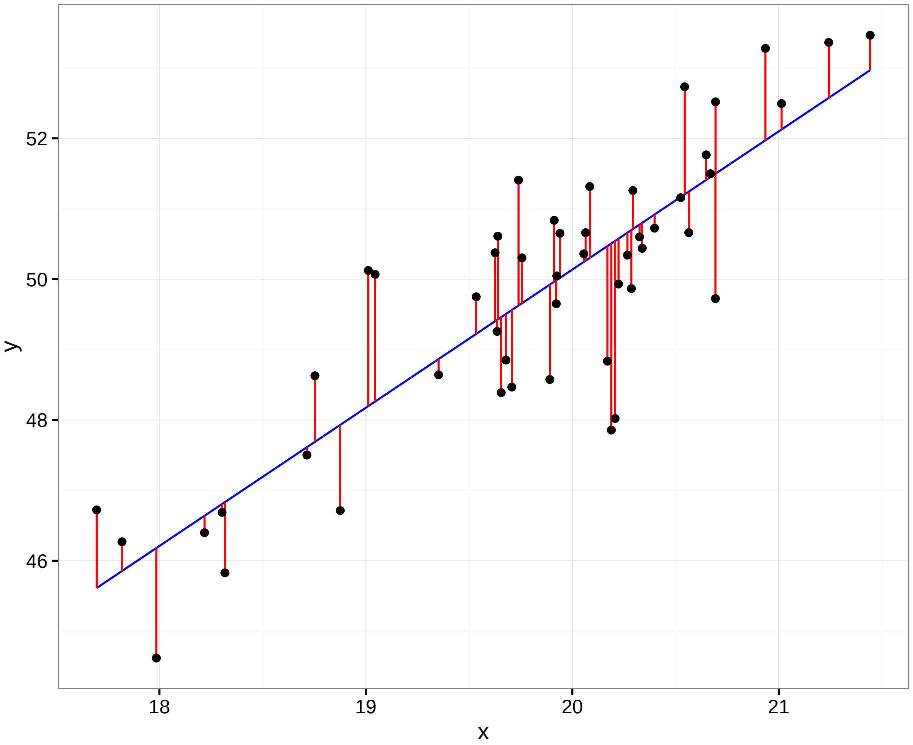
These values are:

$$\hat{\beta}_1 = r_{xy} \frac{s_y}{s_x}$$

$$\hat{\beta}_0 = \overline{y} - \hat{\beta}_1 \overline{x}$$

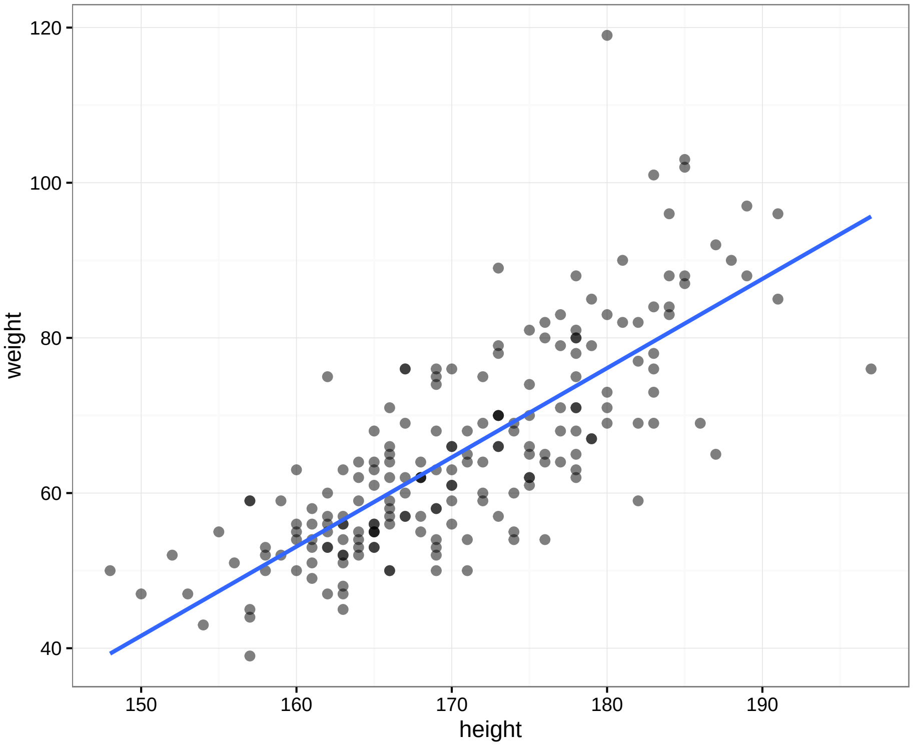
These values have a useful interpretation.

Visualizing Least Squares Line



Example: Height and Weight

```
> ggplot(data=htwt, mapping=aes(x=height, y=weight)) +
+   geom_point(size=2, alpha=0.5) +
+   geom_smooth(method="lm", se=FALSE, formula=y~x)
```

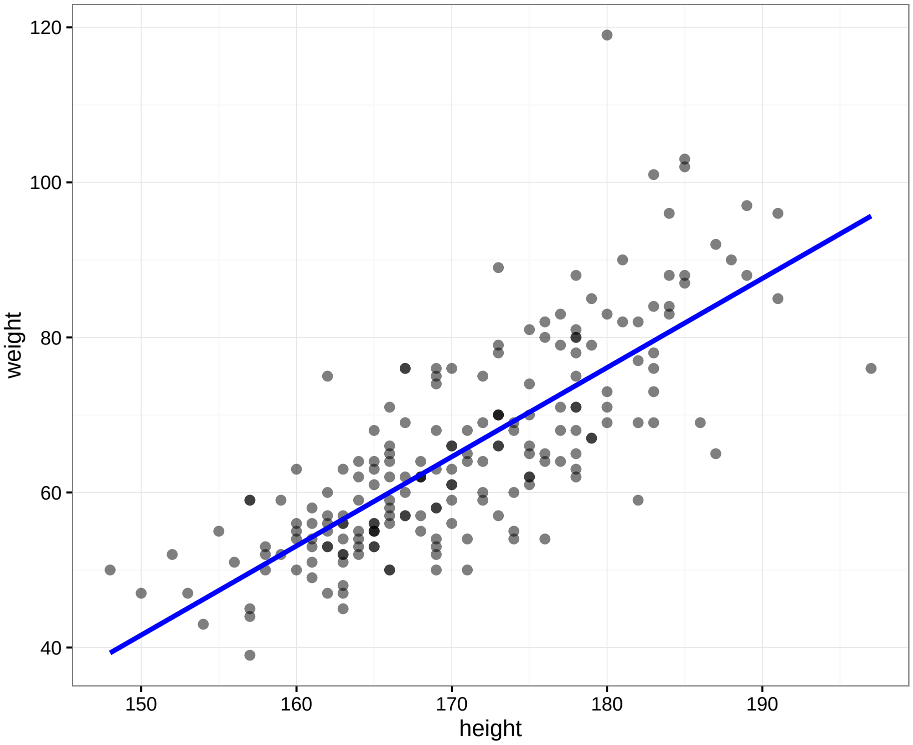


Calculate the Line Directly

```
> beta1 <- cor(htwt$height, htwt$weight) *
+                 sd(htwt$weight) / sd(htwt$height)
> beta1
[1] 1.150092
>
> beta0 <- mean(htwt$weight) - beta1 * mean(htwt$height)
> beta0
[1] -130.9104
>
> yhat <- beta0 + beta1 * htwt$height
```

Plot the Line

```
> df <- data.frame(htwt, yhat=yhat)
> ggplot(data=df) + geom_point(aes(x=height, y=weight), size=2, alpha=0.5) +
+   geom_line(aes(x=height, y=yhat), color="blue", size=1.2)
```



Calculate the Line in R

The syntax for a model in R is

`response variable ~ explanatory variables`

where the `explanatory variables` component can involve several types of terms.

```
> myfit <- lm(weight ~ height, data=htwt)
> myfit

Call:
lm(formula = weight ~ height, data = htwt)

Coefficients:
(Intercept)       height
    -130.91         1.15
```

What's Next?

