Robust Statistics

Statistics with R

Basel R Bootcamp

April 2019

Robust statistics

Parametric assumptions

Parametric statistics models and the associated tests come with **many assumptions**.

For the models and tests to be **consistent** and **efficient** those assumptions should hold.

Non-parametric statistics

Non-parametric statistics model the data as coming from **no specific probability distribution**.

Involves **ranking**, **combinatorics**, and/or **bootstrap samples**

from xkcd.com

Regression assumptions

In English...

Are the data appropriate?

- Quantitative or dichotomous (A1)
- Nonzero variance (A2)
- Not too highly correlated (A3)

Is the model appropriate?

- Linear relationship (A4, A5, A7)
- Constant error variance (A6)
- Normally distributed errors (A8)

A1. All *independent* variables ($X_1, X_2, \ldots, X_k$) are *quantitative* or *dichotomous*, and the *dependent* variable, Y, is *quantitative, continuous,* and *unbounded.*[6] Moreover, all variables are measured without error.

A2. All independent variables have *nonzero variance* (i.e., each independent variable has some variation in value).

A3. There is not *perfect multicollinearity* (i.e., there is no exact linear relationship between two or more of the independent variables).

A4. At each set of values for the k independent variables, $(X_{1j}, X_{2j}, \ldots, X_{kj})$, $E(\varepsilon_j | X_{1j}, X_{2j}, \ldots, X_{kj}) = 0$ (i.e., the mean value of the error term is zero).

A5. For each X_i, $COV(X_{ij}, \varepsilon_j) = 0$ (i.e., each independent variable is uncorrelated with the error term).[7]

A6. At each set of values for the k independent variables, $(X_{1j}, X_{2j}, \ldots, X_{kj})$, $VAR(\varepsilon_j | X_{1j}, X_{2j}, \ldots, X_{kj}) = \sigma^2$, where σ^2 is a constant (i.e., the conditional variance of the error term is constant); this is known as the assumption of *homoscedasticity.*

A7. For any two observations, $(X_{1j}, X_{2j}, \ldots, X_{kj})$ and $(X_{1h}, X_{2h}, \ldots, X_{kh})$, $COV(\varepsilon_j, \varepsilon_h) = 0$ (i.e., error terms for different observations are uncorrelated); this assumption is known as a lack of *autocorrelation.*[8]

A8. At each set of values for the k independent variables, ε_j is normally distributed.

from Berry (1993)

Are the data appropriate?

The most serious data problem are predictor correlations. If k-1 predictors can explain predictor k...

> ...perfectly then the regression model cannot be calculated (**singularity**).
>
> ...with high accuracy then the estimates become very vague (**multicollinearity**).

The **variance inflation factor** (VIF) indicates the amount of variance of each predictor explained by the other predictors. The literature recommends retaining predictors with **VIF < 10 (Stine, 1995)**

Residual variance
for y being regressed on all x

VIF
Variance inflation factor

$$\hat{\sigma}^2_{\beta_j} = \frac{s^2_{e|x}}{s^2_{x_j}(n-1)} \frac{1}{1-R^2_{x_j|x}}$$

Predictor variance
Variance of specific predictor j

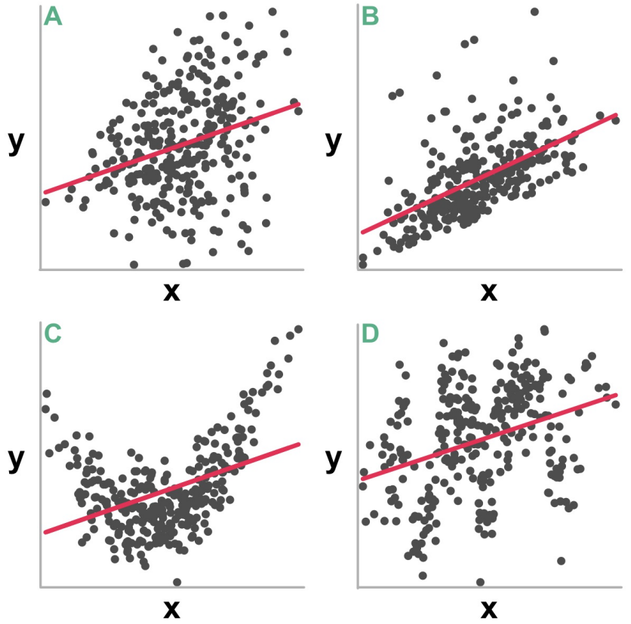
Shared variance
between predictor j and the others

Is the model appropriate?

