QCB 508 – Week 8

John D. Storey

Spring 2020

Contents

Principal Component Analysis

Multivariate Distributions

Let $\boldsymbol{X} = (X_1, X_2, \ldots, X_m)^T$ be a vector of m rv's. We also let realized values be $\boldsymbol{x} = (x_1, x_2, \ldots, x_m)^T$. The joint pmf or pdf is written as

$$f(\boldsymbol{x}) = f(x_1, x_2, \ldots, x_m)$$

and if the rv's are independent then

$$f(\boldsymbol{x}) = \prod_{i=1}^{m} f(x_i).$$

MV Expected Value

The expected value of $\boldsymbol{X} = (X_1, X_2, \ldots, X_m)^T$ is an m-vector:

$$\mathrm{E}[\boldsymbol{X}] = \begin{bmatrix} \mathrm{E}[X_1] \\ \mathrm{E}[X_2] \\ \vdots \\ \mathrm{E}[X_m] \end{bmatrix}$$

MV Variance-Covariance Matrix

The variance-covariance matrix of $\boldsymbol{X}$ is an $m \times m$ matrix with (i, j) entry equal to $\mathrm{Cov}(X_i, X_j)$.

$$\mathrm{Var}(\boldsymbol{X}) = \begin{bmatrix} \mathrm{Var}(X_1) & \mathrm{Cov}(X_1, X_2) & \cdots & \mathrm{Cov}(X_1, X_m) \\ \mathrm{Cov}(X_2, X_1) & \mathrm{Var}(X_2) & \cdots & \vdots \\ \vdots & & \ddots & \vdots \\ \mathrm{Cov}(X_m, X_1) & \cdots & & \mathrm{Var}(X_m) \end{bmatrix}$$

Multivariate Normal

The m-vector $\boldsymbol{X}$ has Multivariate Normal distribution when $\boldsymbol{X} \sim \mathrm{MVN}_m(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\mu}$ is the m-vector of population means and $\boldsymbol{\Sigma}$ is the $m \times m$ variance-covariance matrix. Its pdf is

$$f(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{2\pi|\boldsymbol{\Sigma}|}} \exp\left\{ -\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \right\}.$$

Fun fact: $\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{X} - \boldsymbol{\mu}) \sim \mathrm{MVN}_m(\boldsymbol{0}, \boldsymbol{I})$.

Goal of PCA

For a given set of variables, **principal component analysis** (PCA) finds (constrained) weighted sums of the variables to produce variables (called principal components) that capture consectuive maximum levels of variation in the data.

Specifically, the first principal component is the weighted sum of the variables that results in a component with the highest variation.

This component is then "removed" from the data, and the second principal component is obtained on the resulting residuals.

This process is repeated until there is no variation left in the data.

Population PCA

Suppose we have m random variables $X_1, X_2, \ldots, X_m$. We wish to identify a set of weights $w_1, w_2, \ldots, w_m$ that maximizes

$$\mathrm{Var}\left(w_1 X_1 + w_2 X_2 + \cdots + w_m X_m\right).$$

However, this is unbounded, so we need to constrain the weights. It turns out that constraining the weights so that

$$\|\boldsymbol{w}\|_2^2 = \sum_{i=1}^{m} w_i^2 = 1$$

is both interpretable and mathematically tractable.

Therefore we wish to maximize

$$\text{Var}\left(w_1 X_1 + w_2 X_2 + \cdots + w_m X_m\right)$$

subject to $\|\boldsymbol{w}\|_2^2 = 1$. Let $\boldsymbol{\Sigma}$ be the $m \times m$ population variance-covariance matrix of the random variables $X_1, X_2, \ldots, X_m$. It follows that

$$\text{Var}\left(w_1 X_1 + w_2 X_2 + \cdots + w_m X_m\right) = \boldsymbol{w}^T \boldsymbol{\Sigma} \boldsymbol{w}.$$

Using a Lagrange multiplier, we wish to maximize

$$\boldsymbol{w}^T \boldsymbol{\Sigma} \boldsymbol{w} + \lambda(\boldsymbol{w}^T \boldsymbol{w} - 1).$$

Differentiating with respect to $\boldsymbol{w}$ and setting to $\mathbf{0}$, we get $\boldsymbol{\Sigma}\boldsymbol{w} - \lambda \boldsymbol{w} = 0$ or

$$\boldsymbol{\Sigma}\boldsymbol{w} = \lambda \boldsymbol{w}.$$

For any such $\boldsymbol{w}$ and λ where this holds, note that

$$\text{Var}\left(w_1 X_1 + w_2 X_2 + \cdots + w_m X_m\right) = \boldsymbol{w}^T \boldsymbol{\Sigma} \boldsymbol{w} = \lambda$$

so the variance is λ.

The eigendecompositon of a matrix identifies all such solutions to $\boldsymbol{\Sigma}\boldsymbol{w} = \lambda \boldsymbol{w}$. Specifically, it calculates the decompositon

$$\boldsymbol{\Sigma} = \boldsymbol{W} \boldsymbol{\Lambda} \boldsymbol{W}^T$$

where $\boldsymbol{W}$ is an $m \times m$ orthogonal matrix and $\boldsymbol{\Lambda}$ is a diagonal matrix with entries $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m \geq 0$.

The fact that $\boldsymbol{W}$ is orthogonal means $\boldsymbol{W}\boldsymbol{W}^T = \boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}$.

The following therefore hold:

- For each column j of $\boldsymbol{W}$, say $\boldsymbol{w}_j$, it follows that $\boldsymbol{\Sigma}\boldsymbol{w}_j = \lambda_j \boldsymbol{w}_j$
- $\|\boldsymbol{w}_j\|_2^2 = 1$ and $\boldsymbol{w}_j^T \boldsymbol{w}_k = \mathbf{0}$ for $\lambda_j \neq \lambda_k$
- $\text{Var}(\boldsymbol{w}_j^T \boldsymbol{X}) = \lambda_j$
- $\text{Var}(\boldsymbol{w}_1^T \boldsymbol{X}) \geq \text{Var}(\boldsymbol{w}_2^T \boldsymbol{X}) \geq \cdots \geq \text{Var}(\boldsymbol{w}_m^T \boldsymbol{X})$
- $\boldsymbol{\Sigma} = \sum_{j=1}^{m} \lambda_j \boldsymbol{w}_j \boldsymbol{w}_j^T$
- For $\lambda_j \neq \lambda_k$,

$$\text{Cov}(\boldsymbol{w}_j^T \boldsymbol{X}, \boldsymbol{w}_k^T \boldsymbol{X}) = \boldsymbol{w}_j^T \boldsymbol{\Sigma} \boldsymbol{w}_k = \lambda_k \boldsymbol{w}_j^T \boldsymbol{w}_k = \mathbf{0}$$

Population PCs

The jth **population principal component** (PC) of $X_1, X_2, \ldots, X_m$ is

$$\boldsymbol{w}_j^T \boldsymbol{X} = w_{1j}X_1 + w_{2j}X_2 + \cdots + w_{mj}X_m$$

where $\boldsymbol{w}_j = (w_{1j}, w_{2j}, \ldots, w_{mj})^T$ is column j of $\boldsymbol{W}$ from the eigendecomposition

$$\boldsymbol{\Sigma} = \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^T.$$

The column $\boldsymbol{w}_j$ are called the **loadings** of the jth principal component.

Population Variance Explained

The **variance explained** by the jth PC is λ_j, which is diagonal element j of $\boldsymbol{\Lambda}$.

The **proportion of variance explained** (PVE) by the jth PC is

$$\frac{\lambda_j}{\sum_{k=1}^m \lambda_k}.$$

Sample PCA

Suppose we have m variables, each with n observations:

$$\begin{aligned}
\boldsymbol{x}_1 &= (x_{11}, x_{12}, \ldots, x_{1n}) \\
\boldsymbol{x}_2 &= (x_{21}, x_{22}, \ldots, x_{2n}) \\
&\vdots \\
\boldsymbol{x}_m &= (x_{m1}, x_{m2}, \ldots, x_{mn})
\end{aligned}$$

We can organize these variables into an $m \times n$ matrix $\boldsymbol{X}$ where row i is $\boldsymbol{x}_i$.

PCA can be extended from the population scenario applied to rv's to the sample scenario applied to the observed data $\boldsymbol{X}$.

Consider all possible weighted sums of these variables

$$\tilde{\boldsymbol{x}} = \sum_{i=1}^m u_i \boldsymbol{x_i}$$

where we constrain $\sum_{i=1}^m u_i^2 = 1$.

The first principal component of $\boldsymbol{X}$ is the results $\tilde{\boldsymbol{x}}$ with maximum sample variance

$$s_{\tilde{\boldsymbol{x}}}^2 = \frac{\sum_{j=1}^n \left(\tilde{x}_j - \frac{1}{n}\sum_{k=1}^n \tilde{x}_k\right)^2}{n-1}.$$

This first sample principal component (PC) is then "removed" from the data, and the procedure is repeated until $\min(m, n-1)$ sample PCs are constructed.

The sample PCs are found in a manner analogous to the population PCs. First, we construct the $m \times m$ sample covariance matrix $\boldsymbol{S}$ with (i, j) entry

$$s_{ij} = \frac{\sum_{k=1}^{n}(x_{ik} - \bar{x}_{i\cdot})(x_{jk} - \bar{x}_{j\cdot})}{n-1}.$$

Identifying $\boldsymbol{u}$ that maximizes $s^2_{\tilde{\boldsymbol{x}}}$ also maximizes

$$\boldsymbol{u}^T \boldsymbol{S} \boldsymbol{u}.$$

Following the steps from before, we want to identify $\boldsymbol{u}$ and λ where

$$\boldsymbol{S}\boldsymbol{u} = \lambda \boldsymbol{u}.$$

which is accomplished with the eigendecomposition

$$\boldsymbol{S} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$$

where again $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}$ and $\boldsymbol{\Lambda}$ is a diagonal matrix so that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m \geq 0$.

Sample PCs

Let $x^*_{ij} = x_{ij} - \bar{x}_{i\cdot}$ be the row-wise mean-centered values of $\boldsymbol{X}$, and let $\boldsymbol{X}^*$ be the matrix composed of these values. Also, let $\boldsymbol{u}_j$ be column j of $\boldsymbol{U}$ from $\boldsymbol{S} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$.

Sample PC j is then

$$\tilde{\boldsymbol{x}}_j = \boldsymbol{u}_j^T \boldsymbol{X}^* = \sum_{i=1}^{m} u_{ij} \boldsymbol{x}^*_i$$

for $j = 1, 2, \ldots, \min(m, n-1)$.

The loadings corresponding to PC j are $\boldsymbol{u}_j$.

Note that the mean of PC j is zero, i.e., that

$$\frac{1}{n}\sum_{k=1}^{n} \tilde{x}_{jk} = 0.$$

It can be calculated that the variance of PC j is

$$s^2_{\tilde{\boldsymbol{x}}_j} = \frac{\sum_{k=1}^{n} \tilde{x}^2_{jk}}{n-1} = \lambda_j.$$

Proportion of Variance Explained

The proportion of variance explained by PC j is

$$\text{PVE}_j = \frac{\lambda_j}{\sum_{k=1}^{m} \lambda_k}.$$

Singular Value Decomposition

One way in which PCA is performed is to carry out a **singular value decomposition** (SVD) of the data matrix $\boldsymbol{X}$. Let $q = \min(m, n)$. Recalling that $\boldsymbol{X}^*$ is the row-wise mean centered $\boldsymbol{X}$, we can take the SVD of $\boldsymbol{X}^*/\sqrt{n-1}$ to obtain

$$\frac{1}{\sqrt{n-1}}\boldsymbol{X}^* = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T$$

where $\boldsymbol{U}_{m\times q}$, $\boldsymbol{V}_{n\times q}$, and diagonal $\boldsymbol{D}_{q\times q}$. Also, we have the orthogonality properties $\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_q$. Finally, $\boldsymbol{D}$ is composed of diagonal elements $d_1 \geq d_2 \geq \cdots \geq d_q \geq 0$ where $d_q = 0$ if $q = n$.

Note that

$$\boldsymbol{S} = \frac{1}{n-1}\boldsymbol{X}^*\boldsymbol{X}^{*T} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T\left(\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T\right)^T = \boldsymbol{U}\boldsymbol{D}^2\boldsymbol{U}^T.$$

Therefore:

- The variance of PC j is $\lambda_j = d_j^2$
- The loadings of PC j are contained in the columns of the left-hand matrix from the decomposition of $\boldsymbol{S}$ or $\boldsymbol{X}^*$
- PC j is row j of $\boldsymbol{D}\boldsymbol{V}^T$

My PCA Function

```
> pca <- function(x, space=c("rows", "columns"),
+                 center=TRUE, scale=FALSE) {
+   space <- match.arg(space)
+   if(space=="columns") {x <- t(x)}
+   x <- t(scale(t(x), center=center, scale=scale))
+   x <- x/sqrt(nrow(x)-1)
+   s <- svd(x)
+   loading <- s$u
+   colnames(loading) <- paste0("Loading", 1:ncol(loading))
+   rownames(loading) <- rownames(x)
+   pc <- diag(s$d) %*% t(s$v)
+   rownames(pc) <- paste0("PC", 1:nrow(pc))
+   colnames(pc) <- colnames(x)
+   pve <- s$d^2 / sum(s$d^2)
+   if(space=="columns") {pc <- t(pc); loading <- t(loading)}
+   return(list(pc=pc, loading=loading, pve=pve))
+ }
```

How It Works

Input is as follows:

- `x`: a matrix of numerical values
- `space`: either `"rows"` or `"columns"`, denoting which dimension contains the variables
- `center`: if `TRUE` then the variables are mean centered before calculating PCs
- `scale`: if `TRUE` then the variables are std dev scaled before calculating PCs

Output is a list with the following items:

- `pc`: a matrix of all possible PCs

- `loading`: the weights or "loadings" that determined each PC
- `pve`: the proportion of variation explained by each PC

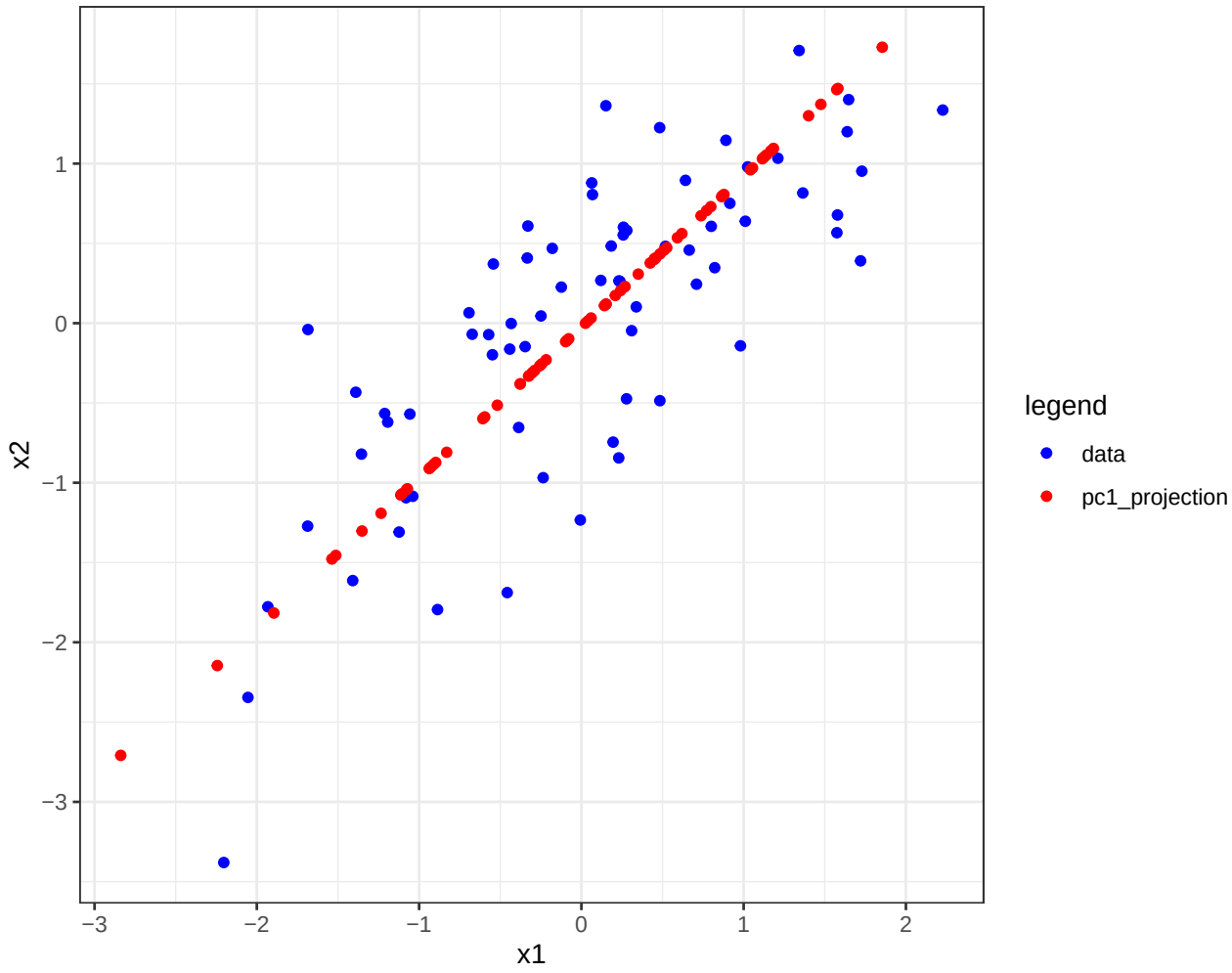
Note that the rows or columns of `pc` and `loading` have names to let you know on which dimension the values are organized.

The Ubiquitous Example

Here's an example very frequently encountered to explain PCA, but it's slightly complicated.

```
> set.seed(508)
> n <- 70
> z <- sqrt(0.8) * rnorm(n)
> x1 <- z + sqrt(0.2) * rnorm(n)
> x2 <- z + sqrt(0.2) * rnorm(n)
> X <- rbind(x1, x2)
> p <- pca(x=X, space="rows")
```

"The first PC finds the direction of maximal variance in the data..."



The above figure was made with the following code:

```
> df <- data.frame(x1=c(x1, lm(x1 ~ p$pc[1,])$fit),
+                  x2=c(x2, lm(x2 ~ p$pc[1,])$fit),
```

```
+                   legend=c(rep("data",n),rep("pc1_projection",n)))
> ggplot(df) + geom_point(aes(x=x1,y=x2,color=legend)) +
+   scale_color_manual(values=c("blue", "red"))
```

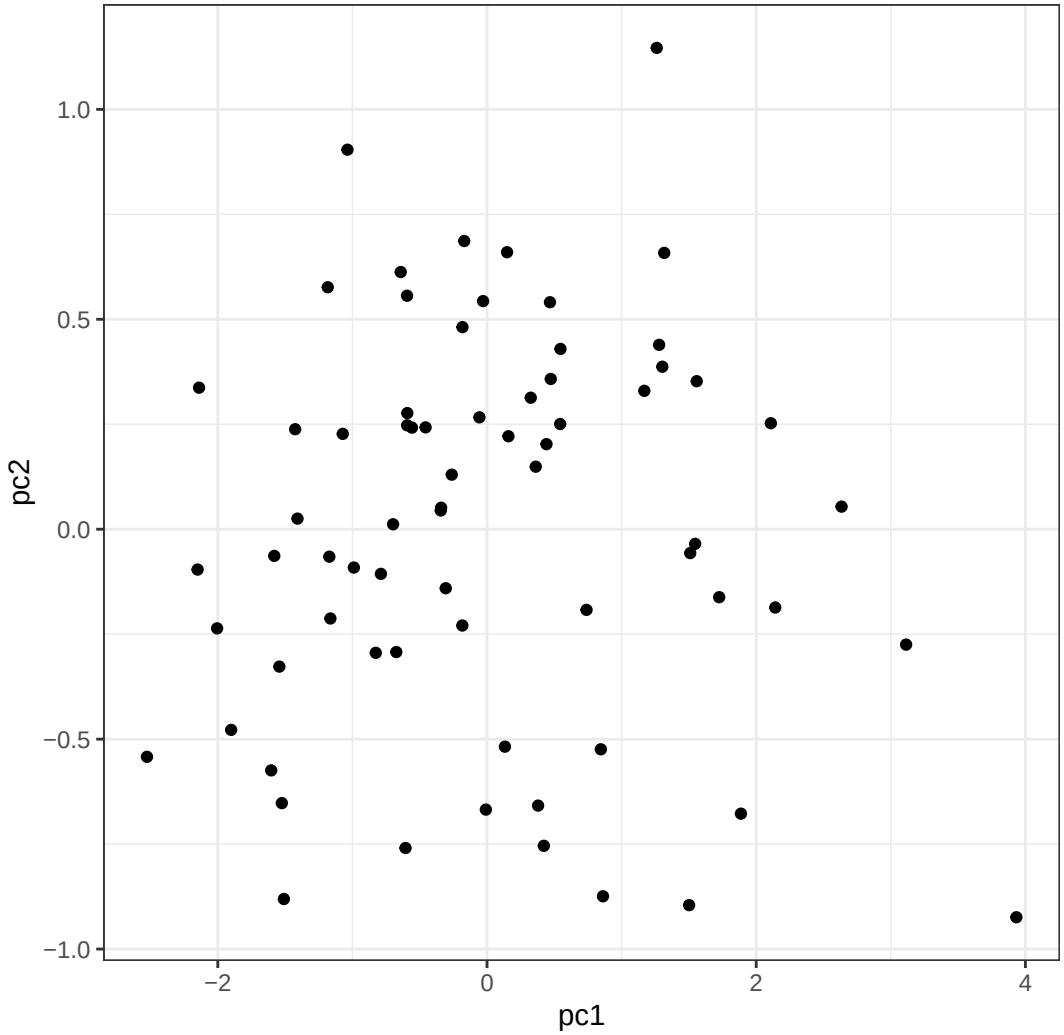
The red dots are therefore the projection of `x1` and `x2` onto the first PC, so they are neither the loadings nor the PC.

