

Introduction to Resampling Methods Using R

Contents

- 1 Sampling from known distributions and simulation
 - 1.1 Sampling from normal distributions
 - 1.2 Specifying seeds
 - 1.3 Sampling from exponential distributions
- 2 Bootstrapping
 - 2.1 Bootstrap distributions
 - 2.2 Bootstrap confidence intervals
 - 2.2.1 Percentile method
 - 2.2.2 Pivot method
 - 2.2.3 Standard bootstrap
- 3 Randomization tests
 - 3.1 Creating random permutations
 - 3.2 Comparing groups
 - 3.2.1 Exact randomization distribution
 - 3.2.2 Random sampling the randomization distribution
 - 3.2.3 Choice of test statistic
 - 3.3 Wilcoxon rank-sum test
 - 3.4 Selecting among two-sample tests
 - 3.5 More than two groups
 - 3.6 Contingency tables
- 4 Methods for correlation and regression
 - 4.1 Randomization test for linear relation
 - 4.1.1 Pearson correlation or slope of regression line
 - 4.1.2 Rank correlation
 - 4.2 Bootstrap intervals for correlation and slope
 - 4.2.1 Bivariate bootstrap sampling
 - 4.2.2 Confidence intervals
 - 4.2.3 *Fixed-X sampling for the slope*
- 5 *Two-sample bootstrap intervals*

1. Sampling from known distributions and simulation

In introductory statistics courses we are told that the t-test is “robust” to departures from normality, especially if the sample size is large. What this means is if we specify a particular Type I error rate, then the actual proportion of false rejections will be close to the Type I error rate. Let’s create and run a simulation to explore this.

Steps.

1. Generate a random sample from some population distribution
2. Calculate sample mean, standard deviation and t test statistic
3. Decide if the null hypothesis is rejected
4. Repeat 1-3, counting the number of rejections

1.1. Sampling from normal distributions

```
counter <- 0 # set counter to 0
t.crit <- qt(0.95,14) #5% critical value

for (i in 1:1000)
{
  x <- rnorm(15, 25, 4) # draw a random sample of size 15
  # from a N(25,4) distribution
  t <- (mean(x)-25)*sqrt(15)/sd(x)
  if (t >= t.crit) # check to see if result is significant
    counter <- counter + 1 # increase counter by 1
}

counter/1000 #compute estimate of Type I error rate
## [1] 0.06
```

If we execute this code again, a different set of random samples will be selected, and a different estimate obtained

```
counter <- 0 # set counter to 0
t.crit <- qt(0.95,14) #5% critical value

for (i in 1:1000)
{
  x <- rnorm(15, 25, 4) # draw a random sample of size 15
  # from a N(25,4) distribution
  t <- (mean(x)-25)*sqrt(15)/sd(x)
  if (t >= t.crit) # check to see if result is significant
```

```

    counter <- counter + 1      # increase counter by 1
  }

  counter/1000 #compute estimate of Type I error rate
## [1] 0.043

```

1.2 Specify a seed to get identical results each time.

```

set.seed(4123)
counter <- 0      # set counter to 0

t.crit <- qt(0.95,14) #5% critical value

for (i in 1:1000)
{
  x <- rnorm(15, 25, 4)      # draw a random sample of size 15
  #from a N(25,4) distribution
  t <- (mean(x)-25)*sqrt(15)/sd(x)
  if (t >= t.crit)          # check to see if result is significant
    counter <- counter + 1    # increase counter by 1
}

  counter/1000 #compute estimate of Type I error rate
## [1] 0.043

```

Execute the same code again:

```

set.seed(4123)
counter <- 0      # set counter to 0
t.crit <- qt(0.95,14) #5% critical value

for (i in 1:1000)
{
  x <- rnorm(15, 25, 4)      # draw a random sample of size 15
  #from a N(25,4) distribution
  t <- (mean(x)-25)*sqrt(15)/sd(x)
  if (t >= t.crit)          # check to see if result is significant
    counter <- counter + 1    # increase counter by 1
}

  counter/1000 #compute estimate of Type I error rate
## [1] 0.043

```

Instead of using a counter, we may want to store the results so they can be explored later. In the code below, a vector is created and used to store the calculated t-statistics.