The appropriateness of the model is best inspected (and demonstrated) using **graphical illustrations.**

Additionally, several **statistics can be computed to support the assessment.**

Assumption	**Statistics**
Linearity	`lm`, `glm` (curve fitting)
Homoscedasticity	`bartlett.test`
Normality	`skewness`, `kurtosis`, `shapiro.test`
Outliers	`cooks.distance`, `mahalanobis`, etc.

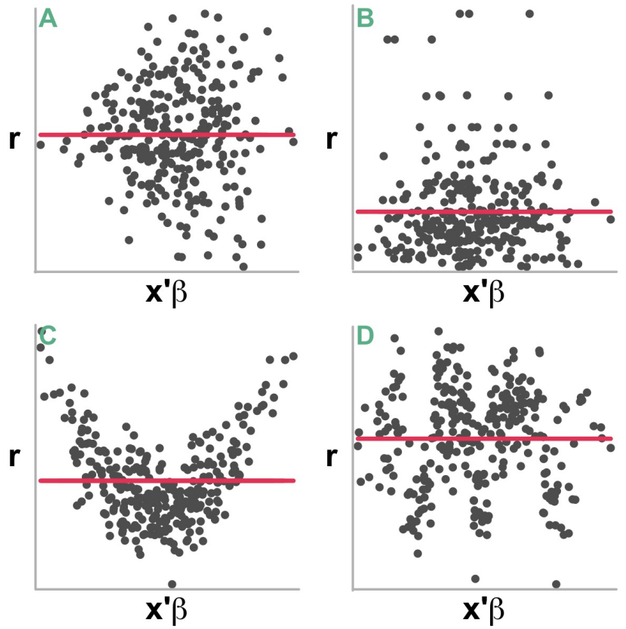


Is the model appropriate?

The appropriateness of the model is best inspected (and demonstrated) using **graphical illustrations.**

Additionally, several **statistics can be computed to support the assessment.**

Assumption	**Statistics**
Linearity	`lm`, `glm` (curve fitting)
Homoscedasticity	`bartlett.test`
Normality	`skewness`, `kurtosis`, `shapiro.test`
Outliers	`cooks.distance`, `mahalanobis`, etc.



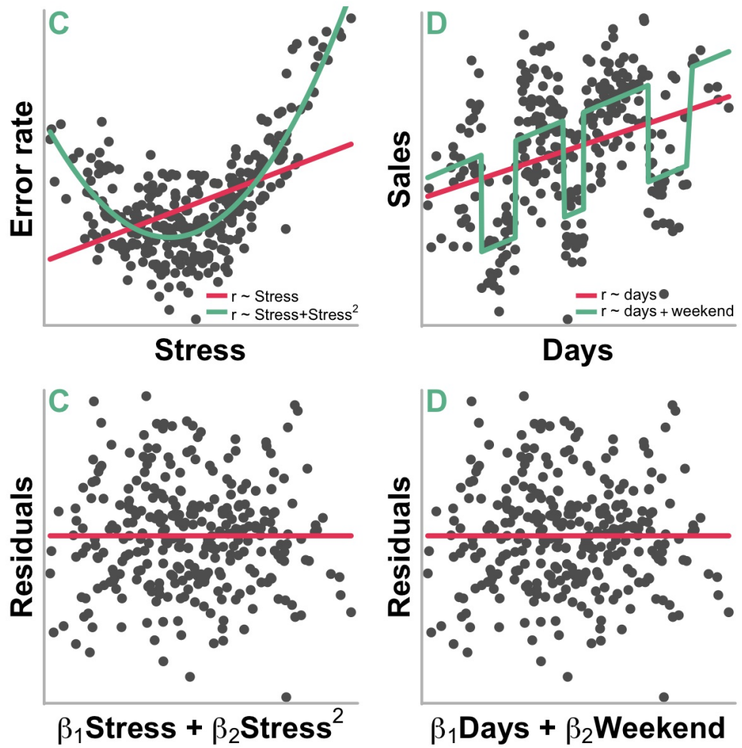
Missing predictors

Most serious are **violations of the linearity assumption**.

Typically they arise from **missing predictors**, such as higher order trends or important temporal variables.