Note that

```
outer(p$loading[,1], p$pc[1,])[1,] + mean(x1)
# yields the same as
lm(x1 ~ p$pc[1,])$fit # and
outer(p$loading[,1], p$pc[1,])[2,] + mean(x2)
# yields the same as
lm(x2 ~ p$pc[1,])$fit
```

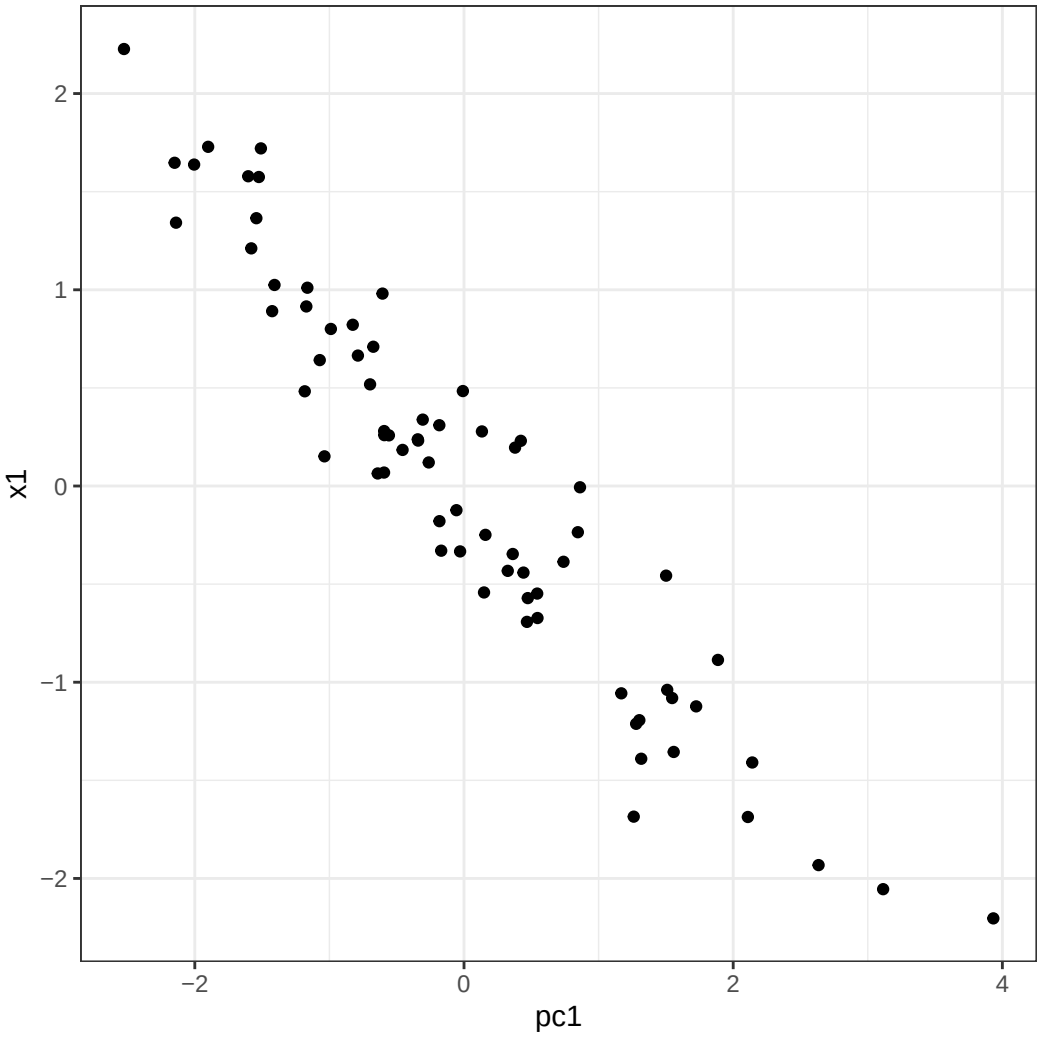
Here is PC1 vs PC2:

```
> data.frame(pc1=p$pc[1,], pc2=p$pc[2,]) %>%
+   ggplot() + geom_point(aes(x=pc1,y=pc2)) +
+   theme(aspect.ratio=1)
```



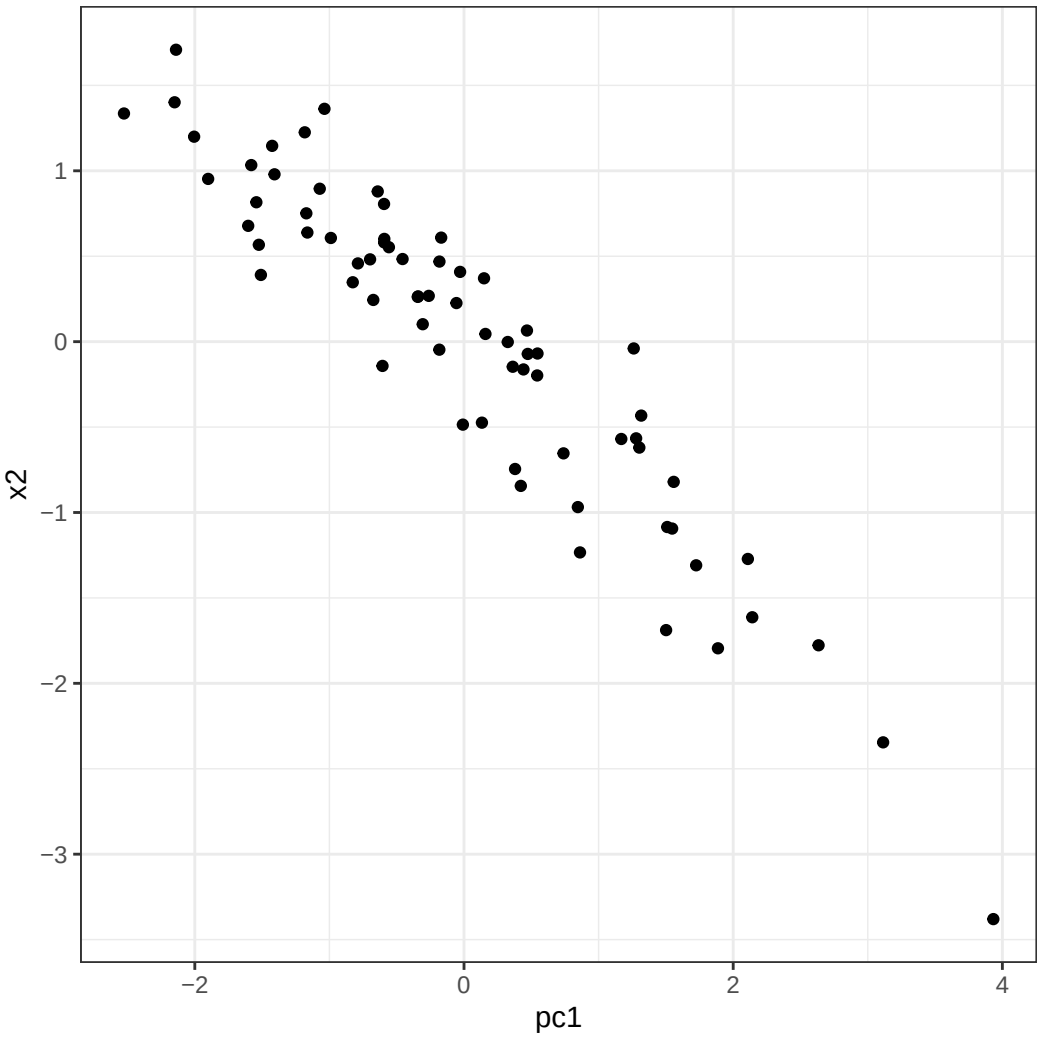
Here is PC1 vs `x1`:

```
> data.frame(pc1=p$pc[1,], x1=x1) %>%
+   ggplot() + geom_point(aes(x=pc1,y=x1)) +
+   theme(aspect.ratio=1)
```



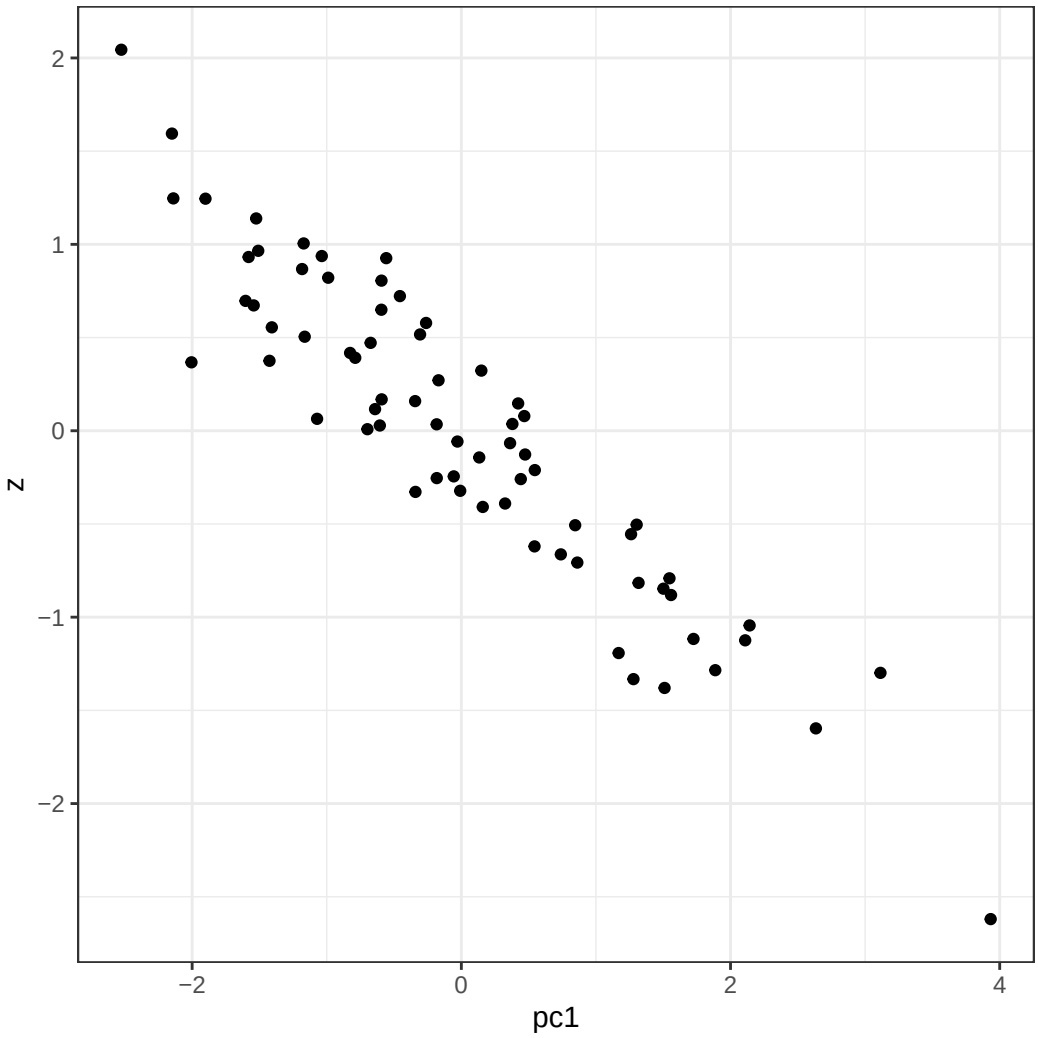
Here is PC1 vs x2:

```
> data.frame(pc1=p$pc[1,], x2=x2) %>%
+   ggplot() + geom_point(aes(x=pc1,y=x2)) +
+   theme(aspect.ratio=1)
```



Here is PC1 vs z:

```
> data.frame(pc1=p$pc[1,], z=z) %>%
+   ggplot() + geom_point(aes(x=pc1,y=z)) +
+   theme(aspect.ratio=1)
```



PCA Examples

Example: Weather Data

These daily temperature data (in tenths of degrees C) come from meteorogical observations for weather stations in the US for the year 2012 provided by NOAA (National Oceanic and Atmospheric Administration).:

```
> load("./data/weather_data.RData")
> dim(weather_data)
[1] 2811   50
>
> weather_data[1:5, 1:7]
                 11       16  18       19  27  30       31
AG000060611 138.0000 175.0000 173 164.0000 218 160 163.0000
AGM00060369 158.0000 162.0000 154 159.0000 165 125 171.0000
AGM00060425 272.7619 272.7619 152 163.0000 163 108 158.0000
AGM00060444 128.0000 102.0000 100 111.0000 125  33 125.0000
AGM00060468 105.0000 122.0000  97 263.5714 155  52 263.5714
```

This matrix contains temperature data on 50 days and 2811 stations that were randomly selected.

Convert temperatures to Fahrenheit:

```
> weather_data <- 0.18*weather_data + 32
> weather_data[1:5, 1:6]
                 11       16    18       19    27    30
AG000060611 56.84000 63.50000 63.14 61.52000 71.24 60.80
AGM00060369 60.44000 61.16000 59.72 60.62000 61.70 54.50
AGM00060425 81.09714 81.09714 59.36 61.34000 61.34 51.44
AGM00060444 55.04000 50.36000 50.00 51.98000 54.50 37.94
AGM00060468 50.90000 53.96000 49.46 79.44286 59.90 41.36
>
> apply(weather_data, 1, median) %>%
+   quantile(probs=seq(0,1,0.1))
       0%        10%        20%        30%        40%
  8.886744  49.010000  54.500000  58.460000  62.150000
      50%        60%        70%        80%        90%
 65.930000  69.679318  73.490000  77.990000  82.940000
     100%
140.000000
```

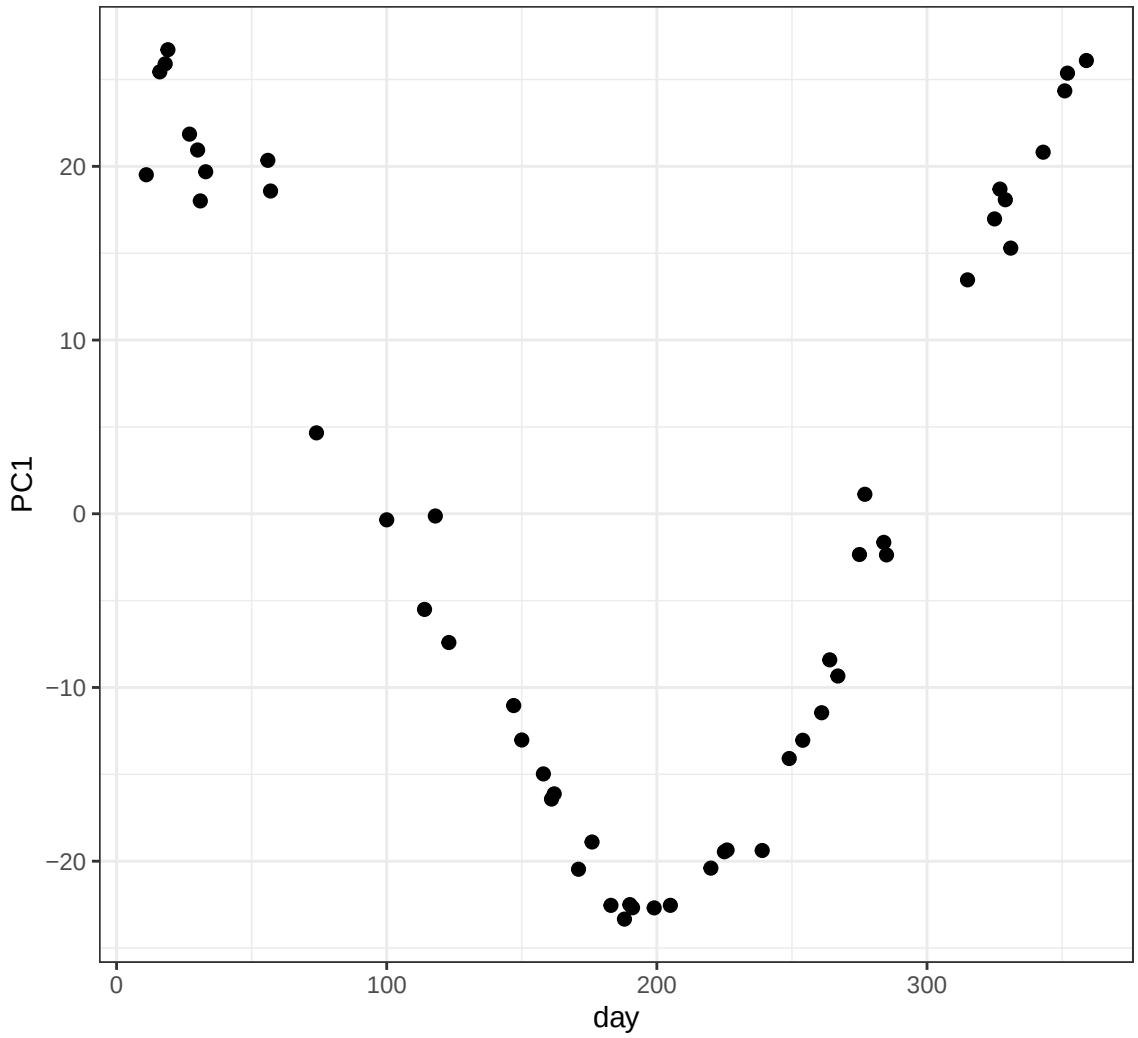
PCA of Weather Data

```
> mypca <- pca(weather_data, space="rows")
>
> names(mypca)
[1] "pc"      "loading" "pve"
> dim(mypca$pc)
[1] 50 50
> dim(mypca$loading)
[1] 2811   50
```

```
> mypca$pc[1:3, 1:3]
            11        16         18
PC1 19.5166741 25.441401 25.9023874
PC2 -2.6025225 -4.310673  0.9707207
PC3 -0.6681223 -1.240748 -3.7276658
> mypca$loading[1:3, 1:3]
                Loading1    Loading2     Loading3
AG000060611 -0.015172744 0.013033849 -0.011273121
AGM00060369 -0.009439176 0.016884418 -0.004611284
AGM00060425 -0.015779138 0.007026312 -0.009907972
```

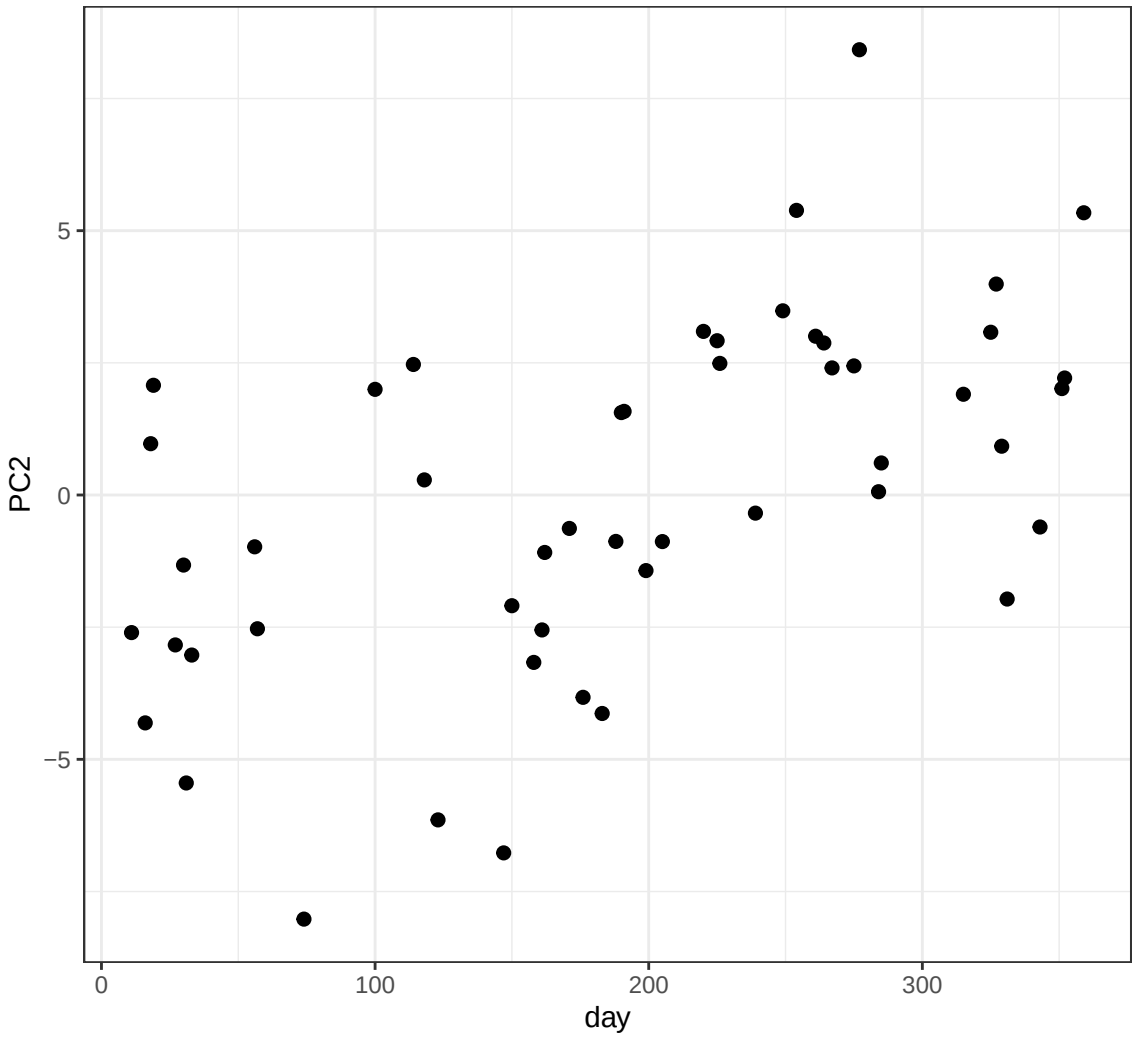
PC1 vs Time

```
> day_of_the_year <- as.numeric(colnames(weather_data))
> data.frame(day=day_of_the_year, PC1=mypca$pc[1,]) %>%
+   ggplot() + geom_point(aes(x=day, y=PC1), size=2)
```



PC2 vs Time

```
> data.frame(day=day_of_the_year, PC2=mypca$pc[2,]) %>%
+   ggplot() + geom_point(aes(x=day, y=PC2), size=2)
```