- Why minimize the sum of squares?
- What is the output provided by R?
- How do we access and interpret this output from R?
- What assumptions are required to use this machinery?
- How do we check these assumptions on data?
- How can we build more complex models?

`lm` Object Class

An `lm` Object is a List

```
> class(myfit)
[1] "lm"
> is.list(myfit)
[1] TRUE
> names(myfit)
 [1] "coefficients"  "residuals"     "effects"
 [4] "rank"          "fitted.values" "assign"
 [7] "qr"            "df.residual"   "xlevels"
[10] "call"          "terms"         "model"
```

From the R Help

> `lm` returns an object of class "lm" or for multiple responses of class c("mlm", "lm").
>
> The functions `summary` and `anova` are used to obtain and print a summary and analysis of variance table of the results. The generic accessor functions coefficients, effects, fitted.values and residuals extract various useful features of the value returned by `lm`.

Some of the List Items

These are some useful items to access from the `lm` object:

- `coefficients`: a named vector of coefficients
- `residuals`: the residuals, that is response minus fitted values.
- `fitted.values`: the fitted mean values.
- `df.residual`: the residual degrees of freedom.
- `call`: the matched call.
- `model`: if requested (the default), the model frame used.

summary()

```
> summary(myfit)

Call:
lm(formula = weight ~ height, data = htwt)

Residuals:
    Min      1Q  Median      3Q     Max 
-19.658  -5.381  -0.555   4.807  42.894 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -130.91040   11.52792  -11.36   <2e-16 ***
height         1.15009    0.06749   17.04   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8.505 on 198 degrees of freedom
Multiple R-squared:  0.5946,    Adjusted R-squared:  0.5925 
F-statistic: 290.4 on 1 and 198 DF,  p-value: < 2.2e-16
```

summary() List Elements

```
> mysummary <- summary(myfit)
> names(mysummary)
 [1] "call"          "terms"         "residuals"    
 [4] "coefficients"  "aliased"       "sigma"        
 [7] "df"            "r.squared"     "adj.r.squared"
[10] "fstatistic"    "cov.unscaled" 
```

Using tidy()

```
> library(broom)
> tidy(myfit)
         term    estimate   std.error statistic      p.value
1 (Intercept) -130.910400 11.52792138 -11.35594 2.438012e-23
2      height    1.150092  0.06749465  17.03975 1.121241e-40
```

More on the Underlying Model

Probability Model

The typical probability model assumed for **ordinary least squares** is:

$Y_i = \beta_0 + \beta_1 X_i + E_i$

where $\text{E}[E_i] = 0$, $\text{Var}[E_i] = \sigma^2$, and $\rho_{E_i, E_j} = 0$ for all $i, j \in \{1, 2, \ldots, n\}$.

Optimality

Fitting this linear model by least squares satisfies two types of optimality:

1. Gauss-Markov Theorem
2. Maximum likelihood estimate when in addition $E_i \sim \text{Normal}(0, \sigma^2)$

Interested students can explore the above links.

Observed Data, Fits, and Residuals

We observe data $(x_1, y_1), \ldots, (x_n, y_n)$. Note that we only observe the left hand side of the generative model $y_i = \beta_0 + \beta_1 x_i + e_i$.

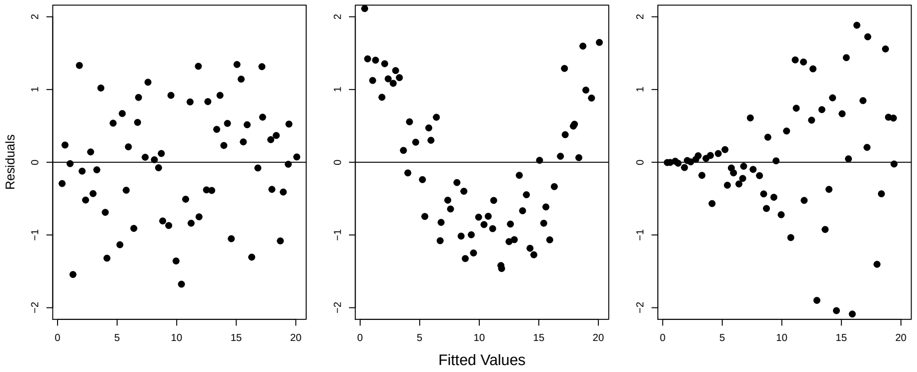
We calculate fitted values and observed residuals:

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

$$\hat{e}_i = y_i - \hat{y}_i$$

By construction, it is the case that $\sum_{i=1}^n \hat{e}_i = 0$.

Fitted Values Vs. Obs. Residuals



Assumptions to Verify

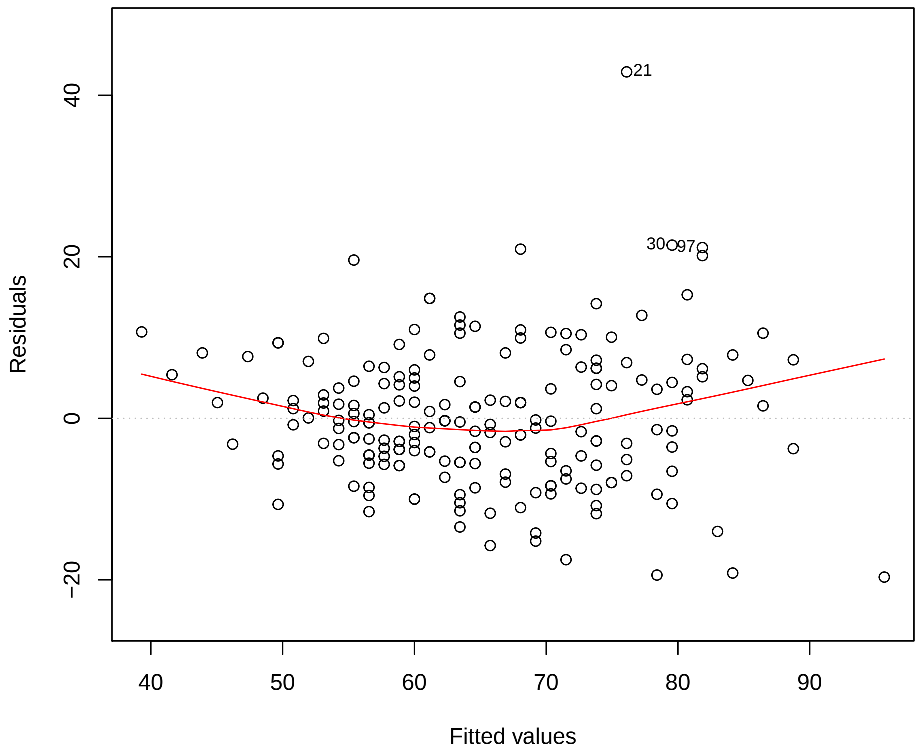
The assumptions on the above linear model are really about the joint distribution of the residuals, which are not directly observed. On data, we try to verify:

1. The fitted values and the residuals show no trends with respect to each other
2. The residuals are distributed approximately Normal$(0, \sigma^2)$
    - A constant variance is called **homoscedasticity**
    - A non-constant variance is called **heteroscedascity**
3. There are no lurking variables

There are two plots we will use in this course to investigate the first two.