Whether additional predictors improve the fit to the data can be tested by comparing the model with and without the addition predictor using a **ANOVA deviance test**:

```
anova(model_1, model_2).
```

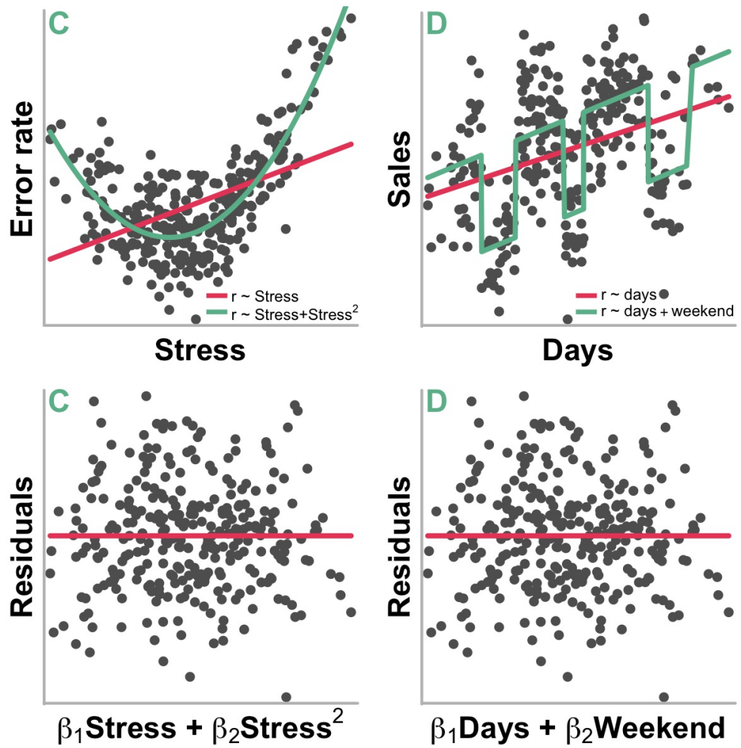


Missing predictors

Testing the merit of including a **quadratic trend of Stress**.

```
# ANOVA deviance test
model_1 <- lm(error_rate ~ stress,
              data = stress_df)
model_2 <- lm(error_rate ~ stress + stress2,
              data = stress_df)
anova(model_1, model_2)
```

```
## Analysis of Variance Table
##
## Model 1: error_rate ~ stress
## Model 2: error_rate ~ stress + stress2
##   Res.Df   RSS Df Sum of Sq   F Pr(>F)
## 1    298 236.5
## 2    297  60.6  1       176 863 <2e-16 ***
## ---
## Signif. codes:
##   0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
##   0.1 ' ' 1
```



Non-parametric statistics

When assumptions don't hold, researchers can make use of a wide array of **more robust**, non-parametric methods.

Approach	Methods
Rank combinatorics	`wilcox.test`, `friedman.test`
Freq. combinatorics	`sign.test`, `chisq.test`
M-estimation	`rq` (quantile regr.), `rfit` (rank regr.)
Bootstrap	`boot` (anything)

robust [roh-**buhst**, **roh**-buhst]

adjective

1. strong and healthy; hardy; vigorous:
   a robust young man; a robust faith; a robust mind.
2. strongly or stoutly built:
   his robust frame.
3. suited to or requiring bodily strength or endurance:
   robust exercise.
4. rough, rude, or boisterous:
   robust drinkers and dancers.
5. rich and full-bodied:
   the robust flavor of freshly brewed coffee.
6. strong and effective in all or most situations and conditions:
   The system requires robust passwords that contain at least one number or symbol.
   Our goal is to devise robust statistical methods.

Wilcoxon test

One classic relying on the combinatorics of ranks is the Wilcoxon or Mann-Whitney U test.