PC Biplots

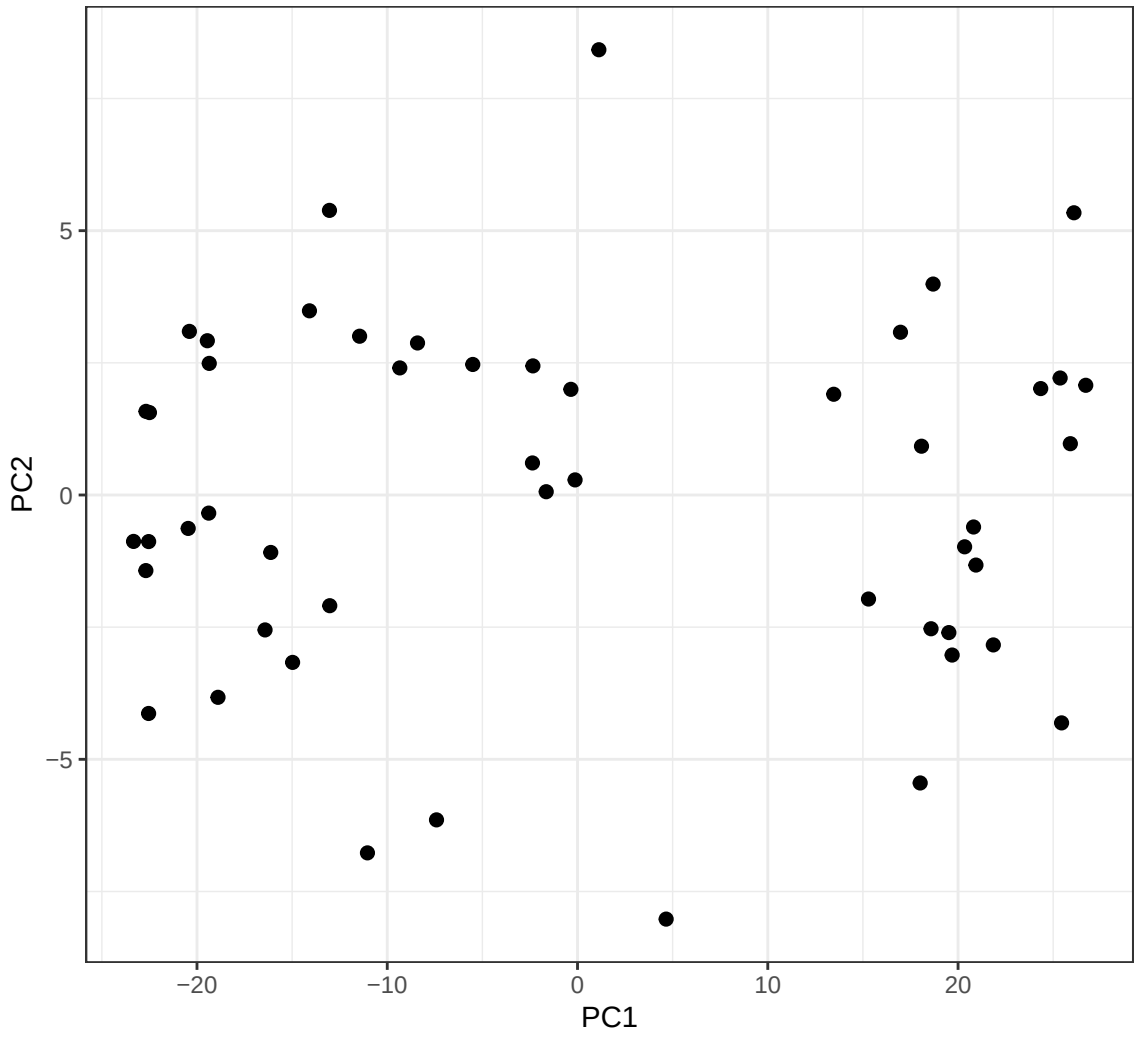
Sometimes it is informative to plot a PC versus another PC. This is called a **PC biplot**.

It is possible that interesting subgroups or clusters of *observations* will emerge.

This does not appear to be the case in the weather data set, however, due to what we observe in the next two plots.

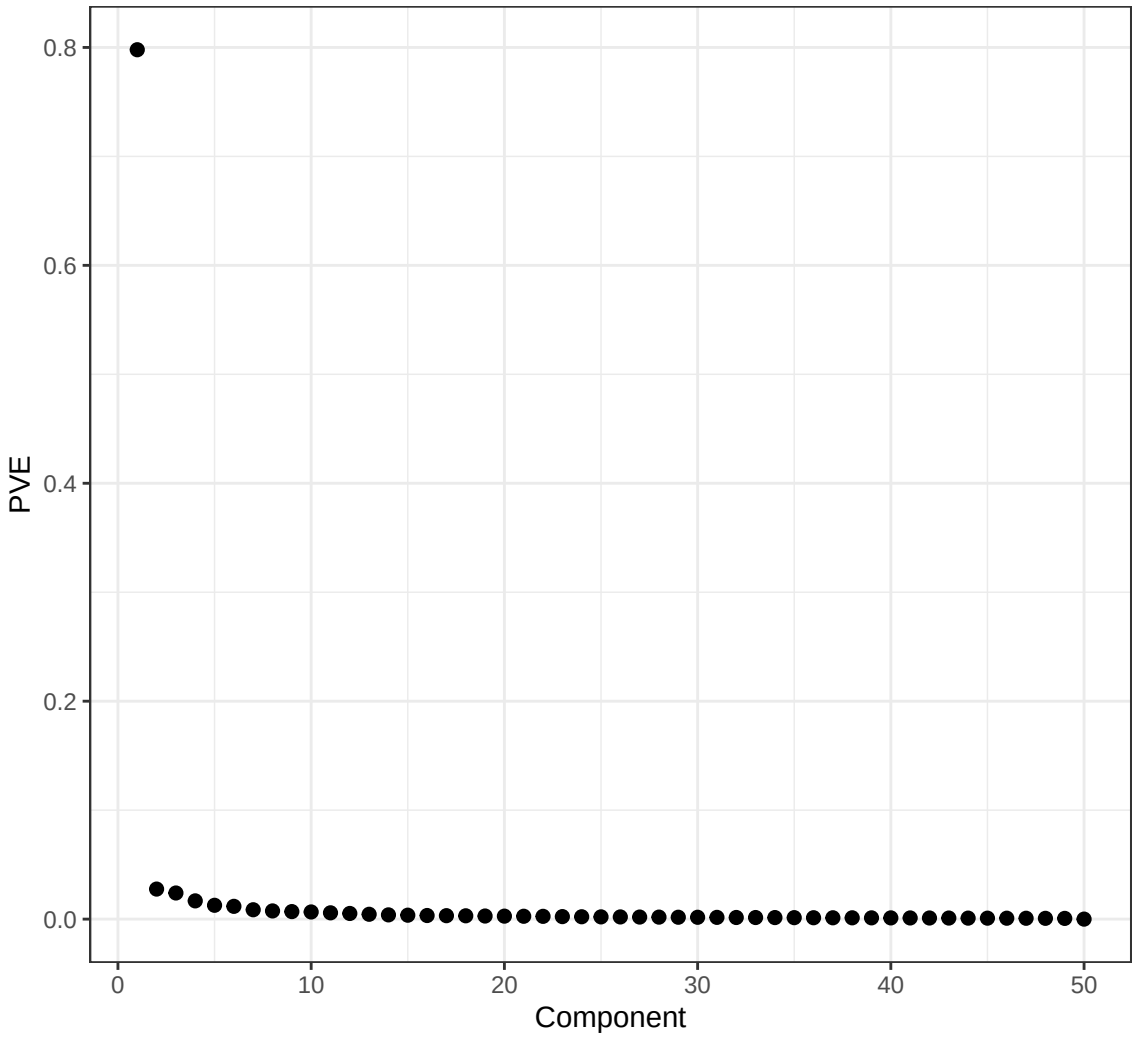
PC1 vs PC2 Biplot

```
> data.frame(PC1=mypca$pc[1,], PC2=mypca$pc[2,]) %>%
+   ggplot() + geom_point(aes(x=PC1, y=PC2), size=2)
```



Proportion of Variance Explained

```
> data.frame(Component=1:length(mypca$pve), PVE=mypca$pve) %>%
+   ggplot() + geom_point(aes(x=Component, y=PVE), size=2)
```



PCs Reproduce the Data

We can multiple the loadings matrix by the PCs matrix to reproduce the data:

```
> # mean centered weather data
> weather_data_mc <- weather_data - rowMeans(weather_data)
>
> # difference between the PC projections and the data
> # the small sum is just machine imprecision
> sum(abs(weather_data_mc/sqrt(nrow(weather_data_mc)-1) -
+           mypca$loading %*% mypca$pc))
[1] 1.329755e-10
```

Loadings

The sum of squared weights – i.e., loadings – equals one for each component:

```
> sum(mypca$loading[,1]^2)
[1] 1
>
> apply(mypca$loading, 2, function(x) {sum(x^2)})
 Loading1  Loading2  Loading3  Loading4  Loading5  Loading6
```

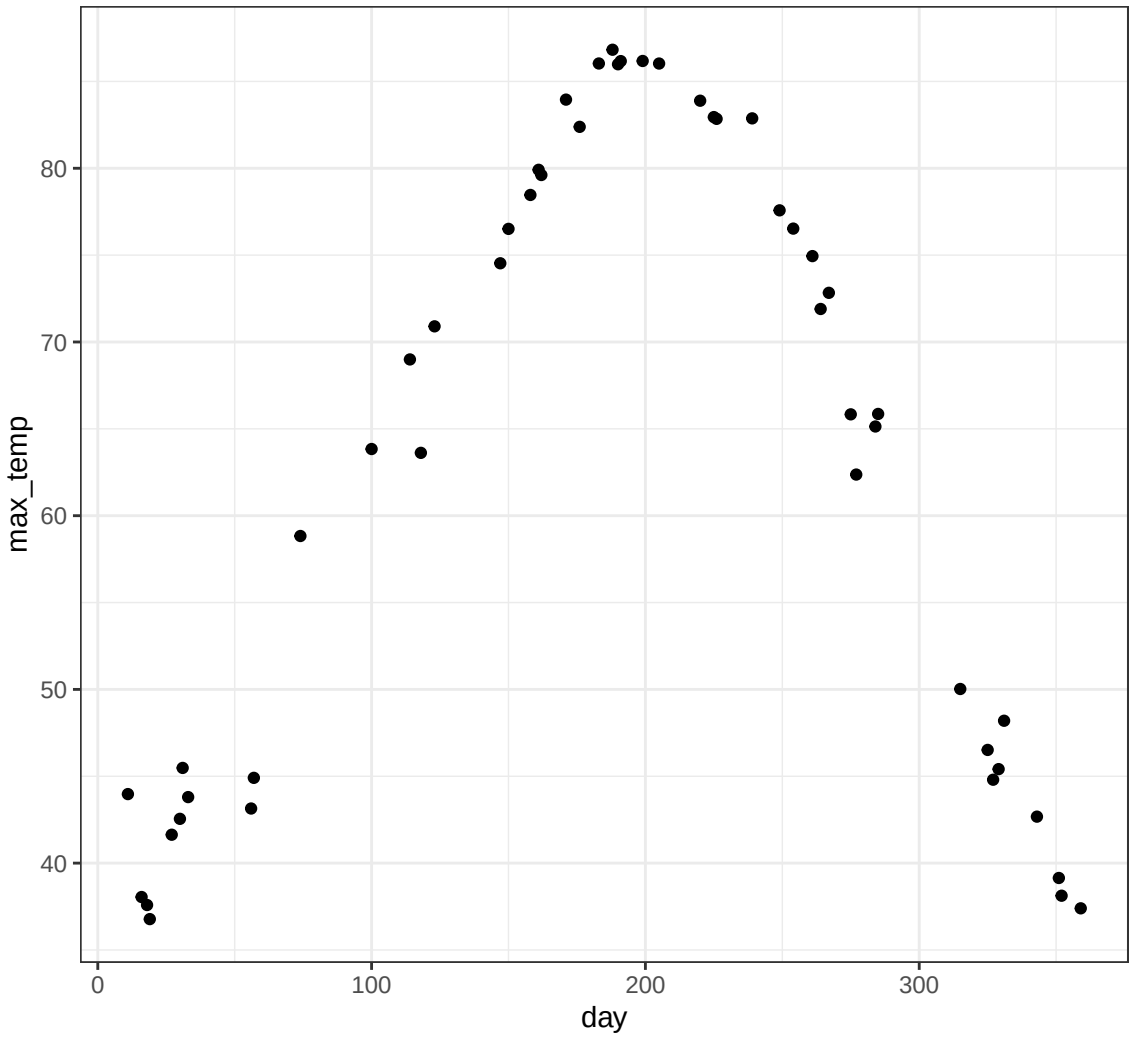
```
        1         1         1         1         1         1
 Loading7  Loading8  Loading9 Loading10 Loading11 Loading12
        1         1         1         1         1         1
Loading13 Loading14 Loading15 Loading16 Loading17 Loading18
        1         1         1         1         1         1
Loading19 Loading20 Loading21 Loading22 Loading23 Loading24
        1         1         1         1         1         1
Loading25 Loading26 Loading27 Loading28 Loading29 Loading30
        1         1         1         1         1         1
Loading31 Loading32 Loading33 Loading34 Loading35 Loading36
        1         1         1         1         1         1
Loading37 Loading38 Loading39 Loading40 Loading41 Loading42
        1         1         1         1         1         1
Loading43 Loading44 Loading45 Loading46 Loading47 Loading48
        1         1         1         1         1         1
Loading49 Loading50
        1         1
```

Pairs of PCs Have Correlaton Zero

PCs by contruction have sample correlation equal to zero:

```
> cor(mypca$pc[1,], mypca$pc[2,])
[1] 3.135149e-17
> cor(mypca$pc[1,], mypca$pc[3,])
[1] 2.273613e-16
> cor(mypca$pc[1,], mypca$pc[12,])
[1] -1.231339e-16
> cor(mypca$pc[5,], mypca$pc[27,])
[1] -2.099516e-17
> # etc...
```

```
> day_of_the_year <- as.numeric(colnames(weather_data))
> y <- -mypca$pc[1,] + mean(weather_data)
> data.frame(day=day_of_the_year, max_temp=y) %>%
+   ggplot() + geom_point(aes(x=day, y=max_temp))
```



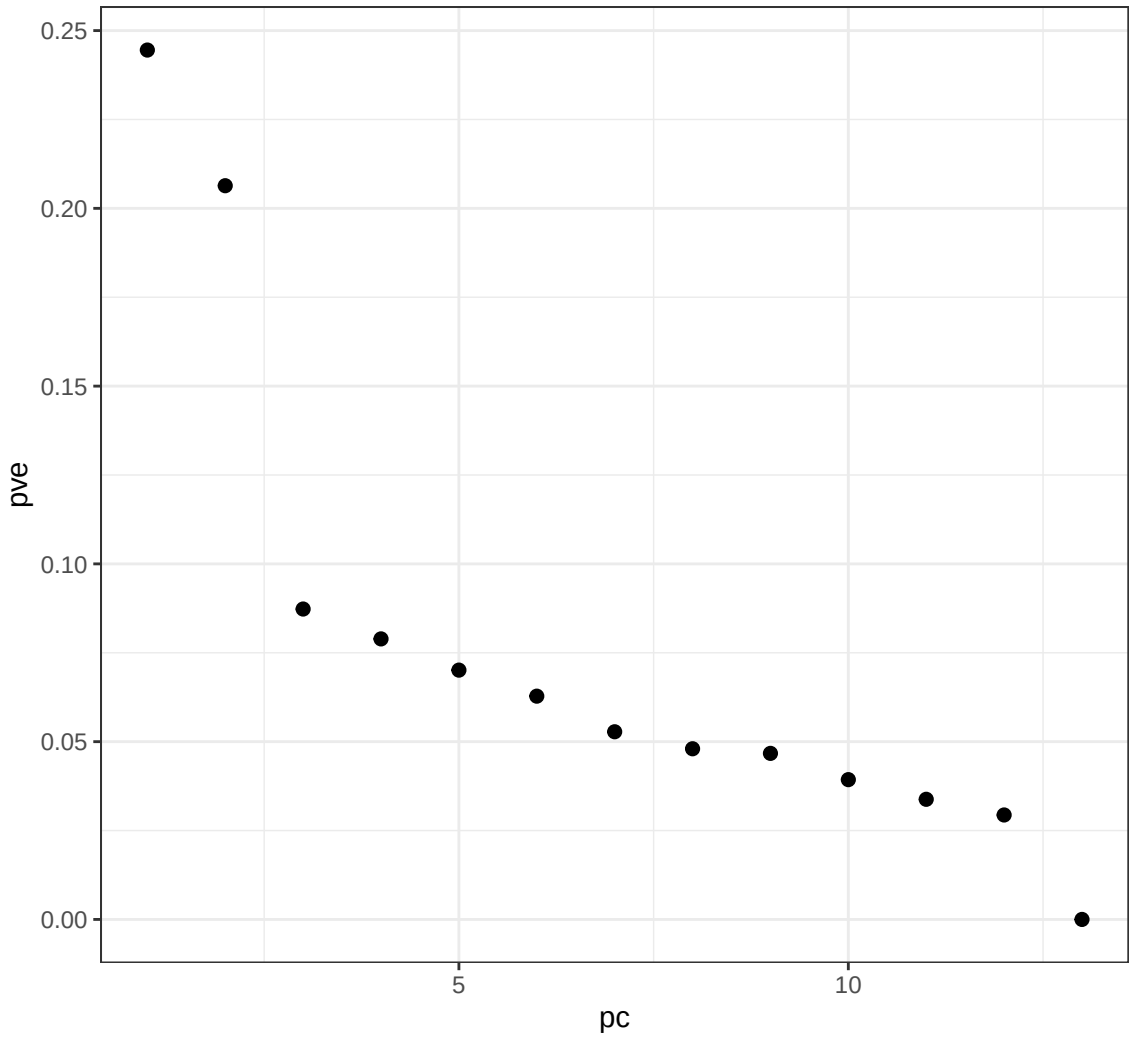
Example: Yeast Gene Expression

Yeast cells were synchronized so that they were on the same approximate cell cycle timing. The goal was to understand how gene expression varies over the cell cycle from a genome-wide perspective.

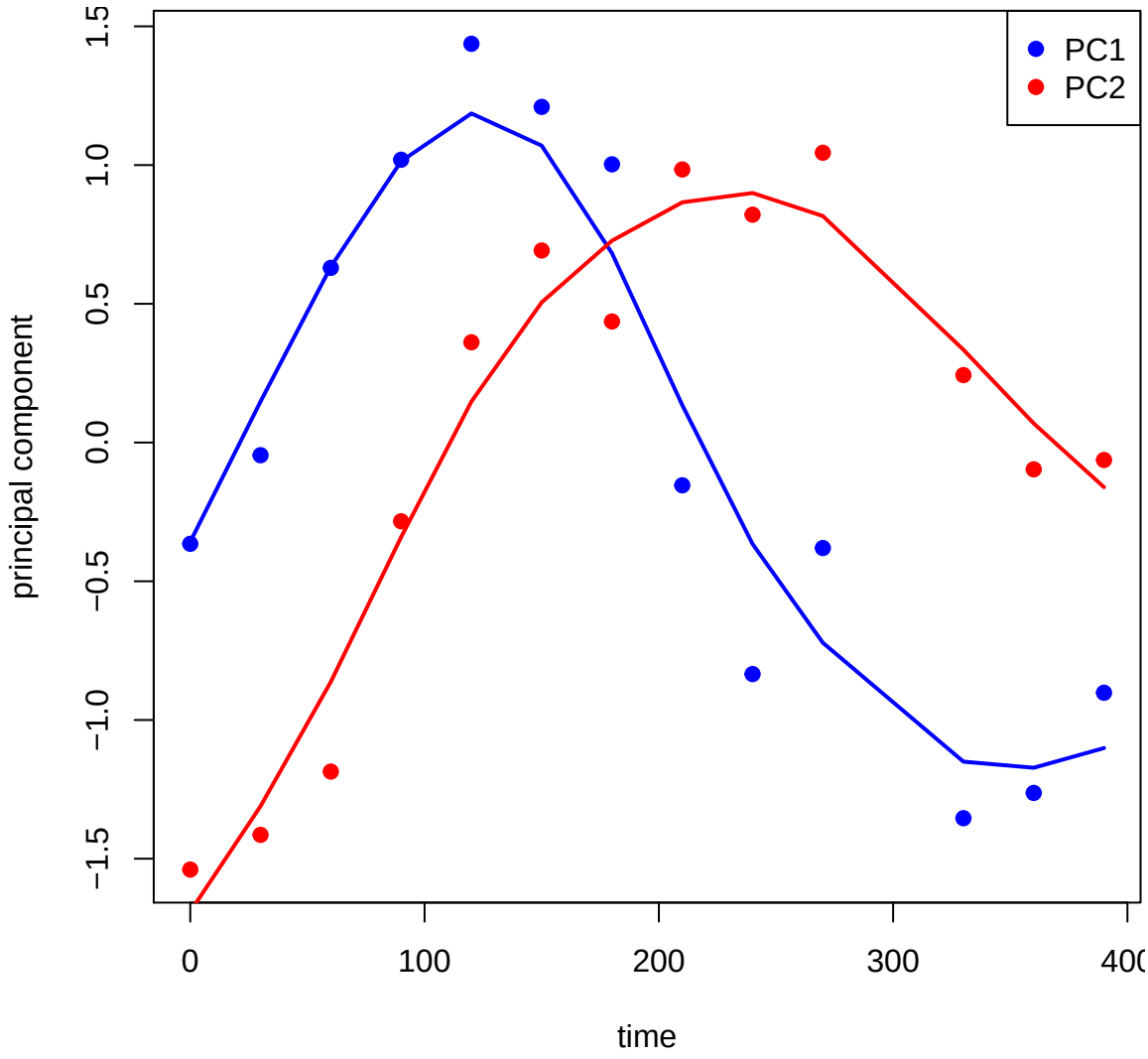
```
> load("./data/spellman.RData")
> time
 [1]   0  30  60  90 120 150 180 210 240 270 330 360 390
> dim(gene_expression)
[1] 5981   13
> gene_expression[1:6,1:5]
                  0         30         60        90        120
YAL001C  0.69542786 -0.4143538  3.2350520 1.6323737 -2.1091820
YAL002W -0.01210662  3.0465649  1.1062193 4.0591467 -0.1166399
YAL003W -2.78570526 -1.0156981 -2.1387564 1.9299681  0.7797033
YAL004W  0.55165887  0.6590093  0.5857847 0.3890409 -1.0009777
YAL005C -0.53191556  0.1577985 -1.2401448 0.8170350 -1.3520947
YAL007C -0.86693416 -1.1642322 -0.6359588 1.1179131  1.9587021
```