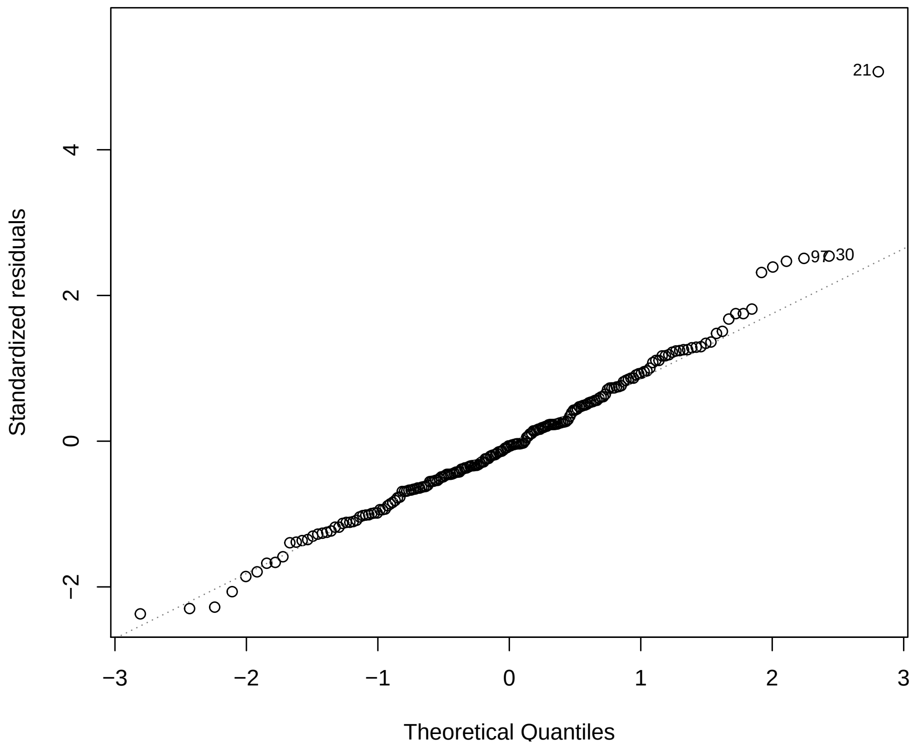
Residual Distribution

```
> plot(myfit, which=1)
```



Normal Residuals Check

```
> plot(myfit, which=2)
```



Proportion of Variation Explained

The proportion of variance explained by the fitted model is called R^2 or r^2. It is calculated by:

$$r^2 = \frac{s_{\hat{y}}^2}{s_y^2}$$

```
> summary(myfit)$r.squared
[1] 0.5945555
>
> var(myfit$fitted.values)/var(htwt$weight)
[1] 0.5945555
```

Categorial Explanatory Variables

Example: Chicken Weights

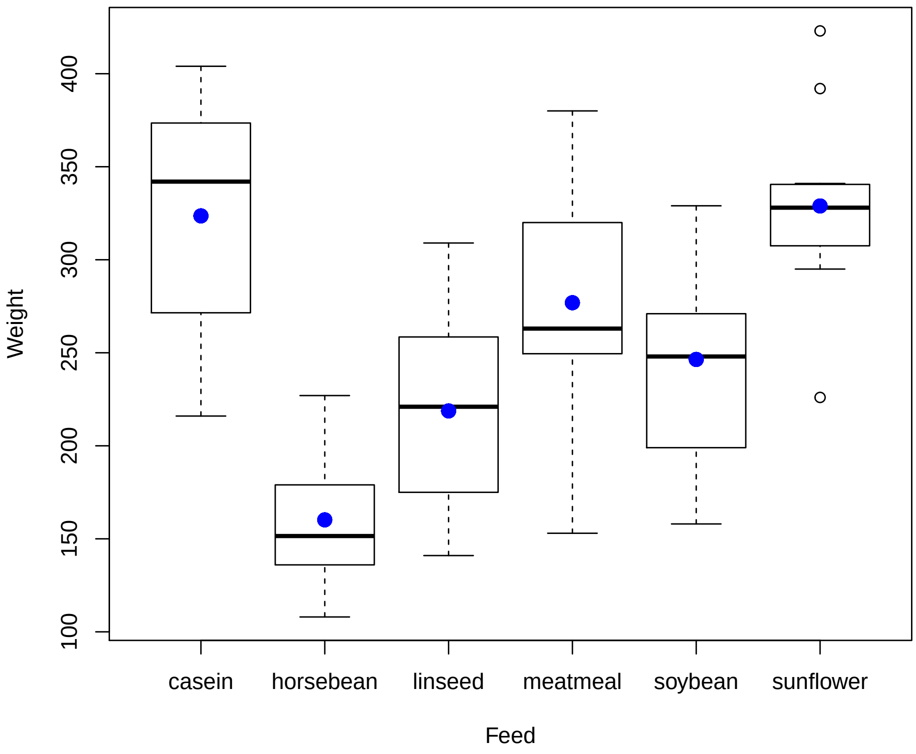
```
> data("chickwts", package="datasets")
> head(chickwts)
  weight      feed
1    179 horsebean
2    160 horsebean
3    136 horsebean
4    227 horsebean
5    217 horsebean
6    168 horsebean
> summary(chickwts$feed)
   casein horsebean   linseed  meatmeal   soybean sunflower
       12        10        12        11        14        12
```

Including Factor Variables in `lm()`

```
> chick_fit <- lm(weight ~ feed, data=chickwts)
> summary(chick_fit)

Call:
lm(formula = weight ~ feed, data = chickwts)

Residuals:
     Min       1Q   Median       3Q      Max
-123.909  -34.413    1.571   38.170  103.091

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    323.583     15.834  20.436  < 2e-16 ***
feedhorsebean -163.383     23.485  -6.957 2.07e-09 ***
feedlinseed   -104.833     22.393  -4.682 1.49e-05 ***
feedmeatmeal   -46.674     22.896  -2.039 0.045567 *
feedsoybean    -77.155     21.578  -3.576 0.000665 ***
feedsunflower    5.333     22.393   0.238 0.812495
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 54.85 on 65 degrees of freedom
Multiple R-squared:  0.5417,    Adjusted R-squared:  0.5064
F-statistic: 15.36 on 5 and 65 DF,  p-value: 5.936e-10
```

Plot the Fit

```
> plot(chickwts$feed, chickwts$weight, xlab="Feed", ylab="Weight")
> points(chickwts$feed, chick_fit$fitted.values, col="blue", pch=20, cex=2)
```



ANOVA (Version 1)

ANOVA (*analysis of variance*) was originally developed as a statistical model and method for comparing differences in mean values between various groups.