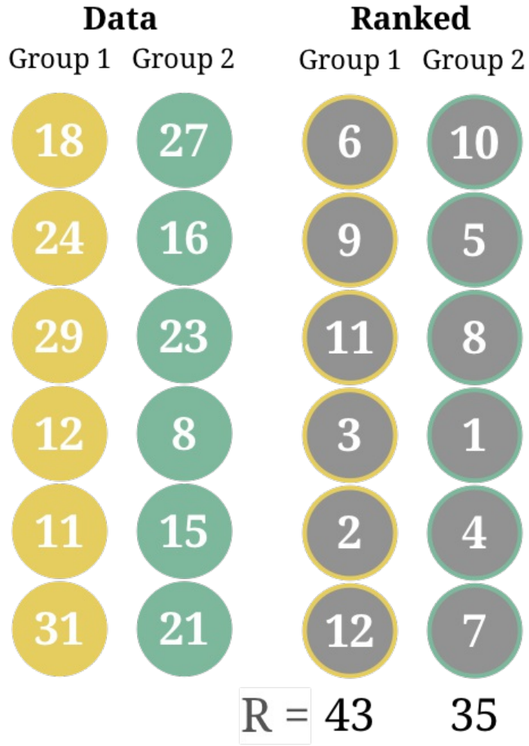
The Wilcoxon test evaluates the assumption that the **ranks sums of jointly ranked groups are identical**. The formula shows the normal approximation for large N.

Test statistic *U*
Rank sum minus expected rank sum

Expected *U*
Expected *U* under H_0

$$z = \frac{R_1 - \frac{n_1(n_1+1)}{2} - \frac{n_1 n_2}{2}}{\sqrt{\frac{n_1 n_2 (n_1+n_2+1)}{12}}}$$

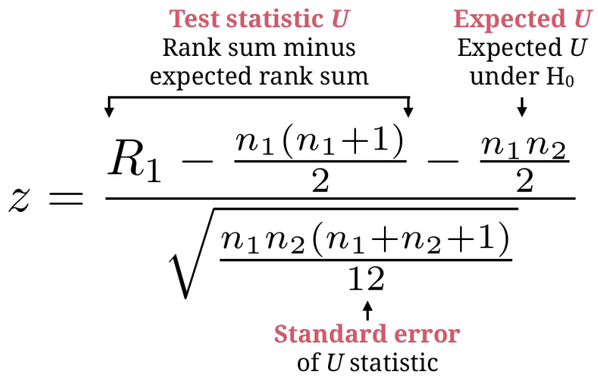
Standard error
of *U* statistic



Wilcoxon test

One classic relying on the combinatorics of ranks is the Wilcoxon or Mann-Whitney U test.

The Wilcoxon test evaluates the assumption that the **ranks sums of jointly ranked groups are identical**. The formula shows the normal approximation for large N.




```
# data
group_1 <- c(18, 24, 29, 12, 11, 31)
group_2 <- c(27, 16, 23, 8, 15, 21)

# wilcoxon test
wilcox.test(group_1, group_2)
```

```
##
##     Wilcoxon rank sum test
##
## data:  group_1 and group_2
## W = 22, p-value = 0.6
## alternative hypothesis: true location shift is not
```

Sign test

A very simple test relying on the combinatorics of frequencies is the **sign test**.

The sign test evaluates the assumption that the **signs of differences (+, -) are equally likely**, i.e., $p_+ = p_- = .5$.

Sum Over all $n \geq n_+$ — **Binomial** distribution for $p = .5$

$$Pr(X \geq n_+) = \sum_{n_+ < n < N} \binom{N}{n} .5^n .5^{N-n}$$

Data		Evaluation	
Time 1	Time 2	Difference	Sign+
18	27	-9	0
24	16	8	1
29	23	6	1
12	8	4	1
11	15	-4	0
31	21	10	1

$\Sigma = 4$

Sign test

A very simple test relying on the combinatorics of frequencies is the **sign test**.

The sign test evaluates the assumption that the **signs of differences (+, -) are equally likely**, i.e., \(p_+ = p_- = .5\).

Sum Over all $n \geq n_+$ — **Binomial** distribution for $p = .5$

$$Pr(X \geq n_+) = \sum_{n_+ < n < N} \binom{N}{n} .5^n .5^{N-n}$$

```
# data
time_1 <- c(18, 24, 29, 12, 11, 31)
time_2 <- c(27, 16, 23, 8, 15, 21)

# sign test
N <- length(time_1)
n_plus <- sum(sign(time_1 - time_2) == 1)
ps <- dbinom(x = n_plus : N,
             size = N,
             prob = .5) # H0 assumption
sum(ps)
```

```
## [1] 0.3438
```