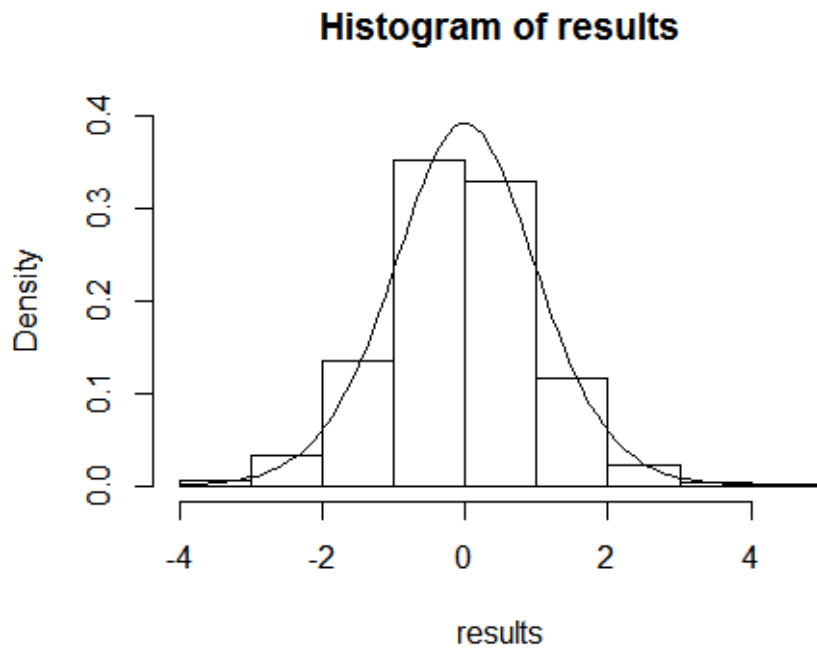
```
set.seed(4123)
nsims <- 1000
t.crit <- qt(0.95,14)           #5% critical value
results <- numeric(nsims)      #Vector to store t statistics

for (i in 1:nsims)
{
  x <- rnorm(15, mean=0, sd=1)  # draw a random sample of size 15
                                # from a N(25,4) distribution
  results[i] <- (mean(x)-0)*sqrt(15)/sd(x)
}

sum(results >= t.crit)/nsims   #compute estimate of error rate
## [1] 0.043
```

Having the results saved in a vector allows us to explore the actual sampling distribution. Below we graphically assess agreement between theoretical and actual distributions.

```
hist(results, freq = F, ylim=c(0,0.4))  # Plot histogram of t
statistics
curve(dt(x,14), add = TRUE)             # superimpose t(14) density
```



1.3 Sampling from an exponential distributions

```

set.seed(4123)
nsims <- 1000
t.crit <- qt(0.95,14)           #5% critical value
results <- numeric(nsims)      #Vector to store t statistics

for (i in 1:nsims)
{
  x <- rexp(15, rate=1/25)     # draw a random sample of size 15 from an
                              # Exp(mean=25) distribution
  results[i] <- (mean(x)-25)*sqrt(15)/sd(x)
}

sum(results >= t.crit)/nsims   #compute estimate of error rate
## [1] 0.015