ANOVA quantifies and tests for differences in response variables with respect to factor variables.

In doing so, it also partitions the total variance to that due to within and between groups, where groups are defined by the factor variables.

`anova()`

The classic ANOVA table:

```
> anova(chick_fit)
Analysis of Variance Table

Response: weight
          Df Sum Sq Mean Sq F value    Pr(>F)    
feed       5 231129   46226  15.365 5.936e-10 ***
Residuals 65 195556    3009                      
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
> n <- length(chick_fit$residuals) # n <- 71
> (n-1)*var(chick_fit$fitted.values)
[1] 231129.2
> (n-1)*var(chick_fit$residuals)
[1] 195556
> (n-1)*var(chickwts$weight) # sum of above two quantities
[1] 426685.2
> (231129/5)/(195556/65) # F-statistic
[1] 15.36479
```

How It Works

```
> levels(chickwts$feed)
[1] "casein"    "horsebean" "linseed"   "meatmeal"  "soybean"  
[6] "sunflower"
> head(chickwts, n=3)
  weight      feed
1    179 horsebean
2    160 horsebean
3    136 horsebean
> tail(chickwts, n=3)
   weight   feed
69    222 casein
70    283 casein
71    332 casein
> x <- model.matrix(weight ~ feed, data=chickwts)
> dim(x)
[1] 71  6
```

Top of Design Matrix

```
> head(x)
  (Intercept) feedhorsebean feedlinseed feedmeatmeal
1           1             1           0            0
2           1             1           0            0
3           1             1           0            0
4           1             1           0            0
5           1             1           0            0
6           1             1           0            0
  feedsoybean feedsunflower
1           0             0
2           0             0
3           0             0
4           0             0
5           0             0
6           0             0
```

Bottom of Design Matrix

```
> tail(x)
   (Intercept) feedhorsebean feedlinseed feedmeatmeal
66           1             0           0            0
67           1             0           0            0
68           1             0           0            0
69           1             0           0            0
70           1             0           0            0
71           1             0           0            0
   feedsoybean feedsunflower
66           0             0
67           0             0
68           0             0
69           0             0
70           0             0
71           0             0
```

Model Fits

```
> chick_fit$fitted.values %>% round(digits=4) %>% unique()
[1] 160.2000 218.7500 246.4286 328.9167 276.9091 323.5833
```

```
> chickwts %>% group_by(feed) %>% summarize(mean(weight))
Source: local data frame [6 x 2]

       feed mean(weight)
     (fctr)        (dbl)
1    casein     323.5833
2 horsebean     160.2000
3   linseed     218.7500
4  meatmeal     276.9091
5   soybean     246.4286
6 sunflower     328.9167
```

Regression with Several Variables

Weight Regressed on Height + Sex

```
> summary(lm(weight ~ height + sex, data=htwt))

Call:
lm(formula = weight ~ height + sex, data = htwt)

Residuals:
    Min      1Q  Median      3Q     Max
-20.131  -4.884  -0.640   5.160  41.490

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -76.6167    15.7150  -4.875 2.23e-06 ***
height        0.8105     0.0953   8.506 4.50e-15 ***
sexM          8.2269     1.7105   4.810 3.00e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8.066 on 197 degrees of freedom
Multiple R-squared:  0.6372,    Adjusted R-squared:  0.6335
F-statistic:   173 on 2 and 197 DF,  p-value: < 2.2e-16
```

Ordinary Least Squares

Suppose we observe data $(x_{11}, x_{21}, \ldots, x_{d1}, y_1), \ldots, (x_{1n}, x_{2n}, \ldots, x_{dn}, y_n)$. We have a response variable y_i and d explanatory variables $(x_{1i}, x_{2i}, \ldots, x_{di})$ per unit of observation.

Ordinary least squares models the variation of y in terms of $\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_d x_d$.

OLS Solution

The estimates of $\beta_0, \beta_1, \ldots, \beta_d$ are found by identifying the values that minimize:

$$\sum_{i=1}^{n} \left[y_i - (\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \ldots + \beta_d x_{di})\right]^2$$

The solutions are expressed in terms of matrix algebra computations (see, e.g., here).

OLS in R

R implements OLS of multiple explanatory variables exactly the same as with a single explanatory variable, except we need to show the sum of all explanatory variables that we want to use.

```
> lm(weight ~ height + sex, data=htwt)

Call:
lm(formula = weight ~ height + sex, data = htwt)

Coefficients:
(Intercept)       height         sexM
   -76.6167       0.8106       8.2269
```

A Twist on OLS

We can include a single variable but on two different scales:

```
> htwt <- htwt %>% mutate(height2 = height^2)
> summary(lm(weight ~ height + height2, data=htwt))

Call:
lm(formula = weight ~ height + height2, data = htwt)

Residuals:
    Min      1Q  Median      3Q     Max
-24.265  -5.159  -0.499   4.549  42.965

Coefficients:
```

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 107.117140 175.246872   0.611    0.542
height       -1.632719   2.045524  -0.798    0.426
height2       0.008111   0.005959   1.361    0.175

Residual standard error: 8.486 on 197 degrees of freedom
Multiple R-squared:  0.5983,    Adjusted R-squared:  0.5943
F-statistic: 146.7 on 2 and 197 DF,  p-value: < 2.2e-16
```

Interactions

It is possible to include products of explanatory variables, which is called an *interaction*.

```
> summary(lm(weight ~ height + sex + height:sex, data=htwt))

Call:
lm(formula = weight ~ height + sex + height:sex, data = htwt)

Residuals:
    Min      1Q  Median      3Q     Max
-20.869  -4.835  -0.897   4.429  41.122

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -45.6730    22.1342  -2.063   0.0404 *
height        0.6227     0.1343   4.637 6.46e-06 ***
sexM        -55.6571    32.4597  -1.715   0.0880 .
height:sexM   0.3729     0.1892   1.971   0.0502 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8.007 on 196 degrees of freedom
Multiple R-squared:  0.6442,    Adjusted R-squared:  0.6388
F-statistic: 118.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