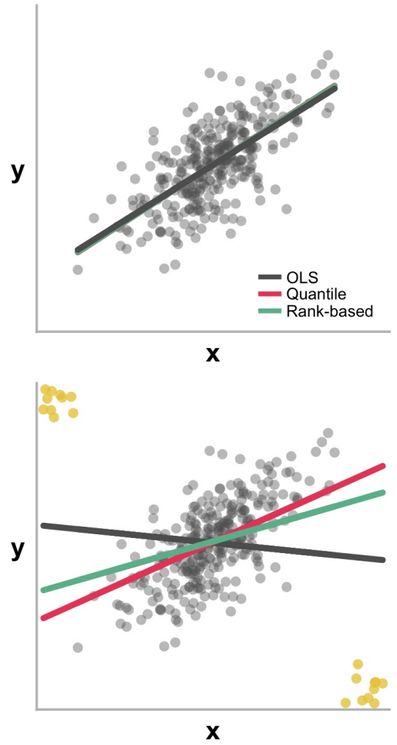
Robust regressions

Various options exist to improve the robustness of regression-based models.

They all **minimize a specific function \(\rho\) of the residuals**:

$$\sum_i \rho(e_i) = \sum_i \rho(y_i - \boldsymbol{x}_i'\boldsymbol{b})$$

Function	Package	Function	Description
`lm`	`stats`	$\rho(e_i) = e_i^2$	Ordinary least square
`rq`	`quantreg`	$\rho(e_i) = e_i(r - I(e_i < 0))$	Quantile regression
`rfit`	`Rfit`	$\rho(e_i) = \sqrt{12}\left(\frac{R_i}{n-1} - .5\right)$	Rank-based regression



Robust regressions

Various options exist to improve the robustness of regression-based models.

They all **minimize a specific function \(\rho\) of the residuals**:

$$\sum_i \rho(e_i) = \sum_i \rho(y_i - \boldsymbol{x}_i'\boldsymbol{b})$$

Function	Package	Function	Description
`lm`	`stats`	$\rho(e_i) = e_i^2$	Ordinary least square
`rq`	`quantreg`	$\rho(e_i) = e_i(r - I(e_i < 0))$	Quantile regression
`rfit`	`Rfit`	$\rho(e_i) = \sqrt{12}\left(\frac{R_i}{n-1} - .5\right)$	Rank-based regression

```
# Quantile regression
library(quantreg)
m <- rq(formula = y ~ x,
        data = outlier_df)
summary(m)
```

```
##
## Call: rq(formula = y ~ x, data = outlier_df)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd
## (Intercept)  0.01626     -0.04042
## x            0.47497      0.30777
##             upper bd
## (Intercept)  0.13365
## x            0.54799
```

Robust regressions

Various options exist to improve the robustness of regression-based models.

They all **minimize a specific function \(\rho\) of the residuals**:

$$\sum_i \rho(e_i) = \sum_i \rho(y_i - \boldsymbol{x}_i'\boldsymbol{b})$$

Function	Package	Function	Description
`lm`	`stats`	$\rho(e_i) = e_i^2$	Ordinary least square
`rq`	`quantreg`	$\rho(e_i) = e_i(r - I(e_i < 0))$	Quantile regression
`rfit`	`Rfit`	$\rho(e_i) = \sqrt{12}\left(\frac{R_i}{n-1} - .5\right)$	Rank-based regression

```
# Rank-based regression
library(Rfit)
m <- rfit(formula = y ~ x,
          data = outlier_df)
summary(m)
```

```
## Call:
## rfit.default(formula = y ~ x, data = outlier_d
##
## Coefficients:
##             Estimate Std. Error
## (Intercept)   0.0328     0.0551
## x             0.3047     0.0417
##             t.value p.value
## (Intercept)    0.60    0.55
## x              7.31 2.2e-12 ***
## ---
## Signif. codes:
##   0 '***' 0.001 '**' 0.01 '*'
##   0.05 '.' 0.1 ' ' 1
## Overall Wald Test: 53.41 p-value: 0
```


Bootstrapping

Bootstrapping refers to statistical tests based on **sampling with replacement from the observed data**.

The core idea is that the distribution of the sample statistic in question, e.g., the mean of a difference or a regression weight, can be **simulated directly from the data without making distributional assumptions**.

The term bootstrapping stems from the saying **pull oneself up by one's bootstraps**, which is sometimes attributed to Baron Münchhausen's story of pulling himself and his horse out of the swamp by his pigtail.