```

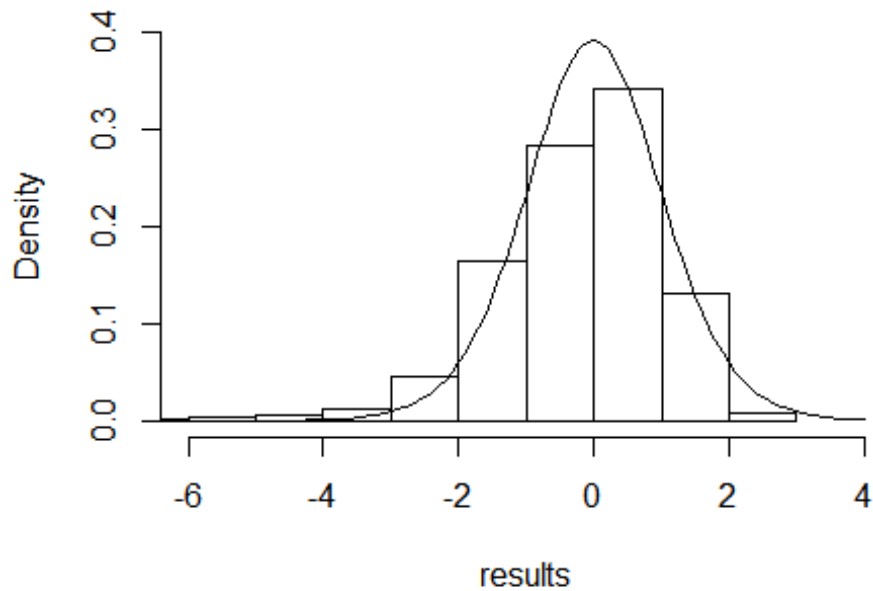
Graphically assess agreement between theoretical and actual distributions.

```

hist(results, freq = F, xlim=c(-6,4), ylim=c(0,0.4)) # Plot histogram of
t statistics
curve(dt(x,14), add = TRUE) # superimpose t(14) density

```

Histogram of results



Available distributions (<http://www.stat.umn.edu/geyer/old/5101/rlook.html>)

Distribution	Functions			
Beta	pbeta	qbeta	dbeta	rbeta
Binomial	pbinom	qbinom	dbinom	rbinom
Cauchy	pcauchy	qcauchy	dcauchy	rcauchy
Chi-Square	pchisq	qchisq	dchisq	rchisq
Exponential	pexp	qexp	dexp	rexp
F	pf	qf	df	rf
Gamma	pgamma	qgamma	dgamma	rgamma
Geometric	pgeom	qgeom	dgeom	rgeom
Hypergeometric	phyper	qhyper	dhyper	rhyper
Logistic	plogis	qlogis	dlogis	rlogis
Log Normal	plnorm	qlnorm	dlnorm	rlnorm
Negative Binomial	pnbinom	qnbinom	dnbinom	rnbinom
Normal	pnorm	qnorm	dnorm	rnorm
Poisson	ppois	qpois	dpois	rpois
Student t	pt	qt	dt	rt
Studentized Range	ptukey	qtukey	dtukey	rtukey
Uniform	punif	qunif	dunif	runif

Weibull pweibull qweibull dweibull rweibull
Wilcoxon Rank Sum Statistic pwilcox qwilcox dwilcox rwilcox
Wilcoxon Signed Rank Statistic psignrank qsignrank dsignrank rsignrank

2. Bootstrap Confidence intervals

Suppose we want to estimate a population parameter, based on random sample.

- *Classical World*: Observe one sample and the value of the sample statistic. Sampling distribution is determined by considering all possible (unobserved) samples from the same assumed population. Cannot directly observe the sampling distribution.
- *Bootstrap World*: Rather than assume a population, consider the observed sample to be the best estimate of the population. In fact, we will assume that it represents the probability distribution for the population. We can then generate all (or at least very many) possible samples by taking bootstrap samples (with replacement), from this estimated population and thus observe the sampling distribution of the sample estimator.

2.1 Drawing bootstrap samples using R.

We start with a very small data set, a set of new employee test scores:

23, 31, 37, 46, 49, 55, 57

First select a sample of size 7, with replacement and compute the mean of the bootstrap sample.

```
score <- c(37,49,55,57,23,31,46)
mean <- mean(score)
mean

## [1] 42.57143

boot <- sample(score, size=7, replace=TRUE)
boot

## [1] 31 37 31 31 31 37 31

mean.boot <- mean(boot)
mean.boot

## [1] 32.71429
```


We need to do this many times to estimate the sampling distribution of the mean.