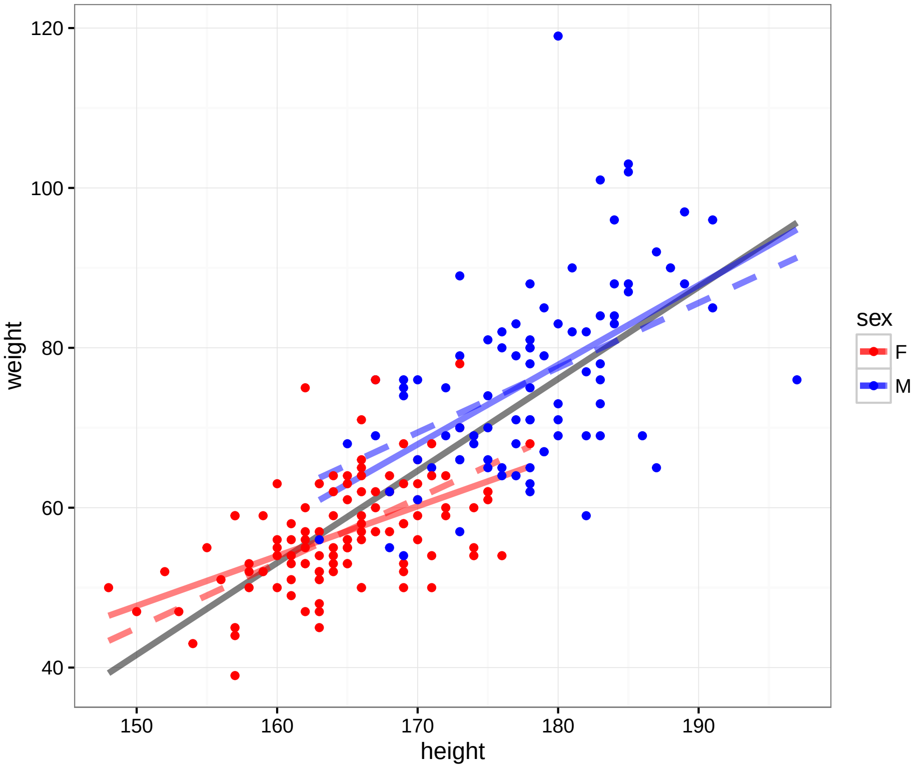
More on Interactions

What happens when there is an interaction between a quantitative explanatory variable and a factor explanatory variable? In the next plot, we show three models:

- Grey solid: `lm(weight ~ height, data=htwt)`

- Color dashed: `lm(weight ~ height + sex, data=htwt)`
- Color solid: `lm(weight ~ height + sex + height:sex, data=htwt)`

Visualizing Three Different Models



Comparing Linear Models

Example: Davis Data

Suppose we are considering the three following models:

```
> f1 <- lm(weight ~ height, data=htwt)
> f2 <- lm(weight ~ height + sex, data=htwt)
> f3 <- lm(weight ~ height + sex + height:sex, data=htwt)
```

How do we determine if the additional terms in models `f2` and `f3` are needed?

ANOVA (Version 2)

A generalization of ANOVA exists that allows us to compare two nested models, quantifying their differences in terms of goodness of fit and performing a hypothesis test of whether this difference is statistically significant.

A model is *nested* within another model if their difference is simply the absence of certain terms in the smaller model.

The null hypothesis is that the additional terms have coefficients equal to zero, and the alternative hypothesis is that at least one coefficient is nonzero.

Both versions of ANOVA can be described in a single, elegant mathematical framework.

Comparing Two Models with `anova()`

This provides a comparison of the improvement in fit from model `f2` compared to model `f1`:

```
> anova(f1, f2)
Analysis of Variance Table

Model 1: weight ~ height
Model 2: weight ~ height + sex
  Res.Df   RSS Df Sum of Sq      F    Pr(>F)
1    198 14321
2    197 12816  1    1504.9 23.133 2.999e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

When There's a Single Variable Difference

Compare above `anova(f1, f2)` p-value to that for the `sex` term from the `f2` model:

```
> tidy(f2)
         term    estimate   std.error statistic      p.value
1 (Intercept) -76.6167326 15.71504644 -4.875374 2.231334e-06
2      height   0.8105526  0.09529565  8.505662 4.499241e-15
3        sexM   8.2268893  1.71050385  4.809629 2.998988e-06
```

Calculating the F-statistic

```
> anova(f1, f2)
Analysis of Variance Table

Model 1: weight ~ height
Model 2: weight ~ height + sex
  Res.Df   RSS Df Sum of Sq      F    Pr(>F)
1    198 14321
2    197 12816  1    1504.9 23.133 2.999e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

How the F-statistic is calculated:

```
> n <- nrow(htwt)
> ss1 <- (n-1)*var(f1$residuals)
> ss1
[1] 14321.11
> ss2 <- (n-1)*var(f2$residuals)
> ss2
[1] 12816.18
> ((ss1 - ss2)/anova(f1, f2)$Df[2])/(ss2/f2$df.residual)
[1] 23.13253
```

ANOVA on More Distant Models

We can compare models with multiple differences in terms:

```
> anova(f1, f3)
Analysis of Variance Table

Model 1: weight ~ height
Model 2: weight ~ height + sex + height:sex
  Res.Df   RSS Df Sum of Sq      F    Pr(>F)
1    198 14321
2    196 12567  2      1754 13.678 2.751e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Compare Multiple Models at Once

We can compare multiple models at once:

```
> anova(f1, f2, f3)
Analysis of Variance Table

Model 1: weight ~ height
Model 2: weight ~ height + sex
Model 3: weight ~ height + sex + height:sex
  Res.Df   RSS Df Sum of Sq       F    Pr(>F)    
1    198 14321                                   
2    197 12816  1   1504.93 23.4712 2.571e-06 ***
3    196 12567  1    249.04  3.8841   0.05015 .  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Variable Transformations

Rationale

In order to obtain reliable model fits and inference on linear models, the model assumptions described earlier must be satisfied.

Sometimes it is necessary to *transform* the response variable and/or some of the explanatory variables.

This process should involve data visualization and exploration.

Power and Log Transformations

It is often useful to explore power and log transforms of the variables, e.g., $\log(y)$ or y^λ for some λ (and likewise $\log(x)$ or x^λ).

You can read more about the Box-Cox family of power transformations.

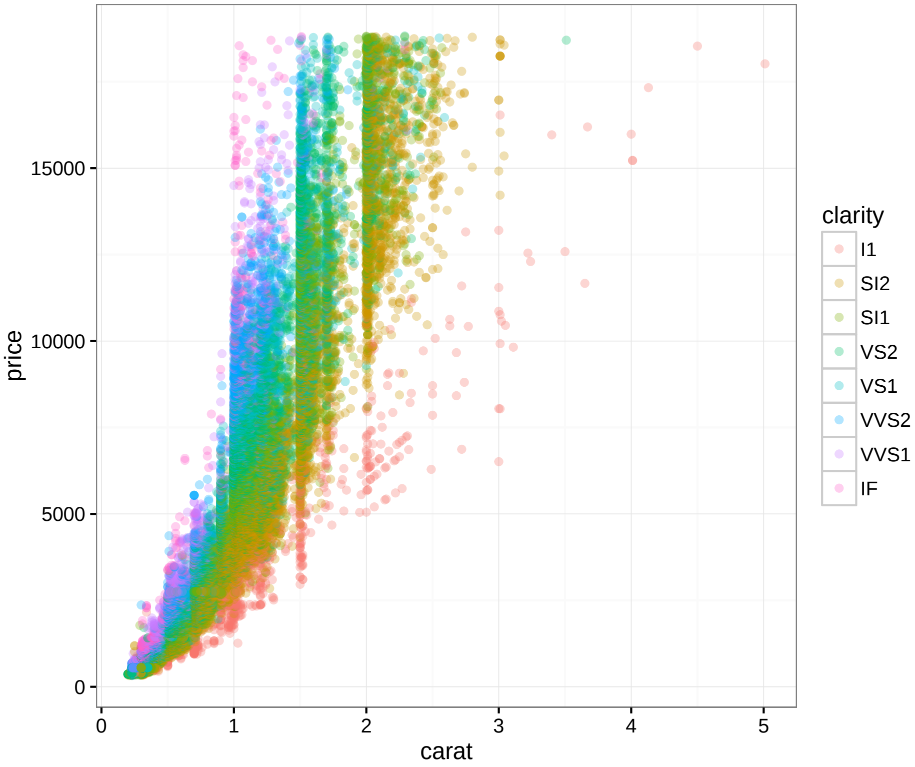
Diamonds Data

```
> data("diamonds", package="ggplot2")
> head(diamonds)
Source: local data frame [6 x 10]

  carat       cut  color clarity depth table price     x     y
  (dbl)    (fctr) (fctr)  (fctr) (dbl) (dbl) (int) (dbl) (dbl)
1  0.23     Ideal      E     SI2  61.5    55   326  3.95  3.98
2  0.21   Premium      E     SI1  59.8    61   326  3.89  3.84
```

```
3  0.23      Good      E     VS1  56.9    65   327  4.05  4.07
4  0.29   Premium      I     VS2  62.4    58   334  4.20  4.23
5  0.31      Good      J     SI2  63.3    58   335  4.34  4.35
6  0.24 Very Good      J    VVS2  62.8    57   336  3.94  3.96
Variables not shown: z (dbl)
```