Proportion Variance Explained

```
> p <- pca(gene_expression, space="rows")
> ggplot(data.frame(pc=1:13, pve=p$pve)) +
+   geom_point(aes(x=pc,y=pve), size=2)
```



PCs vs Time (with Smoothers)



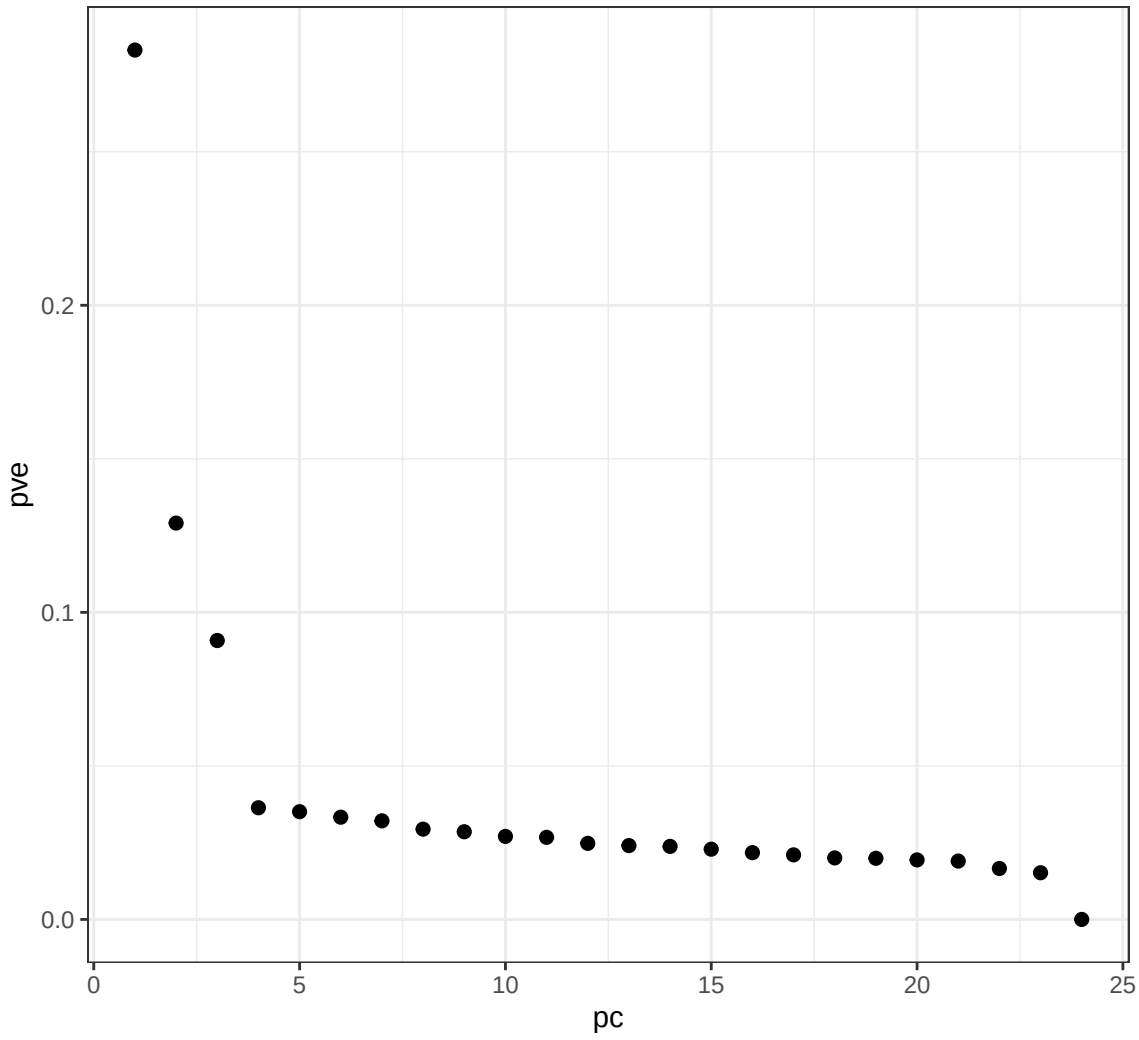
Example: HapMap Genotypes

Individuals with ancestries corresponding to different geographical locations were sampled in the HapMap project. The researchers obtained genome-wide SNP genotypes for each individual. We curated a small data set that cleanly separates human subpopulations into three distinct ancestral continents (Africa, (East) Asia, Europe).

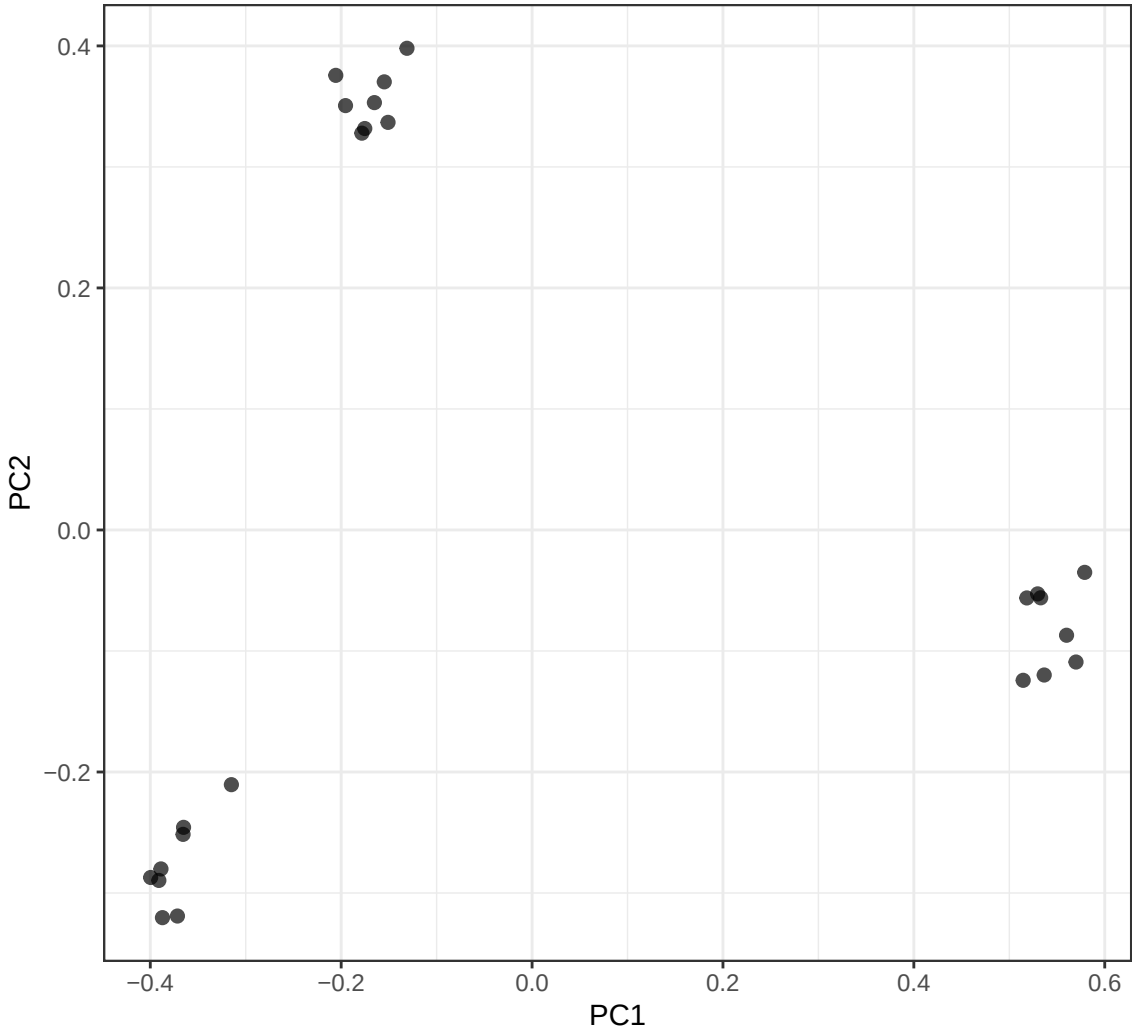
```
> hapmap <- read.table("./data/hapmap_sample.txt")
> dim(hapmap)
[1] 400  24
> hapmap[1:6,1:6]
           NA18516 NA19138 NA19137 NA19223 NA19200 NA19131
rs2051075        0       1       2       1       1       1
rs765546         2       2       0       0       0       0
rs10019399       2       2       2       1       1       2
rs7055827        2       2       1       2       0       2
rs6943479        0       0       2       0       1       0
rs2095381        1       2       1       2       1       1
```

Proportion Variance Explained

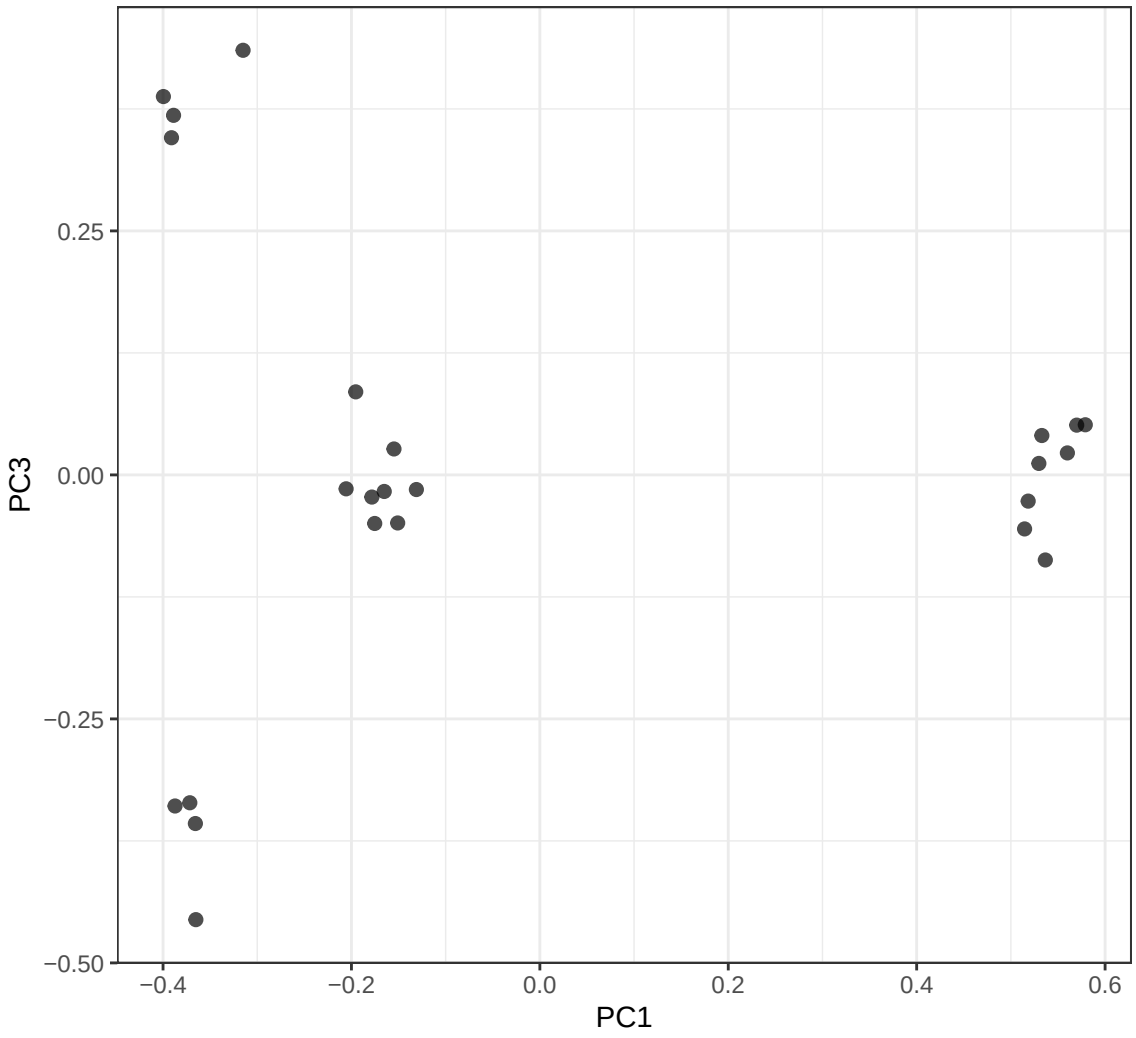
```
> p <- pca(hapmap, space="rows")
> ggplot(data.frame(pc=(1:ncol(hapmap)), pve=p$pve)) +
+   geom_point(aes(x=pc,y=pve), size=2)
```



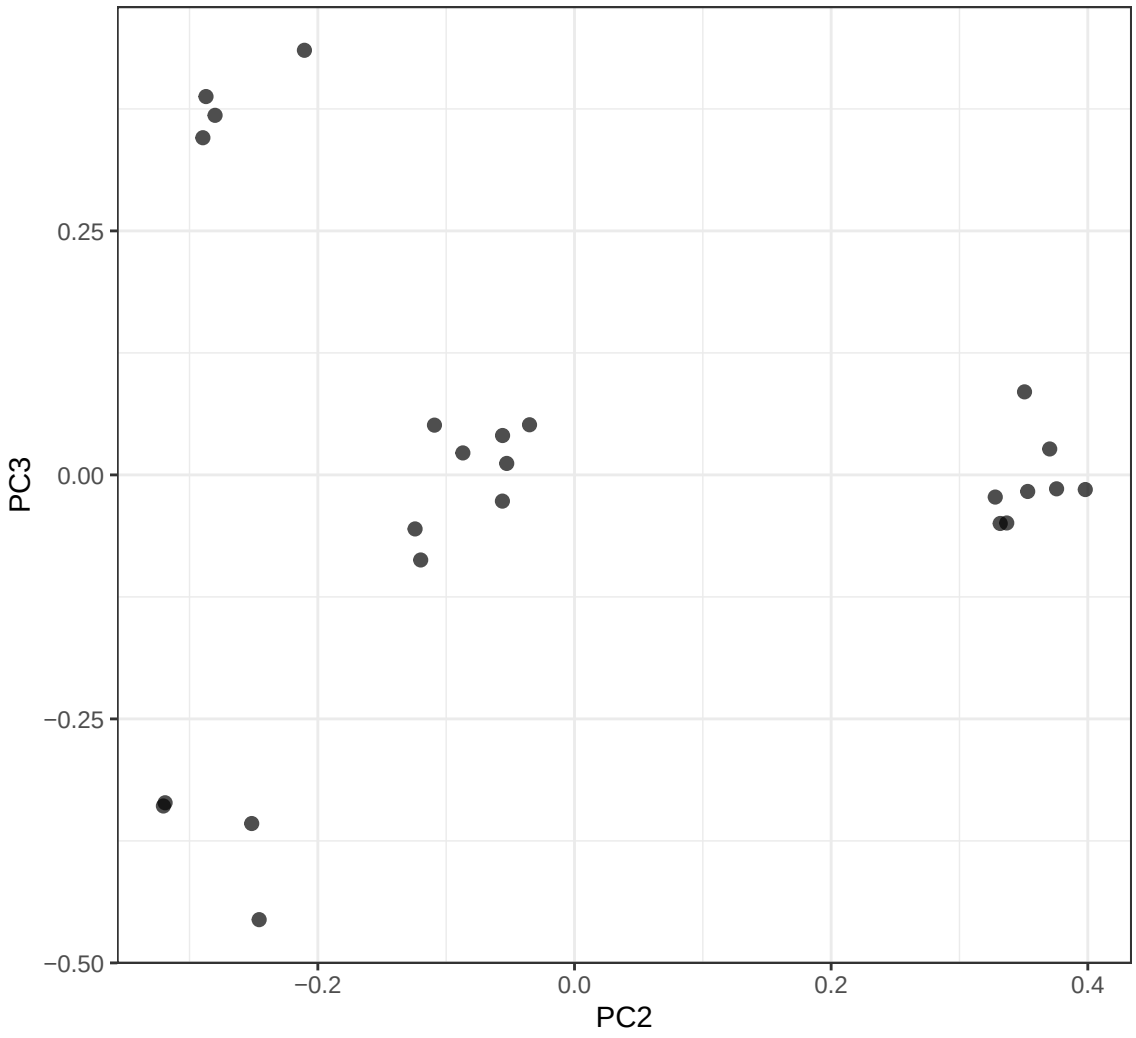
PC1 vs PC2 Biplot



PC1 vs PC3 Biplot



PC2 vs PC3 Biplot



Statistical Models

Probabilistic Models

So far we have covered inference of paramters that quantify a population of interest.

This is called inference of probabilistic models.

Multivariate Models

Some of the probabilistic models we considered involve calculating conditional probabilities such as $\Pr(\boldsymbol{Z}|\boldsymbol{X};\boldsymbol{\theta})$ or $\Pr(\boldsymbol{\theta}|\boldsymbol{X})$.

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*. **Statistical modeling** typically refers to this goal.

Variables

Let's suppose our does comes in the form $(\boldsymbol{X}_1, Y_1), (\boldsymbol{X}_2, Y_2), \ldots, (\boldsymbol{X}_n, Y_n) \sim F$.

We will call $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{ip}) \in \mathbb{R}_{1\times p}$ the **explanatory variables** and $Y_i \in \mathbb{R}$ the **dependent variable** or **response variable**.

We can collect all variables as matrices

$$\boldsymbol{Y}_{n\times 1} \quad \text{and} \quad \boldsymbol{X}_{n\times p}$$

where each row is a unique observation.

Statistical Model

Statistical models are concerned with *how* variables are dependent. The most general model would be to infer

$$\Pr(Y|\boldsymbol{X}) = h(\boldsymbol{X})$$

where we would specifically study the form of $h(\cdot)$ to understand how Y is dependent on $\boldsymbol{X}$.

A more modest goal is to infer the transformed conditional expecation

$$g\left(\mathrm{E}[Y|\boldsymbol{X}]\right) = h(\boldsymbol{X})$$

which sometimes leads us back to an estimate of $\Pr(Y|\boldsymbol{X})$.

Parametric vs Nonparametric

A **parametric** model is a pre-specified form of $h(X)$ whose terms can be characterized by a formula and interpreted. This usually involves parameters on which inference can be performed, such as coefficients in a linear model.

A **nonparametric** model is a data-driven form of $h(X)$ that is often very flexible and is not easily expressed or intepreted. A nonparametric model often does not include parameters on which we can do inference.

Simple Linear Regression

For random variables $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$, **simple linear regression** estimates the model

$$Y_i = \beta_1 + \beta_2 X_i + E_i$$

where $\mathrm{E}[E_i] = 0$, $\mathrm{Var}(E_i) = \sigma^2$, and $\mathrm{Cov}(E_i, E_j) = 0$ for all $1 \leq i, j \leq n$ and $i \neq j$.

Note that in this model $\mathrm{E}[Y|X] = \beta_1 + \beta_2 X$.

Ordinary Least Squares

Ordinary least squares (OLS) estimates the model

$$\begin{aligned} Y_i &= \beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip} + E_i \\ &= \boldsymbol{X}_i\boldsymbol{\beta} + E_i \end{aligned}$$

where $\mathrm{E}[E_i] = 0$, $\mathrm{Var}(E_i) = \sigma^2$, and $\mathrm{Cov}(E_i, E_j) = 0$ for all $1 \leq i, j \leq n$ and $i \neq j$.

Note that typically $X_{i1} = 1$ for all i so that $\beta_1 X_{i1} = \beta_1$ serves as the intercept.

Generalized Least Squares

Generalized least squares (GLS) assumes the same model as OLS, except it allows for **heteroskedasticity** and **covariance** among the E_i. Specifically, it is assumed that $\boldsymbol{E} = (E_1, \ldots, E_n)^T$ is distributed as

$$\boldsymbol{E}_{n\times 1} \sim (\boldsymbol{0}, \boldsymbol{\Sigma})$$

where $\boldsymbol{0}$ is the expected value $\boldsymbol{\Sigma} = (\sigma_{ij})$ is the $n \times n$ symmetric covariance matrix.

Matrix Form of Linear Models

We can write the models as

$$\boldsymbol{Y}_{n\times 1} = \boldsymbol{X}_{n\times p}\boldsymbol{\beta}_{p\times 1} + \boldsymbol{E}_{n\times 1}$$

where simple linear regression, OLS, and GLS differ in the value of p or the distribution of the E_i. We can also write the conditional expecation and covariance as

$$\mathrm{E}[\boldsymbol{Y}|\boldsymbol{X}] = \boldsymbol{X}\boldsymbol{\beta},\ \mathrm{Cov}(\boldsymbol{Y}|\boldsymbol{X}) = \boldsymbol{\Sigma}.$$

Least Squares Regression

In simple linear regression, OLS, and GLS, the $\boldsymbol{\beta}$ parameters are fit by minimizing the sum of squares between $\boldsymbol{Y}$ and $\boldsymbol{X}\boldsymbol{\beta}$.

Fitting these models by "least squares" satisfies two types of optimality:

1. Gauss-Markov Theorem
2. Maximum likelihood estimate when in addition $\boldsymbol{E} \sim \mathrm{MVN}_n(\boldsymbol{0}, \boldsymbol{\Sigma})$

Details will follow on these.

Generalized Linear Models

The generalized linear model (GLM) builds from OLS and GLS to allow the response variable to be distributed according to an exponential family distribution. Suppose that $\eta(\theta)$ is function of the expected value into the natural parameter. The estimated model is

$$\eta\left(\mathrm{E}[Y|\boldsymbol{X}]\right) = \boldsymbol{X}\boldsymbol{\beta}$$

which is fit by maximized likelihood estimation.