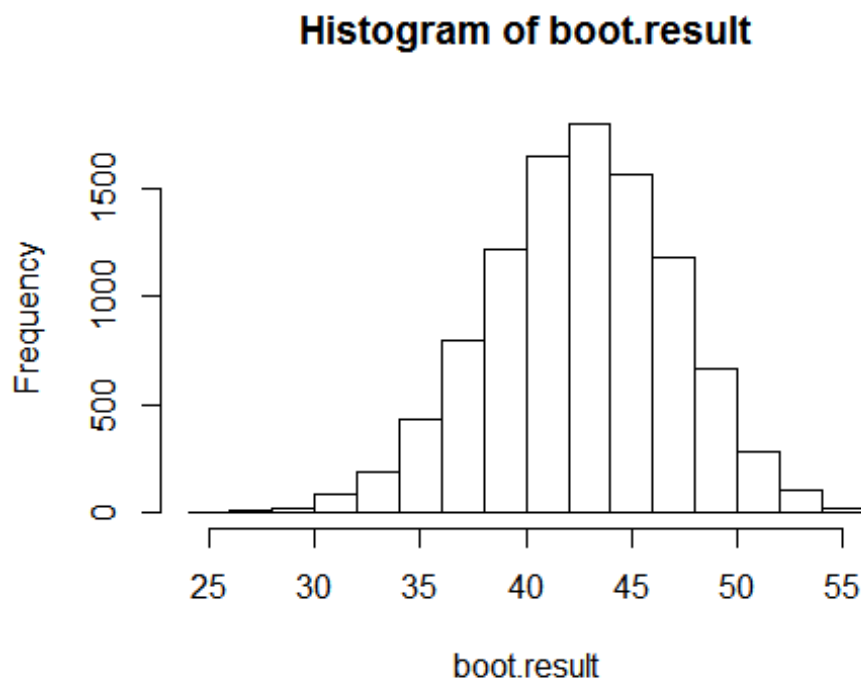
```
score <- c(37,49,55,57,23,31,46)
mean <- mean(score)
mean

## [1] 42.57143

N <- length(score)

nboots <- 10000
boot.result <- numeric(nboots)
for(i in 1:nboots)
{
  boot.samp <- sample(score, N, replace=TRUE)
  boot.result[i] <- mean(boot.samp)
}

hist(boot.result)
```



2.2 Bootstrap confidence intervals

Example. Suppose we have a random sample of size 30 from an exponential distribution with mean 25. We want to use the sample mean to estimate the population mean.

We will discuss three ways to construct confidence intervals using bootstrapping.

2.2.1 Percentile method

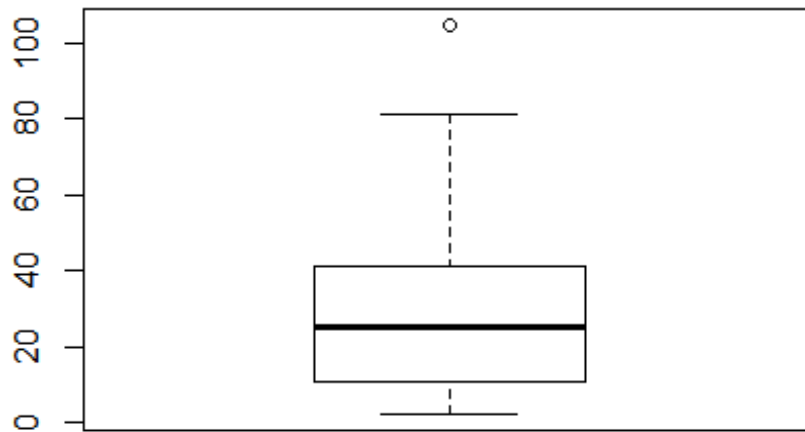
If the estimator of the population parameter is the statistic used to create the distribution (e.g., the sample mean), then the confidence interval is simply the equal-tail quantiles that correspond to the confidence level.