Nonlinear Relationship

```
> ggplot(data = diamonds) +
+   geom_point(mapping=aes(x=carat, y=price, color=clarity), alpha=0.3)
```



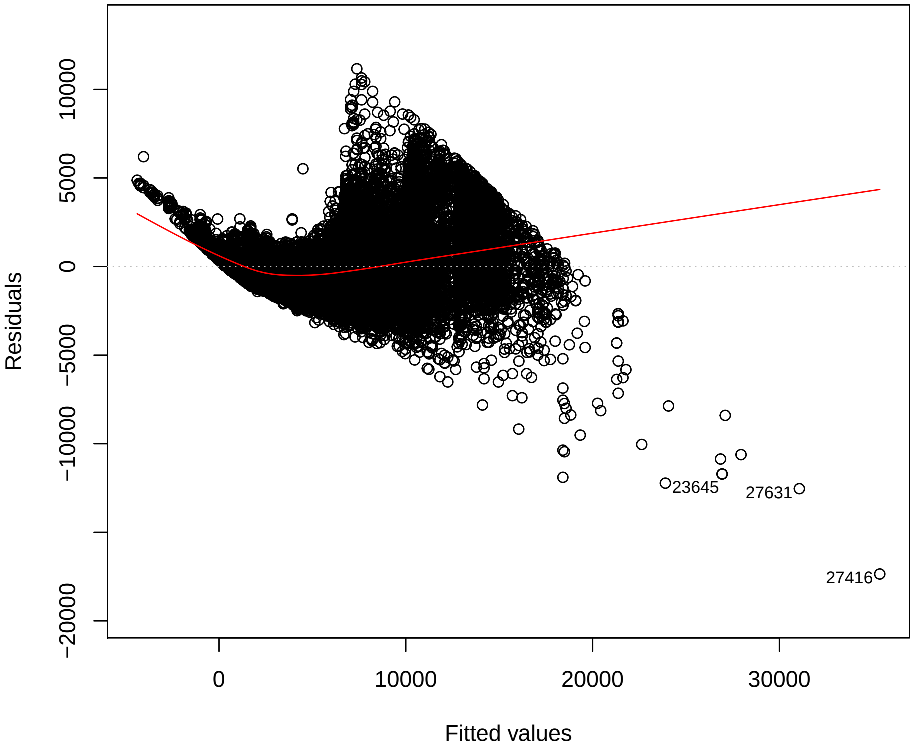
Regression with Nonlinear Relationship

```
> diam_fit <- lm(price ~ carat + clarity, data=diamonds)
> anova(diam_fit)
Analysis of Variance Table
```

```
Response: price
             Df     Sum Sq    Mean Sq  F value    Pr(>F)
carat         1 7.2913e+11 7.2913e+11 435639.9 < 2.2e-16 ***
clarity       7 3.9082e+10 5.5831e+09   3335.8 < 2.2e-16 ***
Residuals 53931 9.0264e+10 1.6737e+06
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

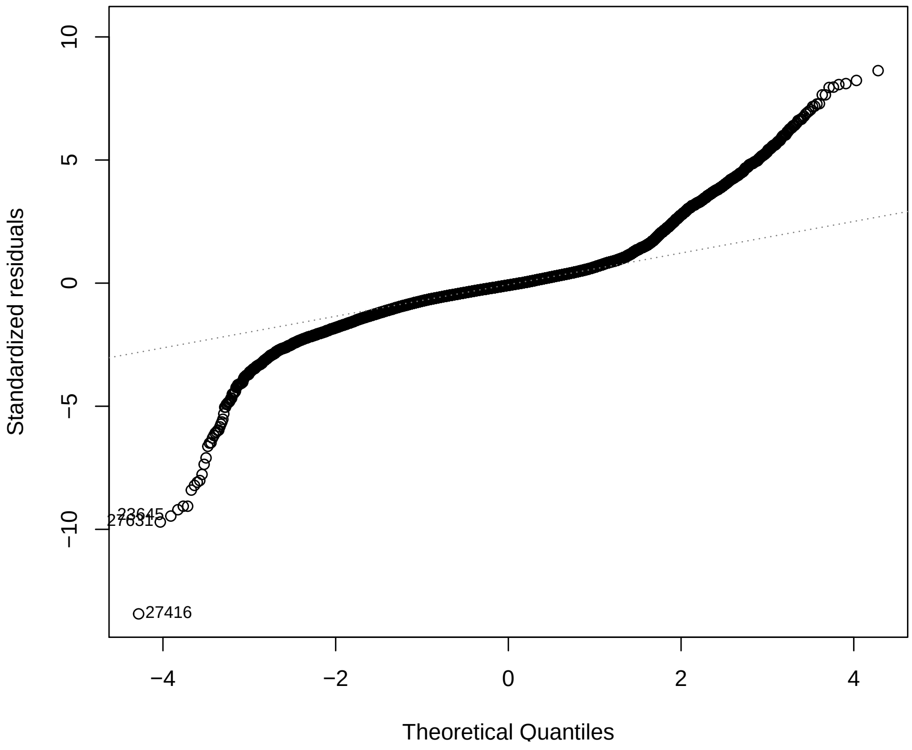
Residual Distribution