Generalized Additive Models

One can formulate semiparametric models where $Y|\boldsymbol{X}$ is distributed according to an exponential family distribution. The models, which are called **generalized additive models** (GAMs), will be of the form

$$\eta\left(\mathrm{E}[Y|\boldsymbol{X}]\right) = \sum_{j=1}^{p}\sum_{k=1}^{d} h_k(X_j)$$

where η is the canonical link function and the $h_k(\cdot)$ functions are very flexible (the nonparametric part).

Some Trade-offs

There are several important trade-offs encountered in statistical modeling:

- Bias vs variance
- Accuracy vs computational time
- Flexibility vs intepretability

These are not mutually exclusive phenomena.

Bias and Variance

Suppose we estimate $Y = h(\boldsymbol{X}) + E$ by some $\hat{Y} = \hat{h}(\boldsymbol{X})$. The following bias-variance trade-off exists:

$$\begin{aligned}
\mathrm{E}\left[\left(Y - \hat{Y}\right)^2\right] &= \mathrm{E}\left[\left(h(\boldsymbol{X}) + E - \hat{h}(\boldsymbol{X})\right)^2\right] \\
&= \mathrm{E}\left[\left(h(\boldsymbol{X}) - \hat{h}(\boldsymbol{X})\right)^2\right] + \mathrm{Var}(E) \\
&= \left(h(\boldsymbol{X}) - \mathrm{E}[\hat{h}(\boldsymbol{X})]\right)^2 + \mathrm{Var}\left(\hat{h}(\boldsymbol{X})\right) + \mathrm{Var}(E) \\
&= \mathrm{bias}^2 + \mathrm{variance} + \mathrm{Var}(E)
\end{aligned}$$

Motivating Examples

Sample Correlation

Least squares regression "modelizes" 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

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^n (x_i - \overline{x})^2 \sum_{i=1}^n (y_i - \overline{y})^2}} \tag{1}$$

$$= \frac{\sum_{i=1}^n (x_i - \overline{x})(y_i - \overline{y})}{(n-1) s_x s_y} \tag{2}$$

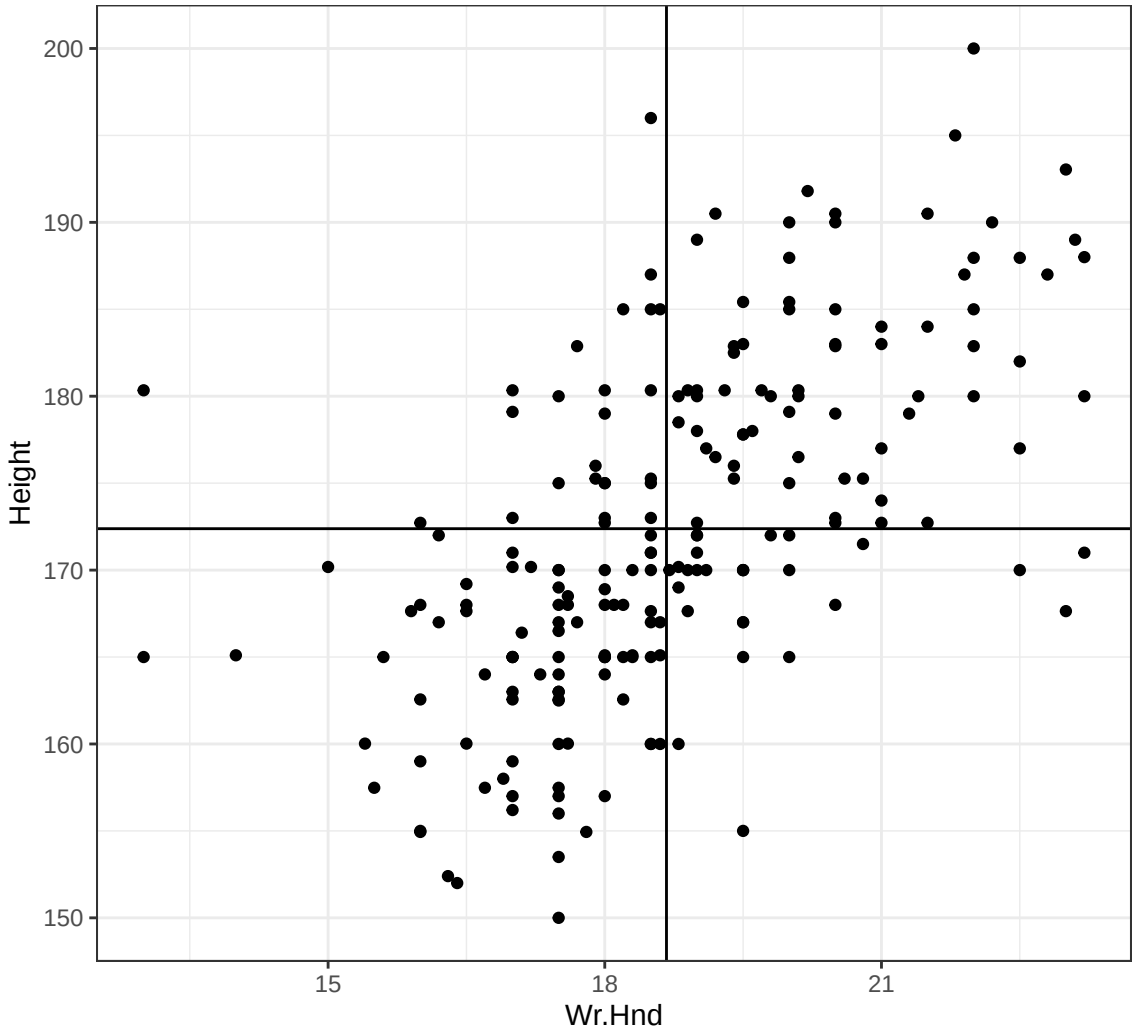
where s_x and s_y are the sample standard deviations of each measured variable.

Example: Hand Size Vs. Height

```
> library("MASS")
> data("survey", package="MASS")
> head(survey)
     Sex Wr.Hnd NW.Hnd W.Hnd    Fold Pulse    Clap Exer Smoke
1 Female   18.5   18.0 Right  R on L    92    Left Some Never
2   Male   19.5   20.5  Left  R on L   104    Left None Regul
3   Male   18.0   13.3 Right  L on R    87 Neither None Occas
4   Male   18.8   18.9 Right  R on L    NA Neither None Never
5   Male   20.0   20.0 Right Neither    35   Right Some Never
6 Female   18.0   17.7 Right  L on R    64   Right Some Never
  Height      M.I    Age
1 173.00   Metric 18.250
2 177.80 Imperial 17.583
3     NA     <NA> 16.917
```

```
4 160.00   Metric 20.333
5 165.00   Metric 23.667
6 172.72 Imperial 21.000
```

```
> 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))
```



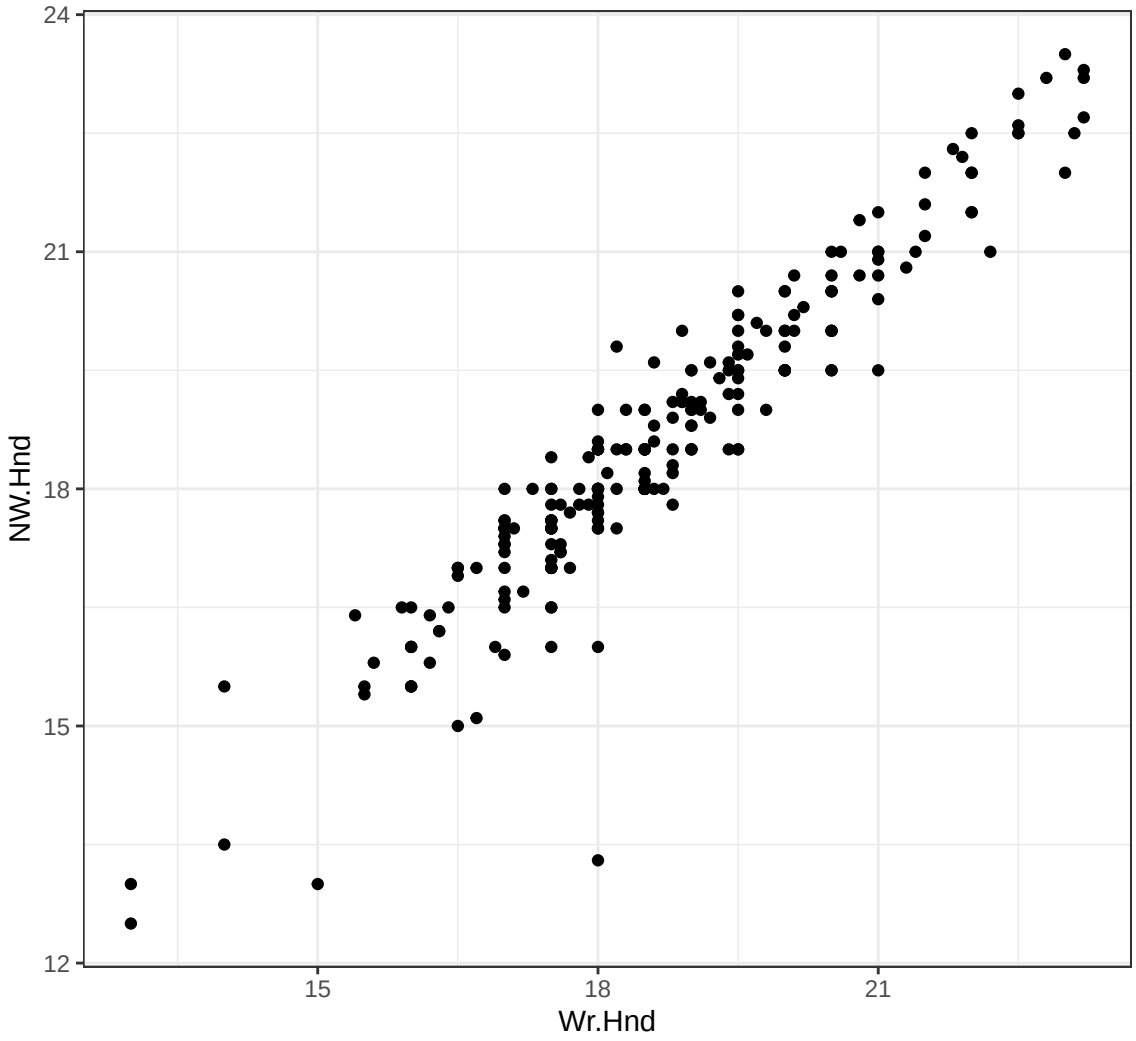
Cor. of Hand Size and Height

```
> 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
```

L/R 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
```

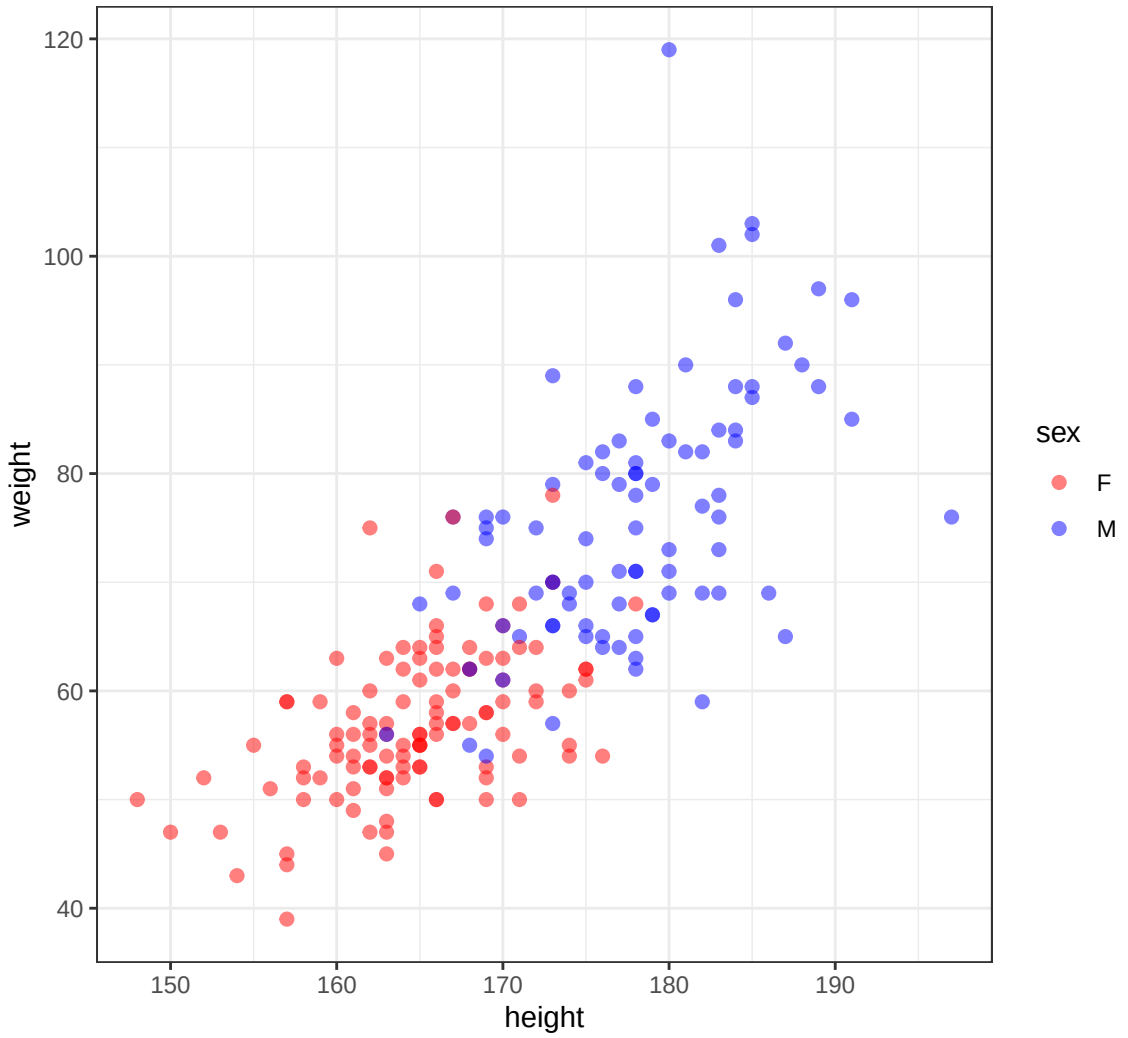
Davis Data

```
> library("car")
> data("Davis", package="car")
Warning in data("Davis", package = "car"): data set 'Davis' not
found
```

```
> htwt <- tbl_df(Davis)
> htwt[12,c(2,3)] <- htwt[12,c(3,2)]
> head(htwt)
# A tibble: 6 x 5
  sex   weight height repwt repht
  <fct>  <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 of Height and Weight

```
> 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 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?

Simple Linear Regression

Definition

For random variables $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$, **simple linear regression** estimates the model

$$Y_i = \beta_1 + \beta_2 X_i + E_i$$

where $\mathrm{E}[E_i] = 0$, $\mathrm{Var}(E_i) = \sigma^2$, and $\mathrm{Cov}(E_i, E_j) = 0$ for all $1 \leq i, j \leq n$ and $i \neq j$.

Rationale

- **Least squares linear regression** is one of the simplest and most useful modeling systems for building a model that explains the variation of one variable in terms of other variables.
- 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

- 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 $\beta_1 + \beta_2 x$, where β_1 and β_2 are chosen to satisfy a least squares optimization.

Line Minimizing Squared Error

The least squares regression line is formed from the value of β_1 and β_2 that minimize:

$$\sum_{i=1}^{n} (y_i - \beta_1 - \beta_2 x_i)^2 .$$

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 $\hat{\beta}_1$ and $\hat{\beta}_2$ be the values that minimize this sum of squares.

Least Squares Solution

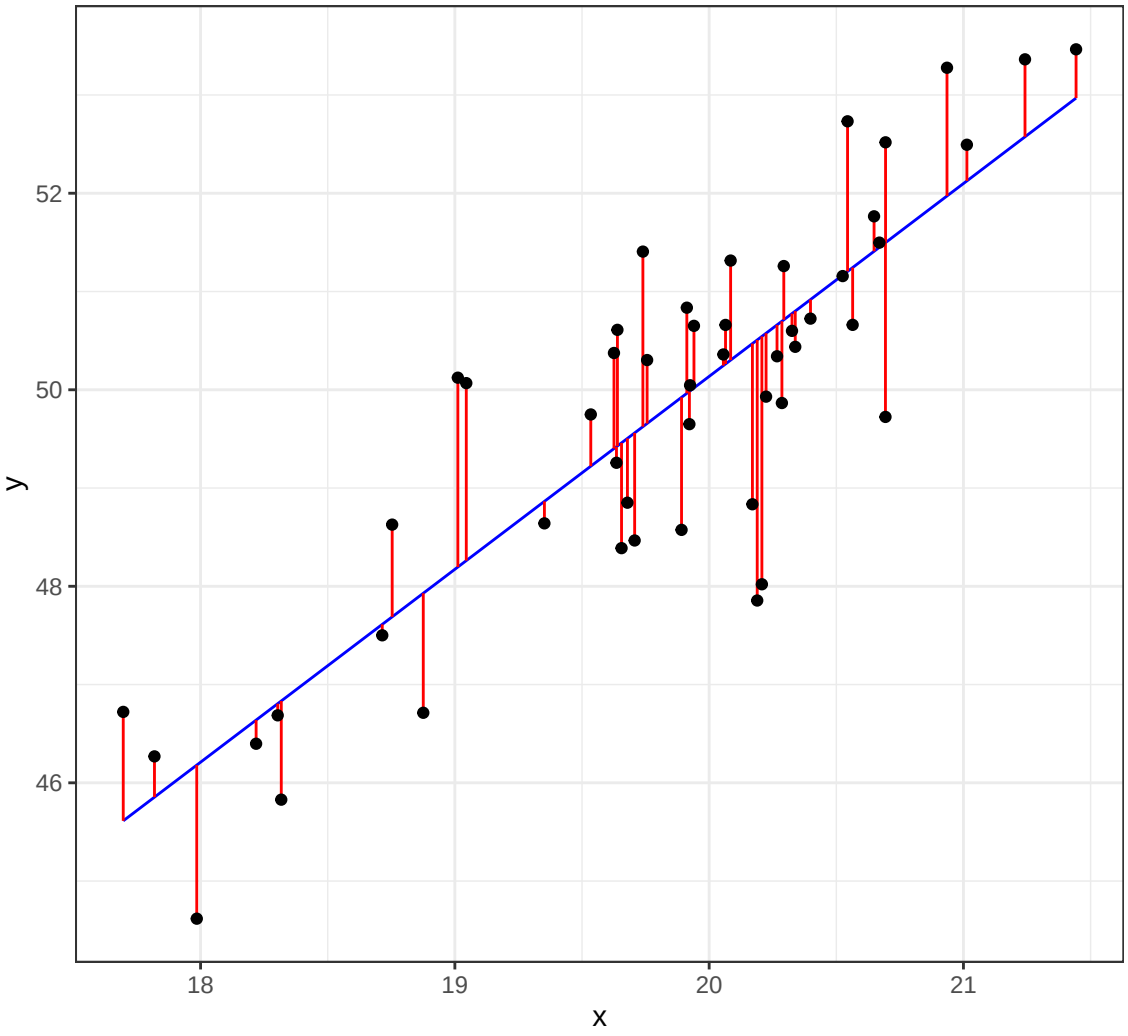
These values are:

$$\hat{\beta}_2 = r_{xy} \frac{s_y}{s_x}$$

$$\hat{\beta}_1 = \overline{y} - \hat{\beta}_2 \overline{x}$$

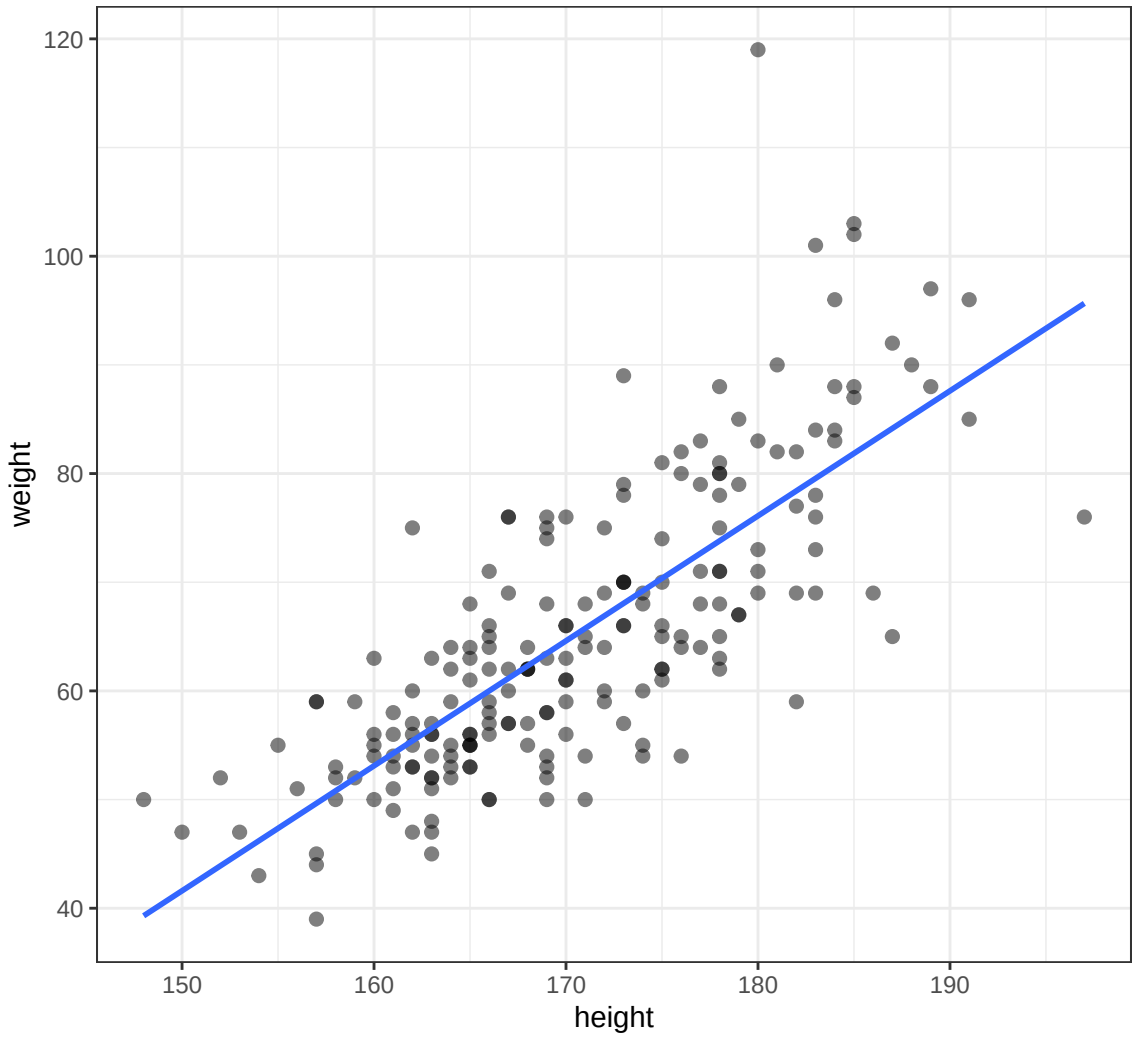
These values have a useful interpretation.

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)
```