Baron Münchhausen, from wikipedia.org

Bootstrapping

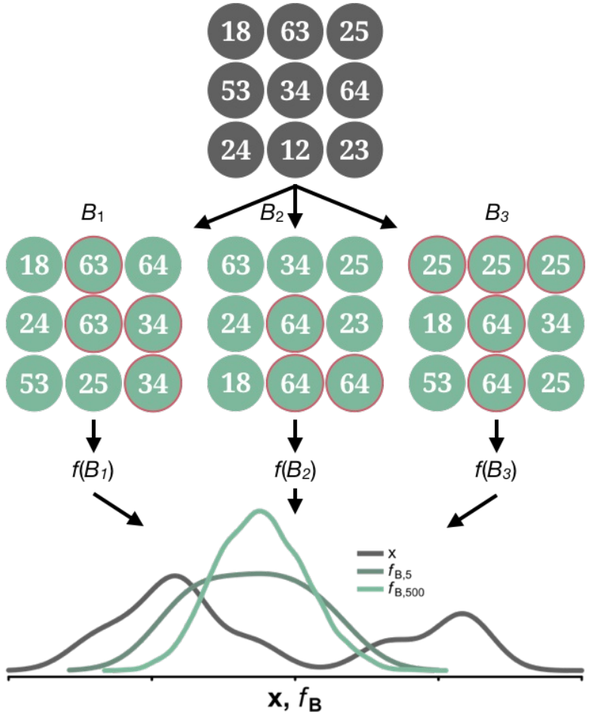
Bootstrapping can be used to obtain sampling distributions for any statistic.

Step 1
Take R **bootstrap samples** B_i of n observations, typically one row in our data.

Step 2
Calculate statistic $f(B_i)$, e.g., the difference in means or regression coefficients, from each of the R bootstrap samples.

Step 3
Evaluate the simulated statistics, e.g., via a confidence intervals.

$$\bar{f}_{B_i} \pm z_{1-\frac{\alpha}{2}} \sigma_{f_{B_i}}$$



Bootstrapping

Bootstrapping can be used to obtain sampling distributions for any statistic.

Step 1
Take R **bootstrap samples** B_i of n observations, typically one row in our data.

Step 2
Calculate statistic $f(B_i)$, e.g., the difference in means or regression coefficients, from each of the R bootstrap samples.

Step 3
Evaluate the simulated statistics, e.g., via a confidence intervals.

$$\bar{f}_{B_i} \pm z_{1-\frac{\alpha}{2}} \sigma_{f_{B_i}}$$

```r
# Bootstrap
library(boot)

# bootstrap function
stat_fun <- function(data, indices){
  data <- data[indices,] # bootstrap
  m <- lm(error_rate ~ stress + stress2,
          data = data)
  coefficients(m)
  }

# bootstrap samples
B <- boot(stress_df,
          statistic = stat_fun,
          R = 1000)
```

Bootstrapping

Bootstrapping can be used to obtain sampling distributions for any statistic.

Step 1
Take \(R\) **bootstrap samples** \(B_i\) of \(n\) observations, typically one row in our data.

Step 2
Calculate statistic \(f(B_i)\), e.g., the difference in means or regression coefficients, from each of the \(R\) bootstrap samples.

Step 3
Evaluate the simulated statistics, e.g., via a confidence intervals.

$$\bar{f}_{B_i} \pm z_{1-\frac{\alpha}{2}} \sigma_{f_{B_i}}$$

```
# Bootstrap CI for stress
boot.ci(B, index = 2)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = B, index = 2)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 0.1517,  0.2907 )   ( 0.1412,  0.2829 )
##
## Level     Percentile            BCa
## 95%   ( 0.1671,  0.3088 )   ( 0.1582,  0.2926 )
## Calculations and Intervals on Original Scale
```

Bootstrapping

Bootstrapping can be used to obtain sampling distributions for any statistic.

Step 1
Take \(R\) **bootstrap samples** \(B_i\) of \(n\) observations, typically one row in our data.

Step 2
Calculate statistic \(f(B_i)\), e.g., the difference in means or regression coefficients, from each of the \(R\) bootstrap samples.

Step 3
Evaluate the simulated statistics, e.g., via a confidence intervals.

$$\bar{f}_{B_i} \pm z_{1-\frac{\alpha}{2}} \sigma_{f_{B_i}}$$

```
# Bootstrap CI for stress2
boot.ci(B, index = 3)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = B, index = 3)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 0.6600,  0.7900 )   ( 0.6611,  0.7965 )
##
## Level     Percentile            BCa
## 95%   ( 0.6495,  0.7850 )   ( 0.6547,  0.7917 )
## Calculations and Intervals on Original Scale
```

Practical