```
> plot(diam_fit, which=1)
```



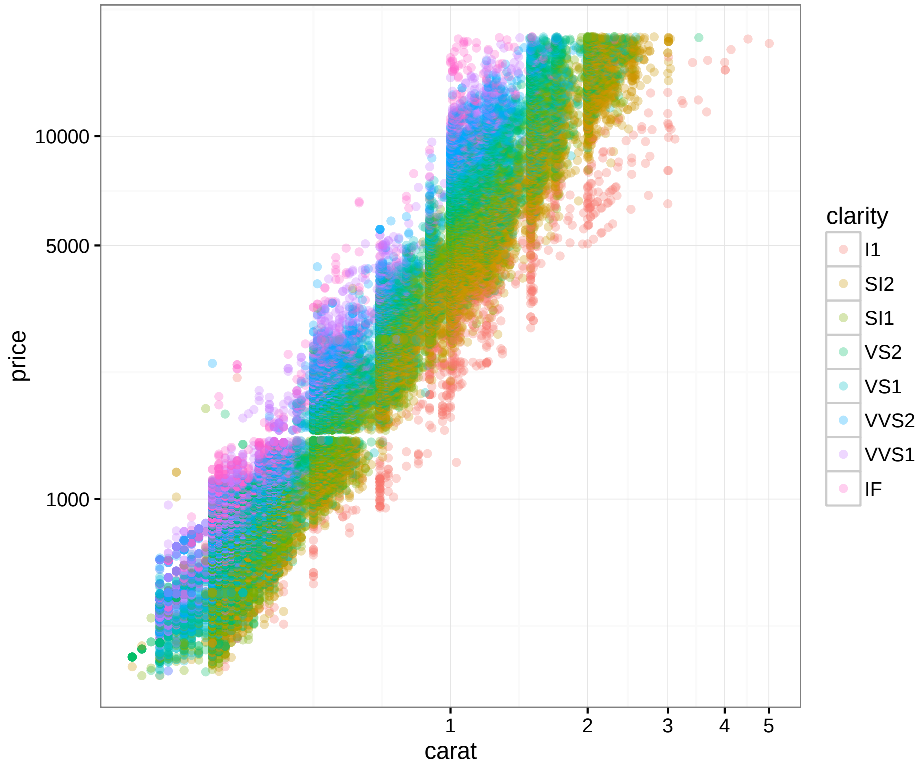
Normal Residuals Check

```
> plot(diam_fit, which=2)
```



Log-Transformation

```
> ggplot(data = diamonds) +
+   geom_point(aes(x=carat, y=price, color=clarity), alpha=0.3) +
+   scale_y_log10(breaks=c(1000,5000,10000)) +
+   scale_x_log10(breaks=1:5)
```

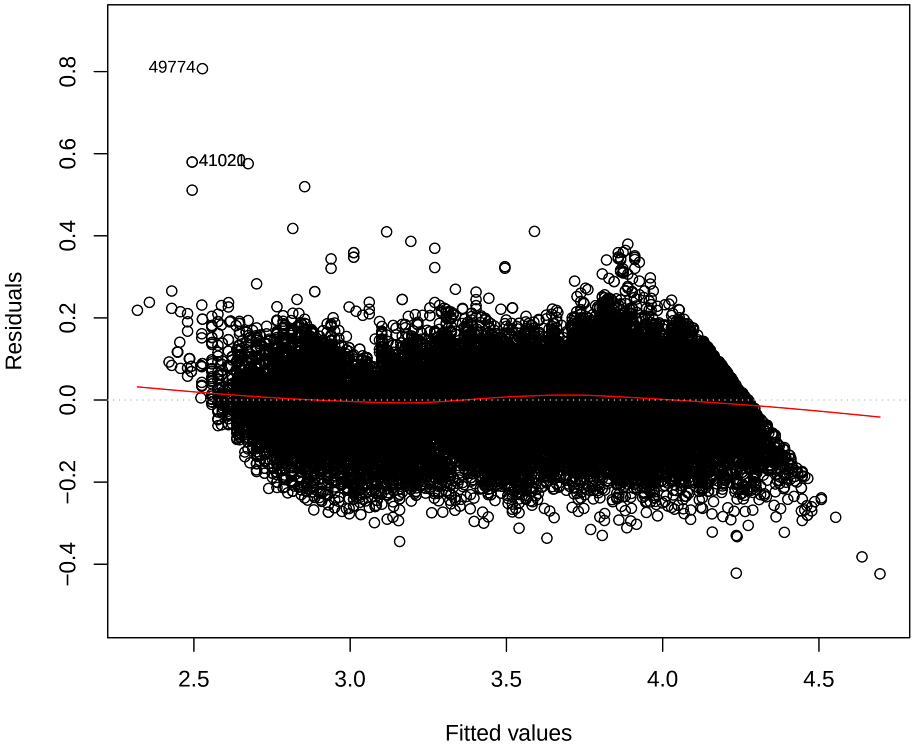


Regression on Log-Transformed Data

```
> diamonds <- mutate(diamonds, log_price = log(price, base=10),
+                    log_carat = log(carat, base=10))
> ldiam_fit <- lm(log_price ~ log_carat + clarity, data=diamonds)
> anova(ldiam_fit)
Analysis of Variance Table

Response: log_price
             Df Sum Sq Mean Sq   F value    Pr(>F)
log_carat     1 9771.9  9771.9 1452922.6 < 2.2e-16 ***
clarity       7  339.1    48.4    7203.3 < 2.2e-16 ***
Residuals 53931  362.7     0.0
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

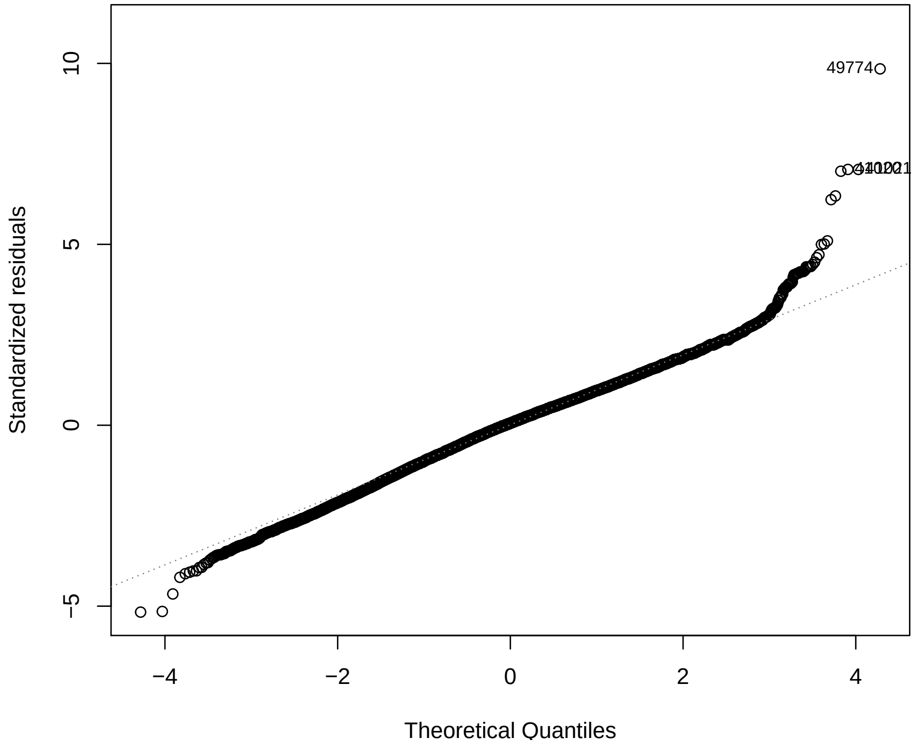
Residual Distribution

```
> plot(ldiam_fit, which=1)
```