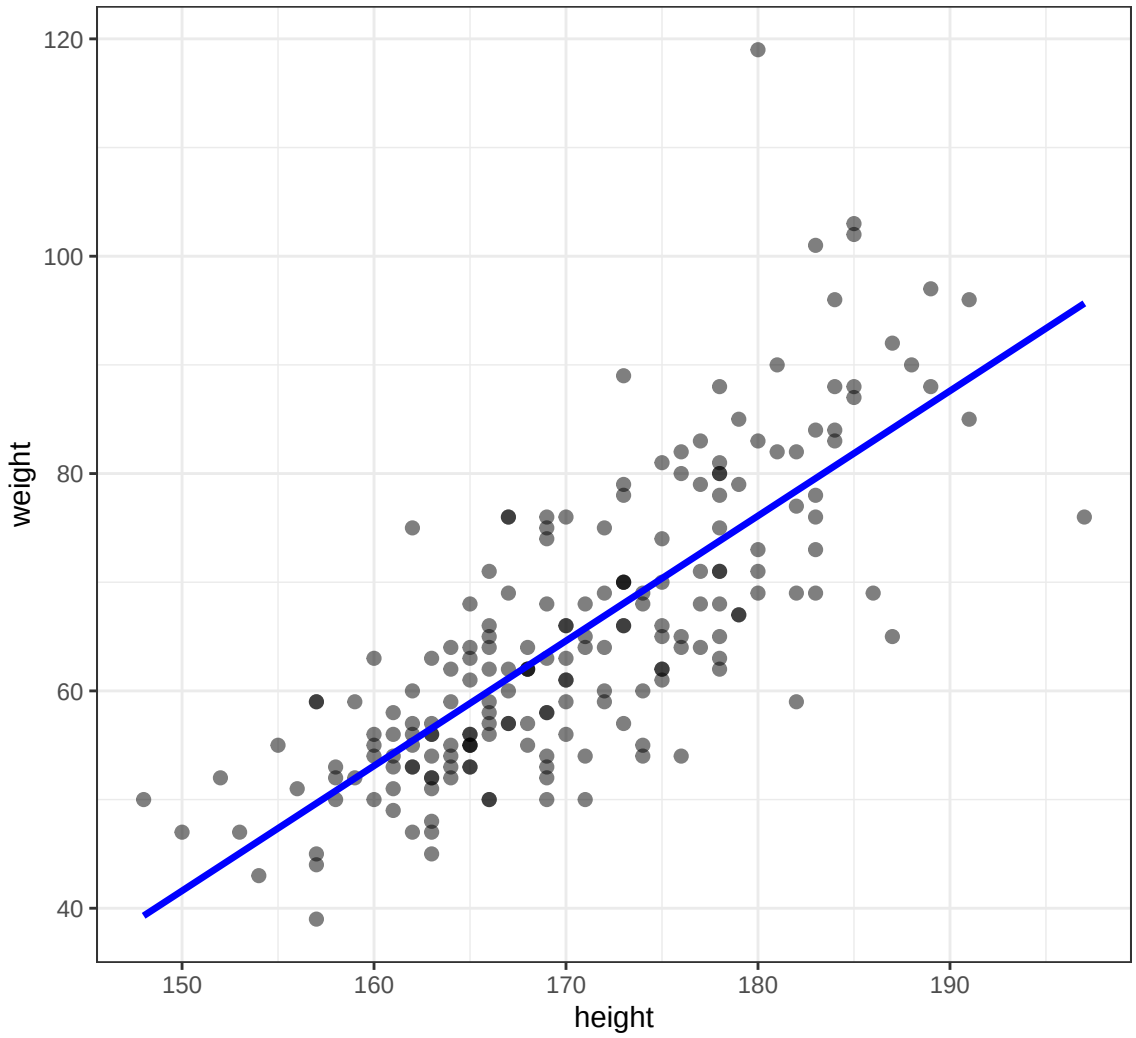


Calculate the Line Directly

```
> beta2 <- cor(htwt$height, htwt$weight) *
+                sd(htwt$weight) / sd(htwt$height)
> beta2
[1] 1.150092
>
> beta1 <- mean(htwt$weight) - beta2 * mean(htwt$height)
> beta1
[1] -130.9104
>
> yhat <- beta1 + beta2 * 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)
```



Observed Data, Fits, and Residuals

We observe data $(x_1, y_1), \ldots, (x_n, y_n)$. Note that we only observe X_i and Y_i from the generative model $Y_i = \beta_1 + \beta_2 X_i + E_i$.

We calculate fitted values and observed residuals:

$$\hat{y}_i = \hat{\beta}_1 + \hat{\beta}_2 x_i$$

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

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

Proportion of Variation Explained

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

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

`lm()` Function in R

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
```

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)
# A tibble: 2 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)  -131.      11.5        -11.4 2.44e-23
2 height          1.15     0.0675      17.0 1.12e-40
```

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
```

Assumptions to Verify

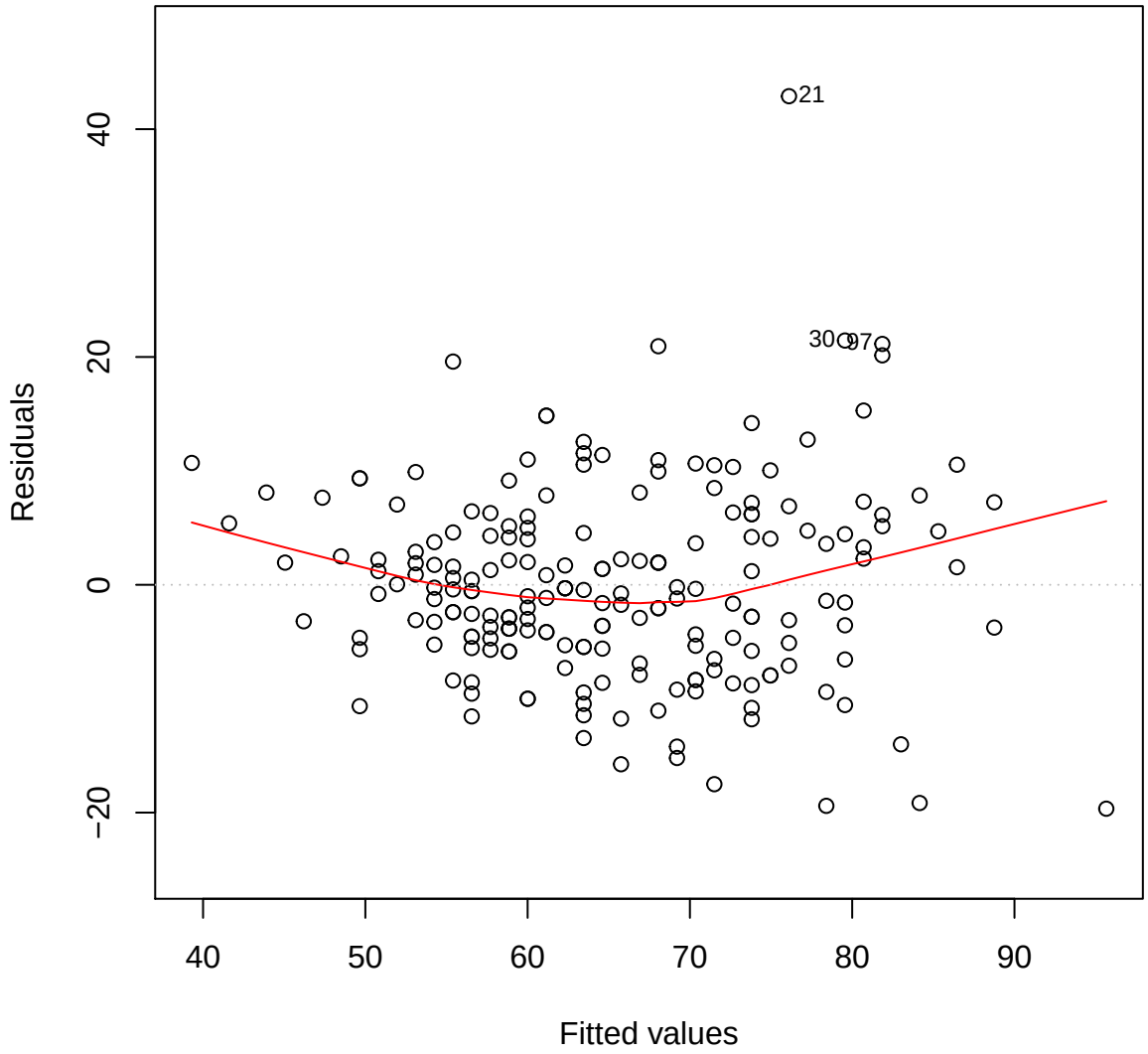
The assumptions on the above linear model are really about the joint distribution of the residuals, which are not directly observed. On data, we try to verify:

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.

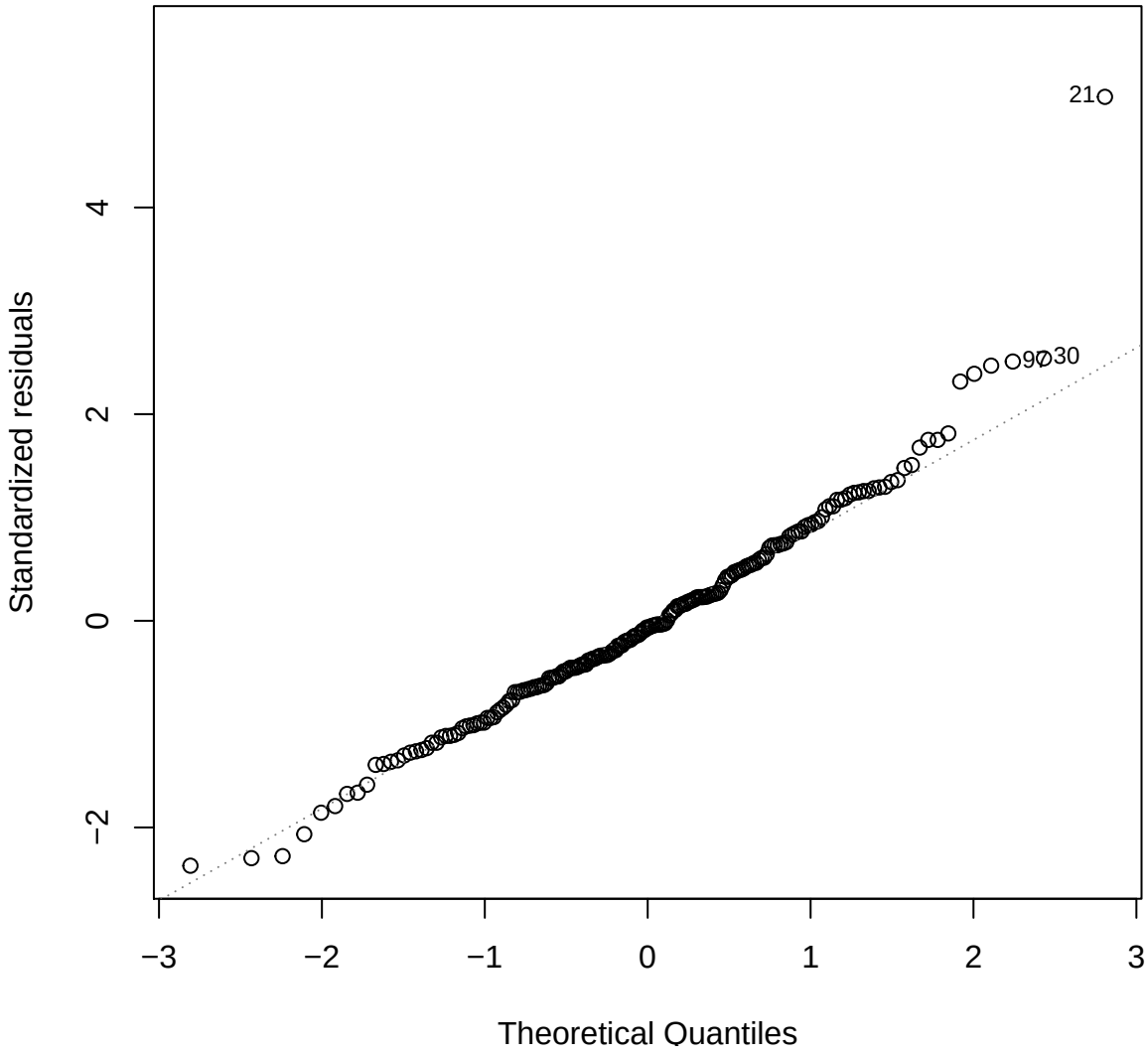
Residual Distribution

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

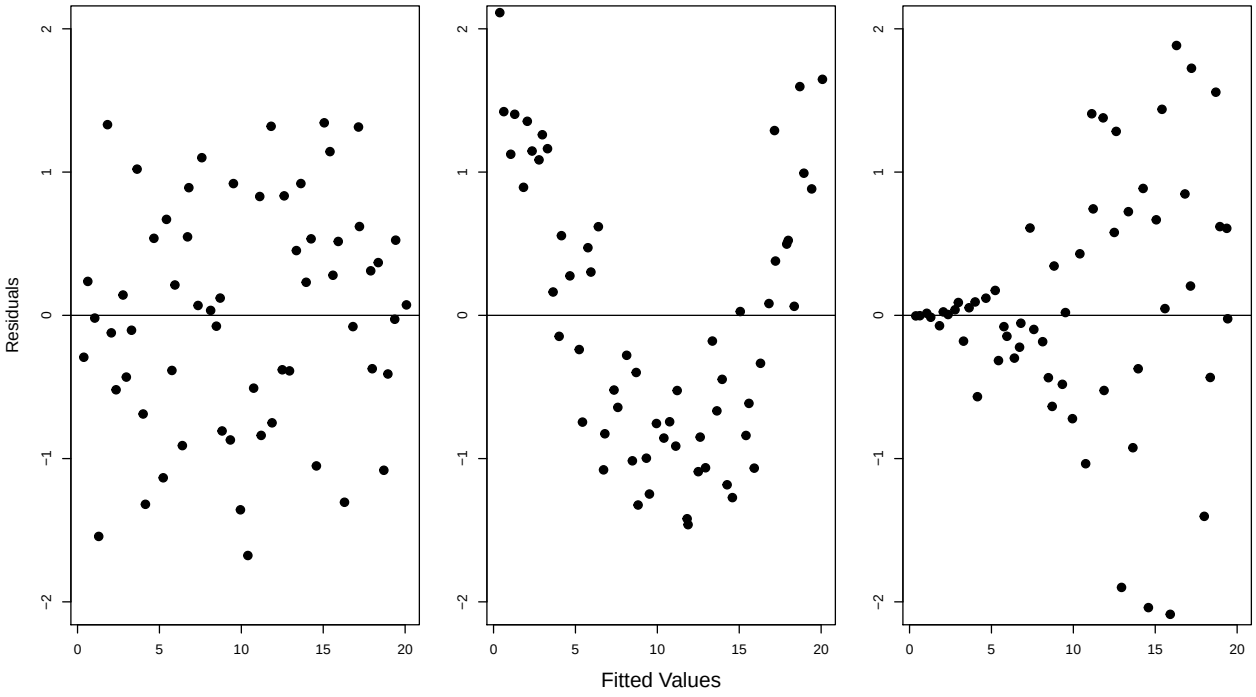


Normal Residuals Check

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


Standardized residuals
Theoretical Quantiles
21
9
30
−3
−2
−1
0
1
2
3
−2
0
2
4

Fitted Values Vs. Obs. Residuals



Ordinary Least Squares

Ordinary least squares (OLS) estimates the model

$$\begin{aligned} Y_i &= \beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip} + E_i \\ &= \boldsymbol{X}_i \boldsymbol{\beta} + E_i \end{aligned}$$

where $\mathrm{E}[E_i] = 0$, $\mathrm{Var}(E_i) = \sigma^2$, and $\mathrm{Cov}(E_i, E_j) = 0$ for all $1 \leq i, j \leq n$ and $i \neq j$.

Note that typically $X_{i1} = 1$ for all i so that $\beta_1 X_{i1} = \beta_1$ serves as the intercept.

OLS Solution

The estimates of $\beta_1, \beta_2, \ldots, \beta_p$ are found by identifying the values that minimize:

$$\begin{aligned} &\sum_{i=1}^{n} [Y_i - (\beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip})]^2 \\ &\qquad = (\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta})^T (\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}) \end{aligned}$$

The solution is expressed in terms of matrix algebra computations:

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{Y}.$$

Sample Variance

Let the predicted values of the model be

$$\hat{\boldsymbol{Y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}.$$

We estimate σ^2 by the OLS sample variance

$$S^2 = \frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{n-p}.$$

Sample Covariance

The p-vector $\hat{\boldsymbol{\beta}}$ has covariance matrix

$$\mathrm{Cov}(\hat{\boldsymbol{\beta}}|\boldsymbol{X}) = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\sigma^2.$$

Its estimated covariance matrix is

$$\widehat{\mathrm{Cov}}(\hat{\boldsymbol{\beta}}) = (\boldsymbol{X}^T\boldsymbol{X})^{-1}S^2.$$

Expected Values

Under the assumption that $\mathrm{E}[E_i] = 0$, $\mathrm{Var}(E_i) = \sigma^2$, and $\mathrm{Cov}(E_i, E_j) = 0$ for all $1 \leq i, j \leq n$ and $i \neq j$, we have the following:

$$\mathrm{E}\left[\hat{\boldsymbol{\beta}}\middle|\,\boldsymbol{X}\right] = \boldsymbol{\beta}$$

$$\mathrm{E}\left[S^2\middle|\,\boldsymbol{X}\right] = \sigma^2$$

$$\mathrm{E}\left[(\boldsymbol{X}^T\boldsymbol{X})^{-1}S^2\middle|\,\boldsymbol{X}\right] = \mathrm{Cov}\left(\hat{\boldsymbol{\beta}}\right)$$

$$\mathrm{Cov}\left(\hat{\beta}_j, Y_i - \hat{Y}_i\right) = \boldsymbol{0}.$$

Standard Error

The standard error of $\hat{\beta}_j$ is the square root of the (j,j) diagonal entry of $(\boldsymbol{X}^T\boldsymbol{X})^{-1}\sigma^2$

$$\mathrm{se}(\hat{\beta}_j) = \sqrt{\left[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\sigma^2\right]_{jj}}$$

and estimated standard error is

$$\hat{\mathrm{se}}(\hat{\beta}_j) = \sqrt{\left[(\boldsymbol{X}^T\boldsymbol{X})^{-1}S^2\right]_{jj}}$$

Proportion of Variance Explained

The proportion of variance explained is defined equivalently to the simple linear regression scneario:

$$R^2 = \frac{\sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^2}{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}.$$

Normal Errors

Suppose we assume $E_1, E_2, \ldots, E_n \overset{\text{iid}}{\sim} \text{Normal}(0, \sigma^2)$. Then

$$\ell\left(\boldsymbol{\beta}, \sigma^2; \boldsymbol{Y}, \boldsymbol{X}\right) \propto -n\log(\sigma^2) - \frac{1}{\sigma^2}(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}).$$

Since minimizing $(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta})$ maximizes the likelihood with respect to $\boldsymbol{\beta}$, this implies $\hat{\boldsymbol{\beta}}$ is the MLE for $\boldsymbol{\beta}$.

It can also be calculated that $\frac{n-p}{n}S^2$ is the MLE for σ^2.

Sampling Distribution

When $E_1, E_2, \ldots, E_n \overset{\text{iid}}{\sim} \text{Normal}(0, \sigma^2)$, it follows that, conditional on $\boldsymbol{X}$:

$$\hat{\boldsymbol{\beta}} \sim \text{MVN}_p\left(\boldsymbol{\beta}, (\boldsymbol{X}^T\boldsymbol{X})^{-1}\sigma^2\right)$$

$$S^2\frac{n-p}{\sigma^2} \sim \chi^2_{n-p}$$
$$\frac{\hat{\beta}_j - \beta_j}{\hat{\text{se}}(\hat{\beta}_j)} \sim t_{n-p}$$

CLT

Under the assumption that $\text{E}[E_i] = 0$, $\text{Var}(E_i) = \sigma^2$, and $\text{Cov}(E_i, E_j) = 0$ for $i \neq j$, it follows that as $n \to \infty$,

$$\sqrt{n}\left(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}\right) \xrightarrow{D} \text{MVN}_p\left(\boldsymbol{0}, (\boldsymbol{X}^T\boldsymbol{X})^{-1}\sigma^2\right).$$

Gauss-Markov Theorem

Under the assumption that $\text{E}[E_i] = 0$, $\text{Var}(E_i) = \sigma^2$, and $\text{Cov}(E_i, E_j) = 0$ for $i \neq j$, the Gauss-Markov theorem shows that among all BLUEs, **best linear unbiased estimators**, the least squares estimate has the smallest mean-squared error.

Specifically, suppose that $\tilde{\boldsymbol{\beta}}$ is a linear estimator (calculated from a linear operator on $\boldsymbol{Y}$) where $\text{E}[\tilde{\boldsymbol{\beta}}|\boldsymbol{X}] = \boldsymbol{\beta}$. Then

$$\text{E}\left[(\boldsymbol{Y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^T(\boldsymbol{Y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})\middle|\boldsymbol{X}\right] \leq \text{E}\left[(\boldsymbol{Y} - \boldsymbol{X}\tilde{\boldsymbol{\beta}})^T(\boldsymbol{Y} - \boldsymbol{X}\tilde{\boldsymbol{\beta}})\middle|\boldsymbol{X}\right].$$

Generalized Least Squares

Generalized least squares (GLS) assumes the same model as OLS, except it allows for **heteroskedasticity** and **covariance** among the E_i. Specifically, it is assumed that $\boldsymbol{E} = (E_1, \ldots, E_n)^T$ is distributed as

$$\boldsymbol{E}_{n\times 1} \sim (\boldsymbol{0}, \boldsymbol{\Sigma})$$

where $\boldsymbol{0}$ is the expected value $\boldsymbol{\Sigma} = (\sigma_{ij})$ is the $n \times n$ covariance matrix.

The most straightforward way to navigate GLS results is to recognize that

$$\boldsymbol{\Sigma}^{-1/2}\boldsymbol{Y} = \boldsymbol{\Sigma}^{-1/2}\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\Sigma}^{-1/2}\boldsymbol{E}$$

satisfies the assumptions of the OLS model.

GLS Solution

The solution to minimizing

$$(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta})$$

is

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{X} \right)^{-1} \boldsymbol{X}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{Y}.$$

Other Results

The issue of estimating $\boldsymbol{\Sigma}$ if it is unknown is complicated. Other than estimates of σ^2, the results from the OLS section recapitulate by replacing $\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{E}$ with

$$\boldsymbol{\Sigma}^{-1/2} \boldsymbol{Y} = \boldsymbol{\Sigma}^{-1/2} \boldsymbol{X} \boldsymbol{\beta} + \boldsymbol{\Sigma}^{-1/2} \boldsymbol{E}.$$

For example, as $n \to \infty$,

$$\sqrt{n} \left(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta} \right) \xrightarrow{D} \text{MNV}_p \left(\mathbf{0}, (\boldsymbol{X}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{X})^{-1} \right).$$

We also still have that

$$\text{E} \left[\hat{\boldsymbol{\beta}} \middle| \boldsymbol{X} \right] = \boldsymbol{\beta}.$$

And when $\boldsymbol{E} \sim \text{MVN}_n(\mathbf{0}, \boldsymbol{\Sigma})$, $\hat{\boldsymbol{\beta}}$ is the MLE.

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
```

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
```

One Variable, Two Scales

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)
```

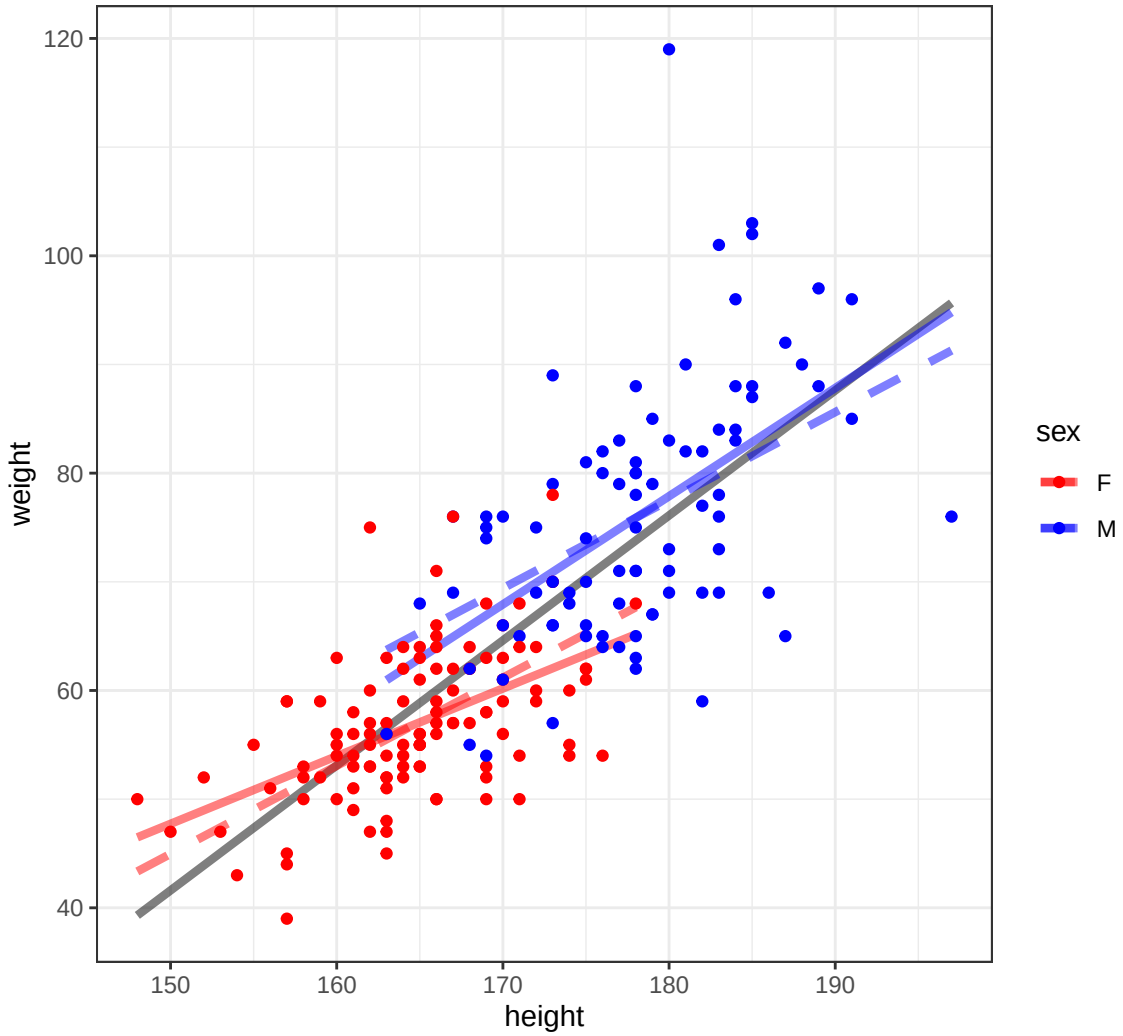
```
Residuals:
    Min      1Q  Median      3Q     Max
-20.869  -4.835  -0.897   4.429  41.122

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



Categorical Explanatory Variables

Example: Chicken Weights

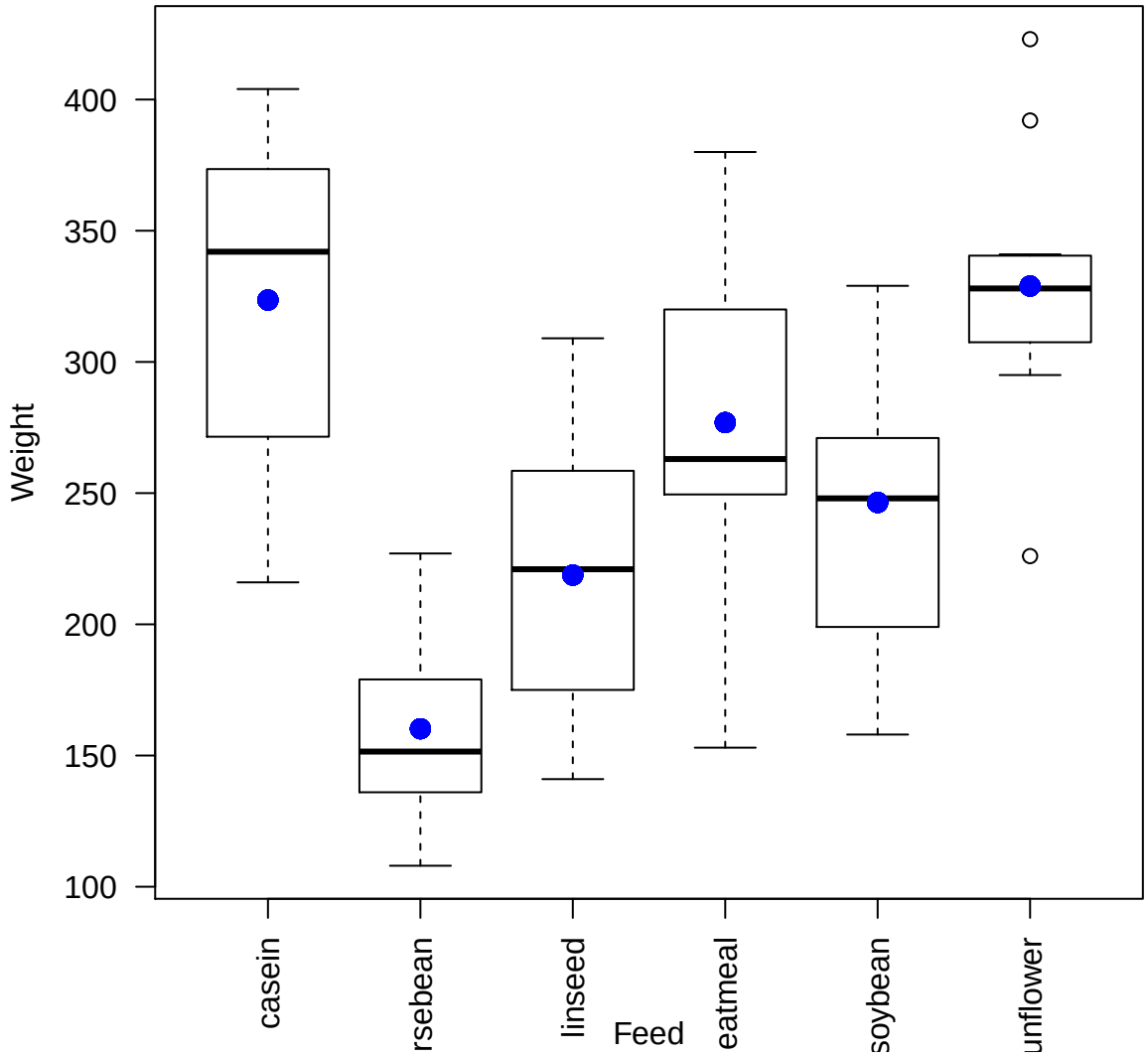
```
> data("chickwts", package="datasets")
> head(chickwts)
  weight      feed
1    179 horsebean
2    160 horsebean
3    136 horsebean
4    227 horsebean
5    217 horsebean
6    168 horsebean
> summary(chickwts$feed)
   casein horsebean   linseed  meatmeal   soybean sunflower
       12        10   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

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", las=2)
> 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))
# A tibble: 6 x 2
  feed      `mean(weight)`
  <fct>              <dbl>
1 casein              324.
2 horsebean           160.
3 linseed             219.
4 meatmeal            277.
5 soybean             246.
6 sunflower           329.
```

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).

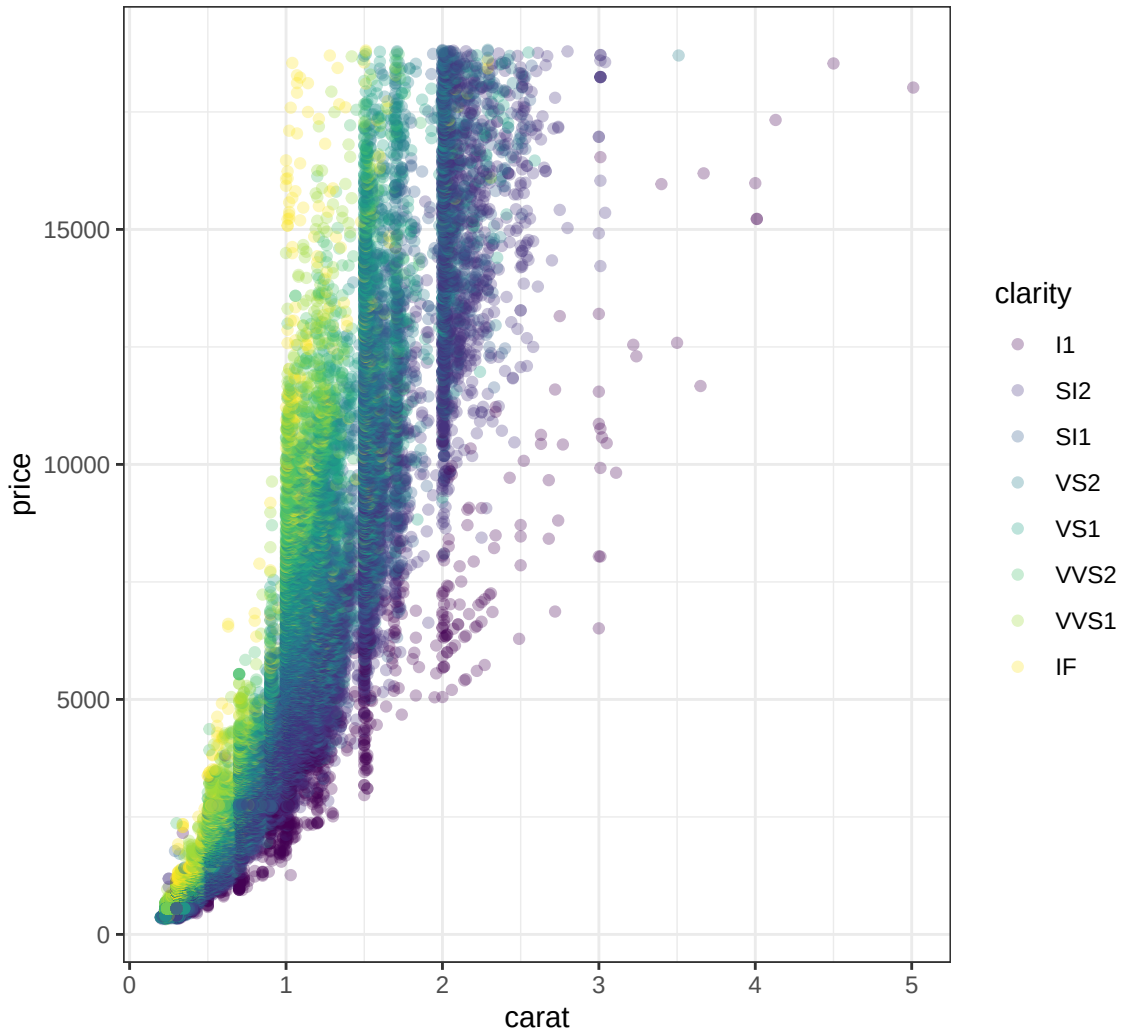
You can read more about the Box-Cox family of power transformations.

Diamonds Data

```
> data("diamonds", package="ggplot2")
> head(diamonds)
# A tibble: 6 x 10
  carat cut   color clarity depth table price     x     y     z
  <dbl> <ord> <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
1 0.23  Ideal E     SI2      61.5    55   326  3.95  3.98  2.43
2 0.21  Prem~ E     SI1      59.8    61   326  3.89  3.84  2.31
3 0.23  Good  E     VS1      56.9    65   327  4.05  4.07  2.31
4 0.290 Prem~ I     VS2      62.4    58   334  4.2   4.23  2.63
5 0.31  Good  J     SI2      63.3    58   335  4.34  4.35  2.75
6 0.24  Very~ J     VVS2     62.8    57   336  3.94  3.96  2.48
```

Nonlinear Relationship

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

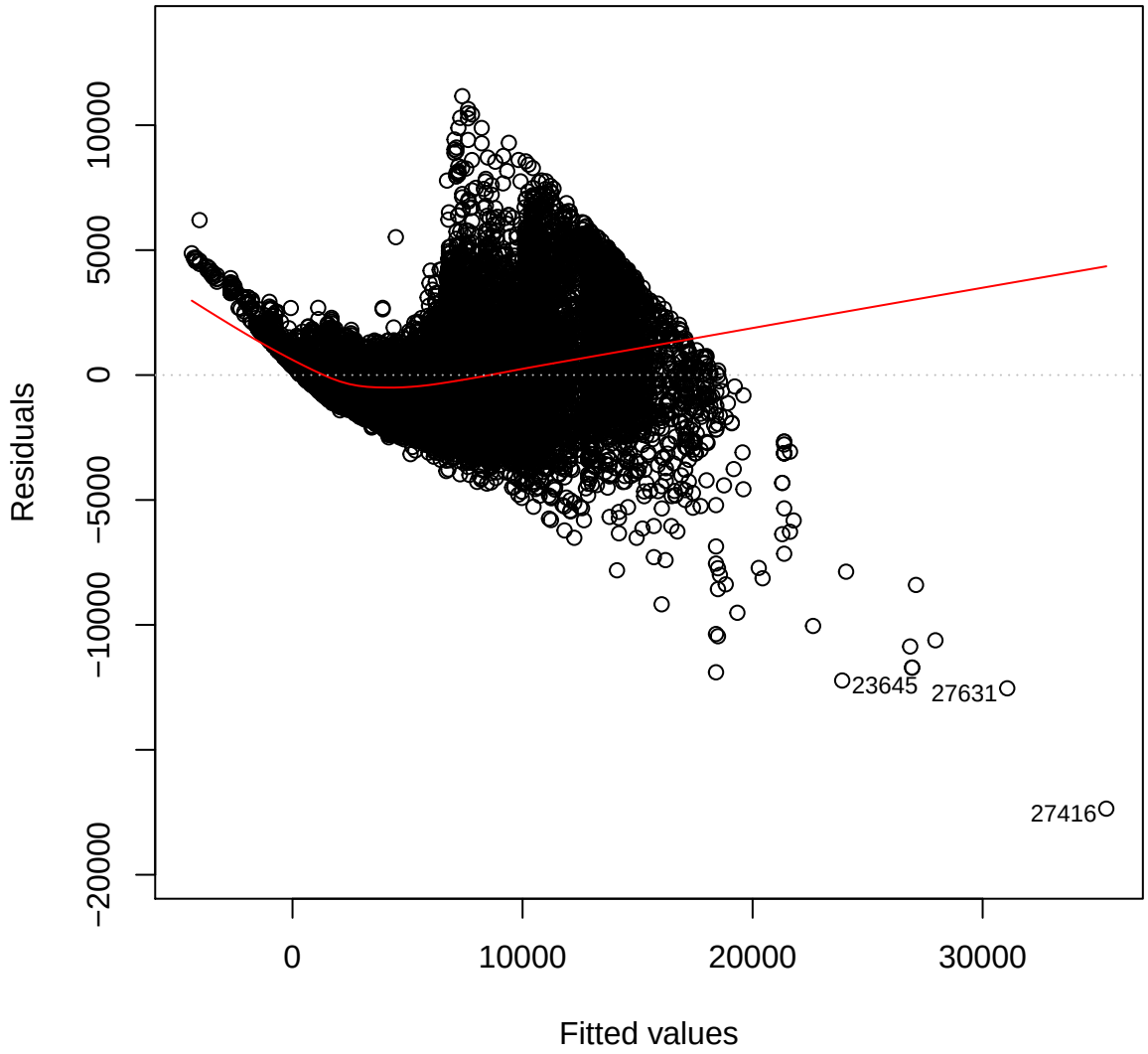


Regression with Nonlinear Relationship