Normal Residuals Check

```
> plot(ldiam_fit, which=2)
```



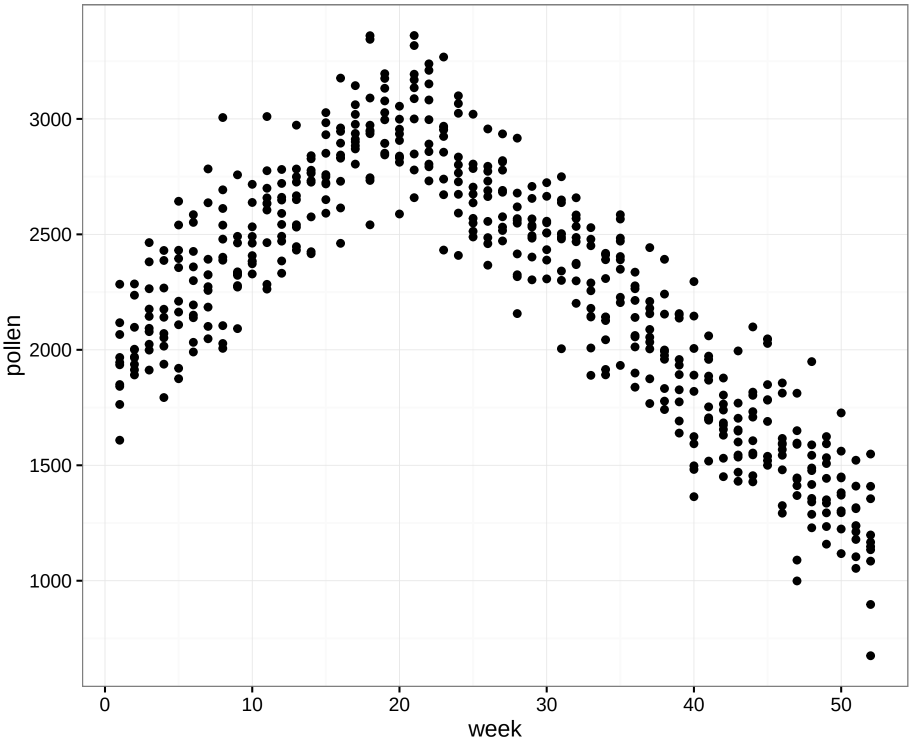
Tree Pollen Study

Suppose that we have a study where tree pollen measurements are averaged every week, and these data are recorded for 10 years.

```
> pollen_study
Source: local data frame [520 x 3]

    week  year   pollen
   (int) (int)    (dbl)
1      1  2001 1841.751
2      2  2001 1965.503
3      3  2001 2380.972
4      4  2001 2141.025
5      5  2001 2210.473
6      6  2001 2585.321
7      7  2001 2392.183
8      8  2001 2104.680
9      9  2001 2278.014
10    10  2001 2383.945
..   ...   ...      ...
```

Tree Pollen Count by Week

```
> ggplot(pollen_study) + geom_point(aes(x=week, y=pollen))
```



A Clever Transformation

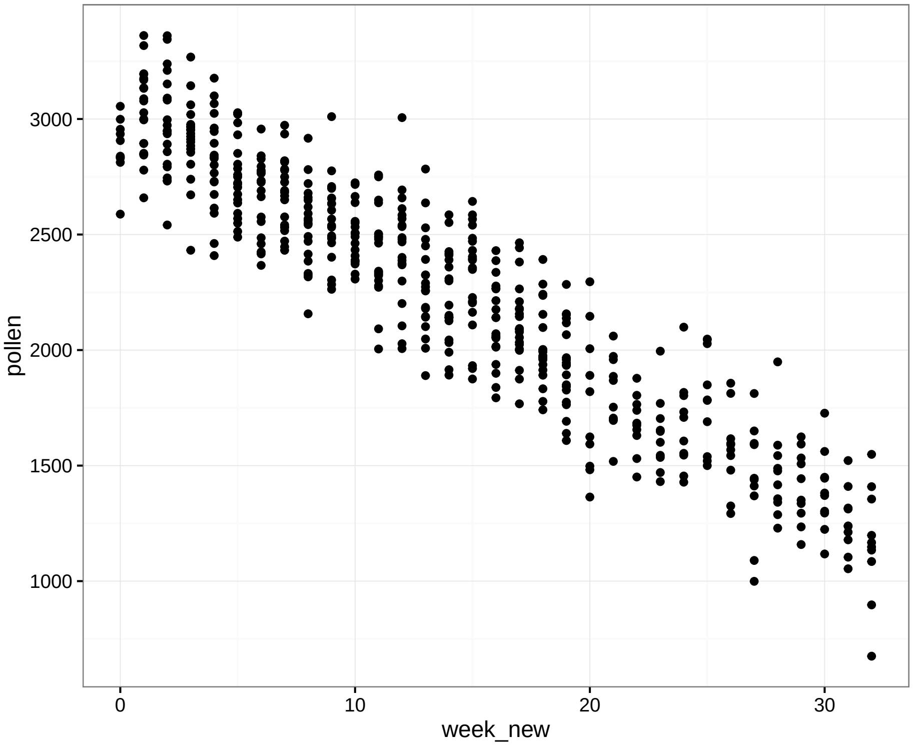
We can see there is a linear relationship between `pollen` and `week` if we transform `week` to be number of weeks from the peak week.

```
> pollen_study <- pollen_study %>%
+                         mutate(week_new = abs(week-20))
```

Note that this is a very different transformation from taking a log or power transformation.

week Transformed

```
> ggplot(pollen_study) + geom_point(aes(x=week_new, y=pollen))
```



Extras

License

https://github.com/SML201/lectures/blob/master/LICENSE.md

Source Code

https://github.com/SML201/lectures/tree/master/week10

Session Information

```
> sessionInfo()
R version 3.2.3 (2015-12-10)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: OS X 10.11.3 (El Capitan)

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods
[7] base

other attached packages:
[1] car_2.1-1       MASS_7.3-45     broom_0.4.0
[4] dplyr_0.4.3     ggplot2_2.1.0   knitr_1.12.3
[7] magrittr_1.5    devtools_1.10.0

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.4        formatR_1.3        nloptr_1.0.4
 [4] plyr_1.8.3         tools_3.2.3        digest_0.6.9
 [7] lme4_1.1-11        evaluate_0.8.3     memoise_1.0.0
[10] nlme_3.1-125       gtable_0.2.0       lattice_0.20-33
[13] mgcv_1.8-11        Matrix_1.2-3       psych_1.5.8
[16] DBI_0.3.1          yaml_2.1.13        parallel_3.2.3
[19] SparseM_1.7        stringr_1.0.0      MatrixModels_0.4-1
[22] grid_3.2.3         nnet_7.3-12        R6_2.1.2
[25] rmarkdown_0.9.5.9  minqa_1.2.4        reshape2_1.4.1
[28] tidyr_0.4.1        scales_0.4.0       htmltools_0.3.5
[31] splines_3.2.3      assertthat_0.1     pbkrtest_0.4-6
[34] mnormt_1.5-3       colorspace_1.2-6   quantreg_5.21
[37] labeling_0.3       stringi_1.0-1      lazyeval_0.1.10
[40] munsell_0.4.3
```