```
> 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
```

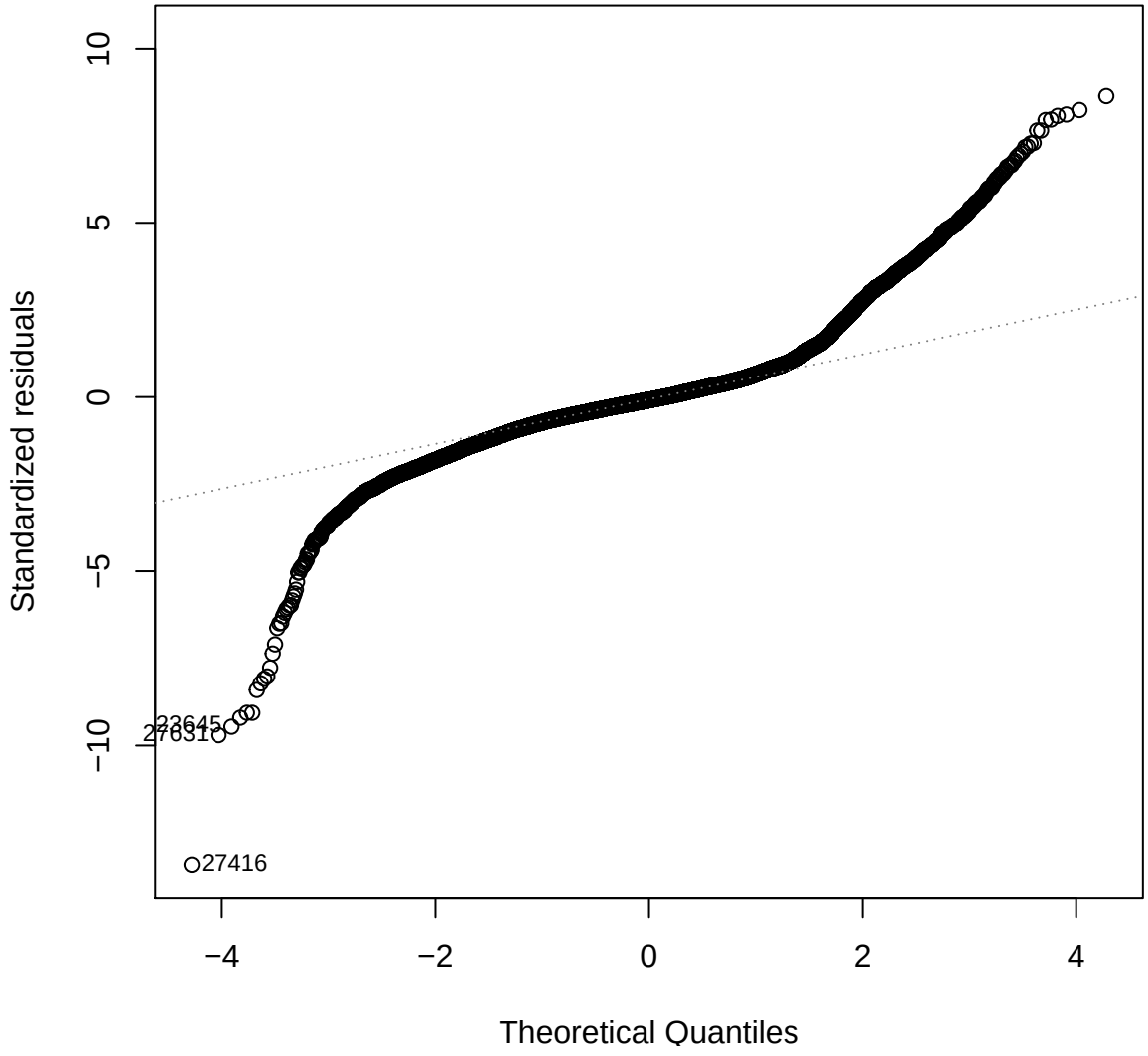
Residual Distribution

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



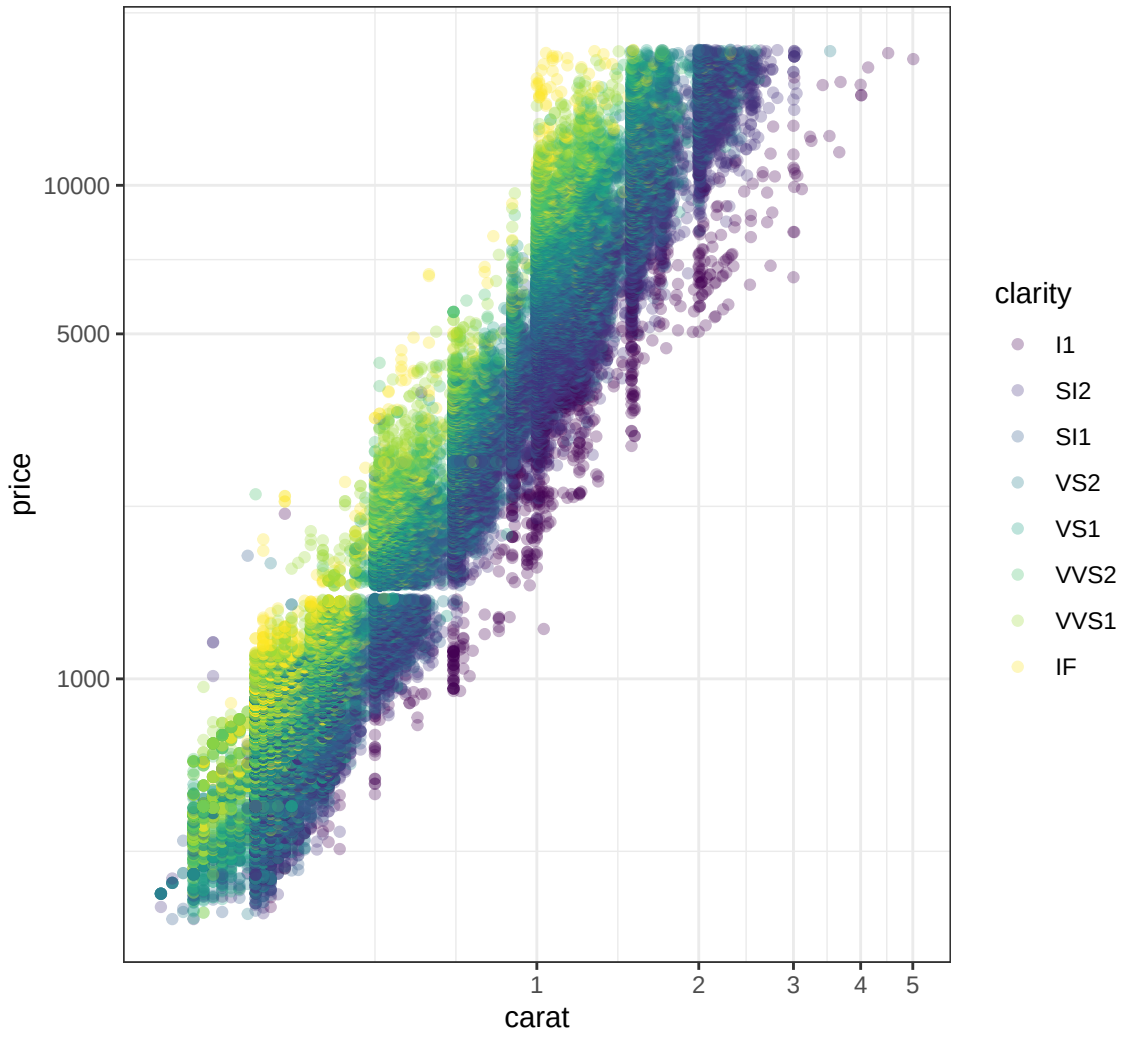
Normal Residuals Check

```
> 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)
```

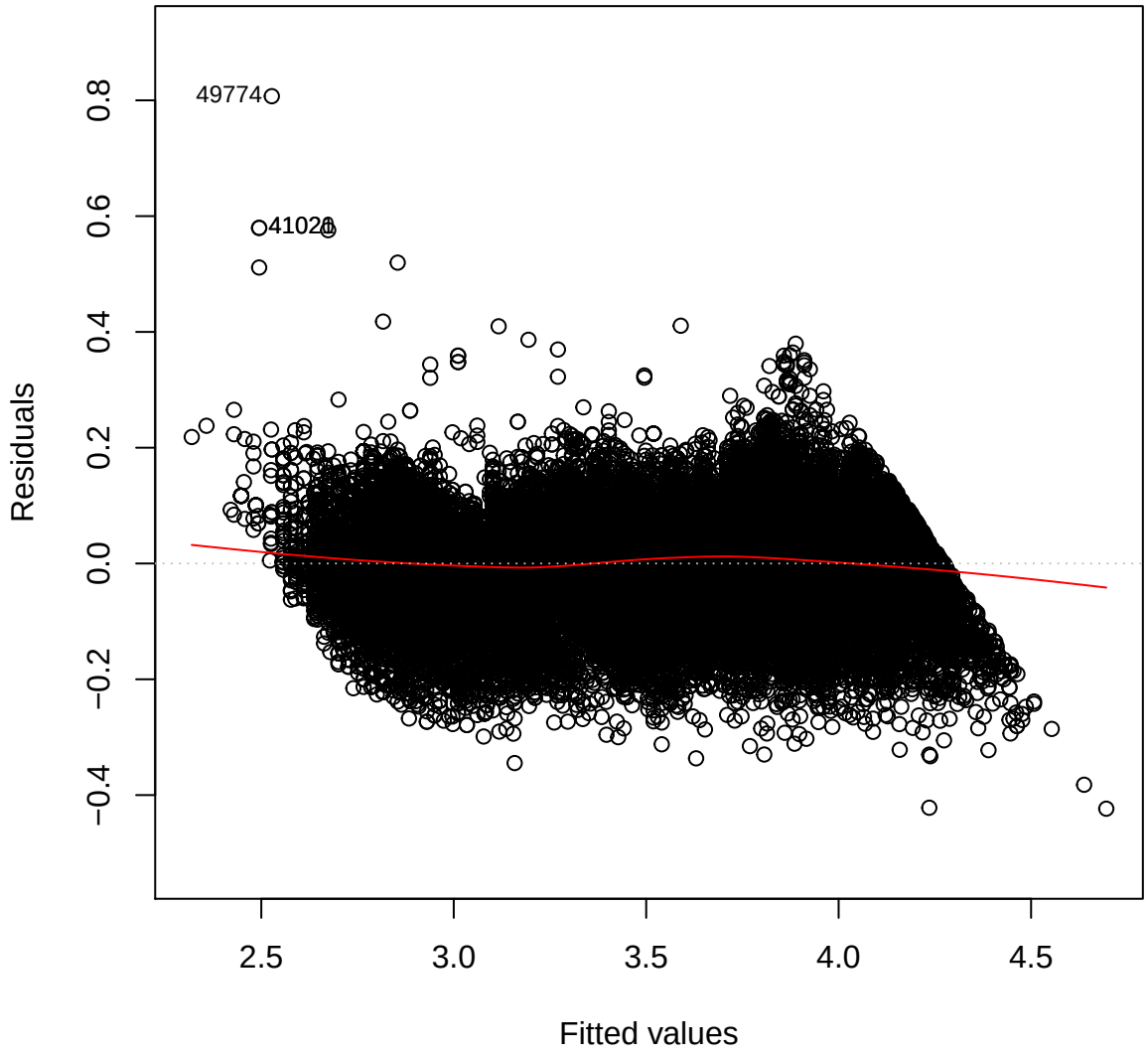


OLS on Log-Transformed Data

```
> 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
```

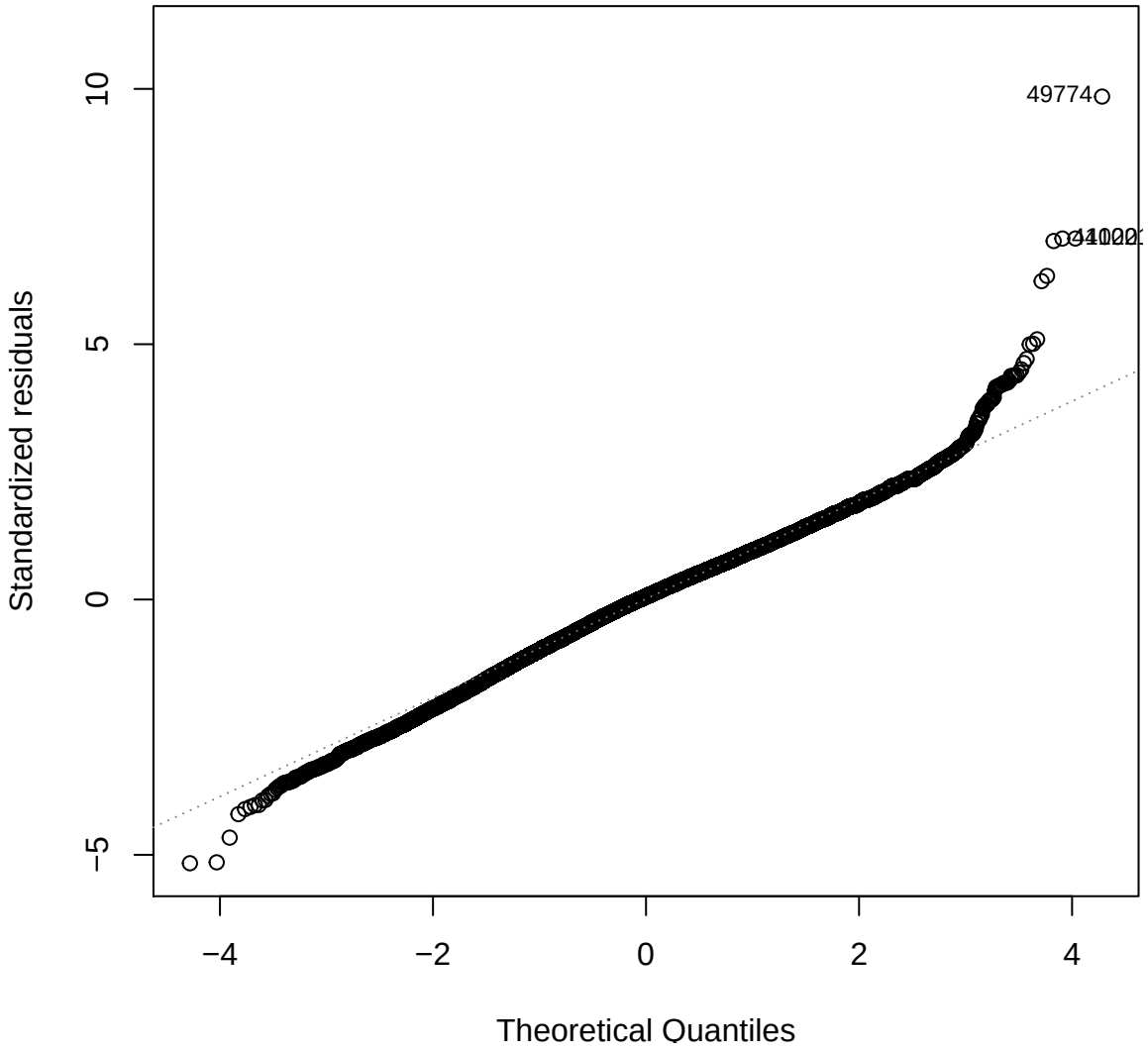
Residual Distribution

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



Normal Residuals Check

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



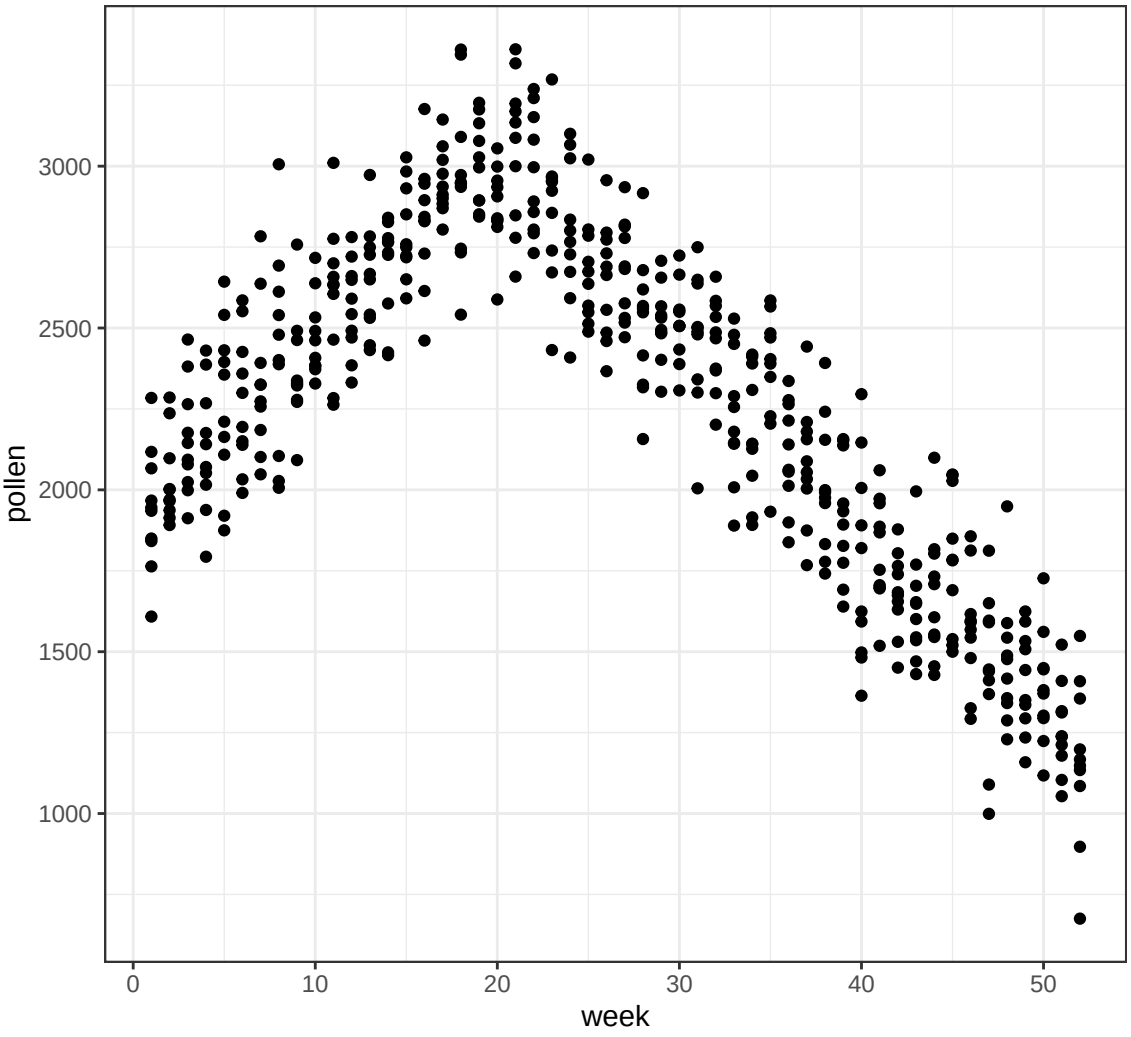
Tree Pollen Study

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

```
> pollen_study
# A tibble: 520 x 3
    week  year pollen
   <int> <int>  <dbl>
 1     1  2001  1842.
 2     2  2001  1966.
 3     3  2001  2381.
 4     4  2001  2141.
 5     5  2001  2210.
 6     6  2001  2585.
 7     7  2001  2392.
 8     8  2001  2105.
 9     9  2001  2278.
10    10  2001  2384.
# ... with 510 more rows
```

Tree Pollen Count by Week

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



A Clever Transformation

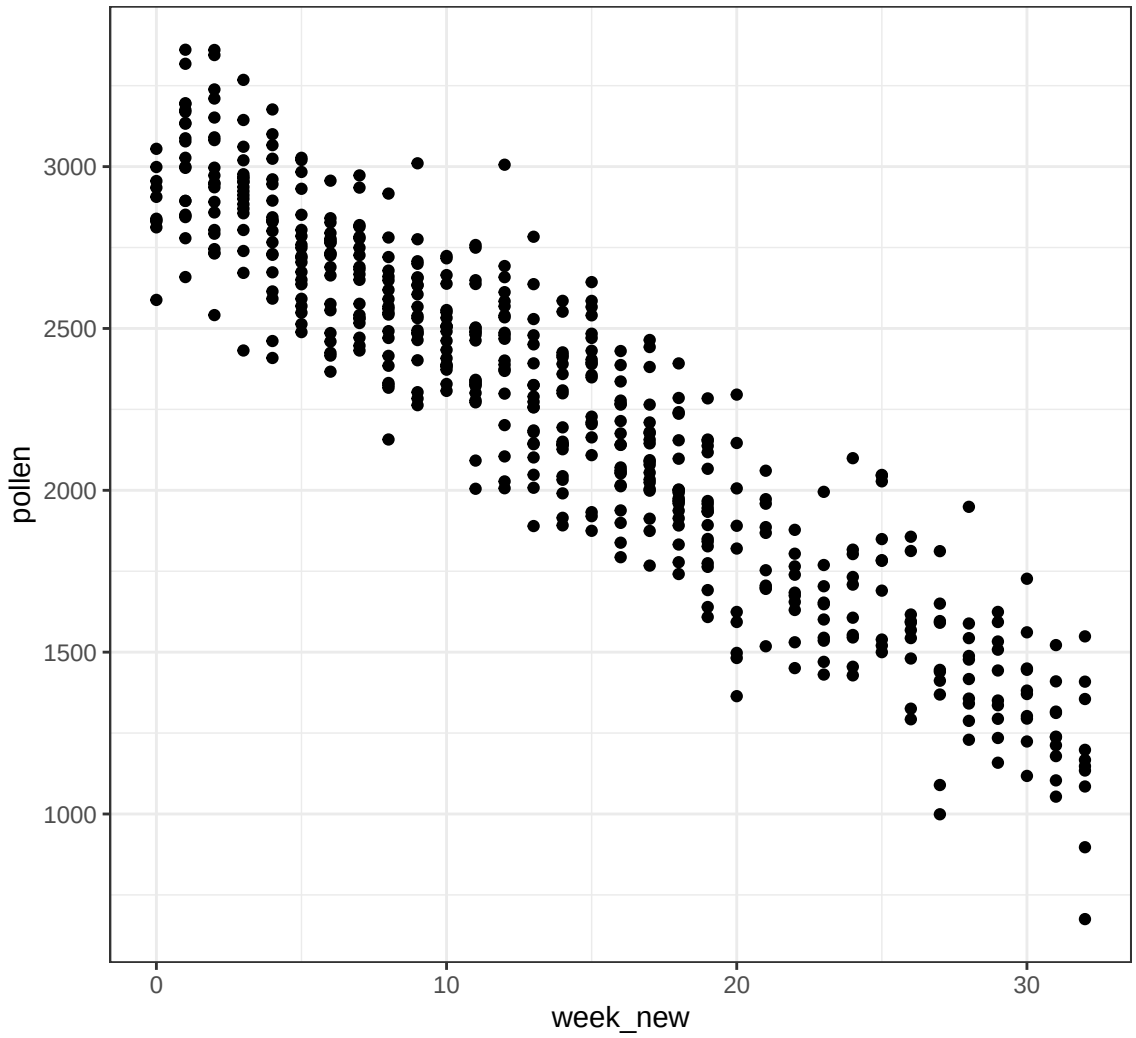
We can see there is a linear relationship between `pollen` and `week` if we transform `week` to be number of weeks from the peak week.

```
> 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

Source

License

Source Code

Session Information

```
> sessionInfo()
R version 3.6.0 (2019-04-26)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: macOS  10.15.3

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib

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] broom_0.5.2     car_3.0-6       carData_3.0-3
 [4] MASS_7.3-51.5   forcats_0.5.0   stringr_1.4.0
 [7] dplyr_0.8.4     purrr_0.3.3     readr_1.3.1
[10] tidyr_1.0.2     tibble_2.1.3    ggplot2_3.2.1
[13] tidyverse_1.3.0 knitr_1.28

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.3         lubridate_1.7.4   lattice_0.20-40
 [4] utf8_1.1.4         assertthat_0.2.1  digest_0.6.25
 [7] R6_2.4.1           cellranger_1.1.0  backports_1.1.5
[10] reprex_0.3.0       evaluate_0.14     httr_1.4.1
[13] highr_0.8          pillar_1.4.3      rlang_0.4.5
[16] lazyeval_0.2.2     curl_4.3          readxl_1.3.1
[19] rstudioapi_0.11    data.table_1.12.8 rmarkdown_2.1
[22] labeling_0.3       foreign_0.8-75    munsell_0.5.0
[25] compiler_3.6.0     modelr_0.1.6      xfun_0.12
[28] pkgconfig_2.0.3    htmltools_0.4.0   tidyselect_1.0.0
[31] codetools_0.2-16   rio_0.5.16        viridisLite_0.3.0
[34] fansi_0.4.1        crayon_1.3.4      dbplyr_1.4.2
[37] withr_2.1.2        grid_3.6.0        nlme_3.1-144
[40] jsonlite_1.6.1     gtable_0.3.0      lifecycle_0.1.0
[43] DBI_1.1.0          magrittr_1.5      scales_1.1.0
[46] zip_2.0.4          cli_2.0.2         stringi_1.4.6
[49] farver_2.0.3       fs_1.3.1          xml2_1.2.2
[52] generics_0.0.2     vctrs_0.2.3       openxlsx_4.1.4
[55] tools_3.6.0        glue_1.3.1        hms_0.5.3
[58] abind_1.4-5        yaml_2.2.1        colorspace_1.4-1
[61] rvest_0.3.5        haven_2.2.0
```