Bootstrap 95% percentile confidence interval

```
set.seed(4123)
x.exp <- rexp(30, rate=1/25)
x.exp

## [1]  2.528853  2.235845 40.423011  5.255557  3.355874  2.724010
## [7] 10.787030  9.792154 31.882324 15.816492 13.925713 38.726646
## [13] 19.283214 70.833128 30.556819 31.620638 64.401698 34.713028
## [19]  6.153141 81.498008 16.034828 80.867315 41.204157 77.066567
## [25] 14.237094 47.647705 35.505526 13.203980 104.795832 11.897545

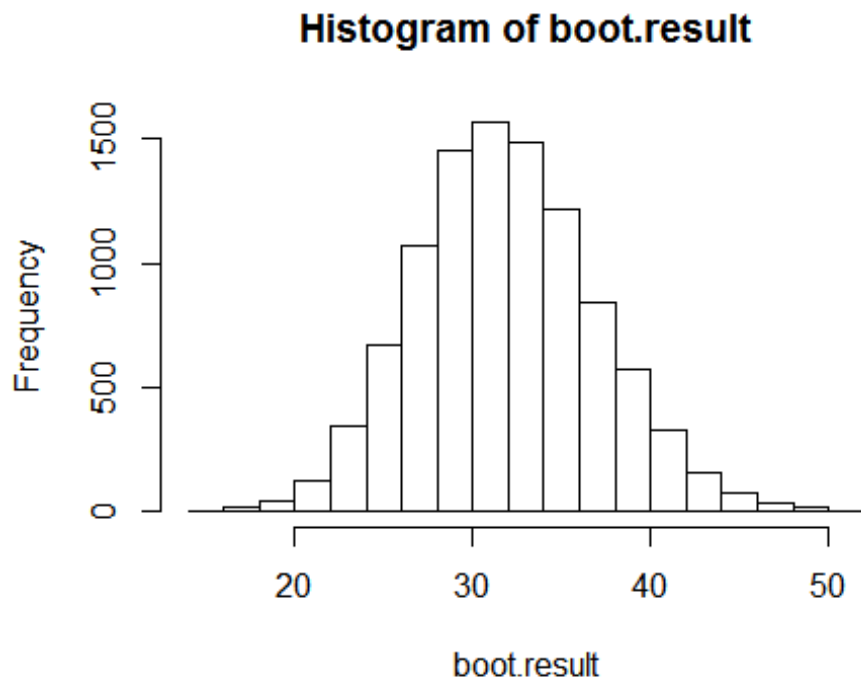
boxplot(x.exp)
```



```
n <- length(x.exp)
mean.exp <- mean(x.exp)

nboots <- 10000
boot.result <- numeric(nboots)
for(i in 1:nboots)
{
  boot.samp <- sample(x.exp, n, replace=TRUE)
  boot.result[i] <- mean(boot.samp)
}

hist(boot.result)
```



```

mean.exp
## [1] 31.96579

quantile(boot.result, c(0.025,0.975))
##      2.5%      97.5%
## 22.49355 42.19327

```

2.2.2 Pivot method

A pivot quantity is a function of the estimator whose distribution does not depend on the parameter being estimated.

Example: Estimating the population mean, μ based on the sample mean, \bar{Y} . Then the statistic $T = \frac{\bar{Y} - \mu}{S / \sqrt{n}} \sim t(n-1)$ has a Student's t distribution with $n-1$ degrees of freedom.

Because the distribution of T does not depend on μ , T is a pivot quantity.

When such a quantity exists, we can then use bootstrapping to estimate the distribution of the pivot quantity—essentially a custom table—and use quantiles from the table to create the confidence interval.

For the statistic $T = \frac{Y - \mu}{S / \sqrt{n}}$, the confidence interval is given by

$$\bar{Y} - t_{b,0.975} \left(\frac{S}{\sqrt{n}} \right) \leq \mu \leq \bar{Y} + t_{b,0.025} \left(\frac{S}{\sqrt{n}} \right),$$

where \bar{Y}, S are the values from the original sample.

Bootstrap 95% T-pivot confidence intervals

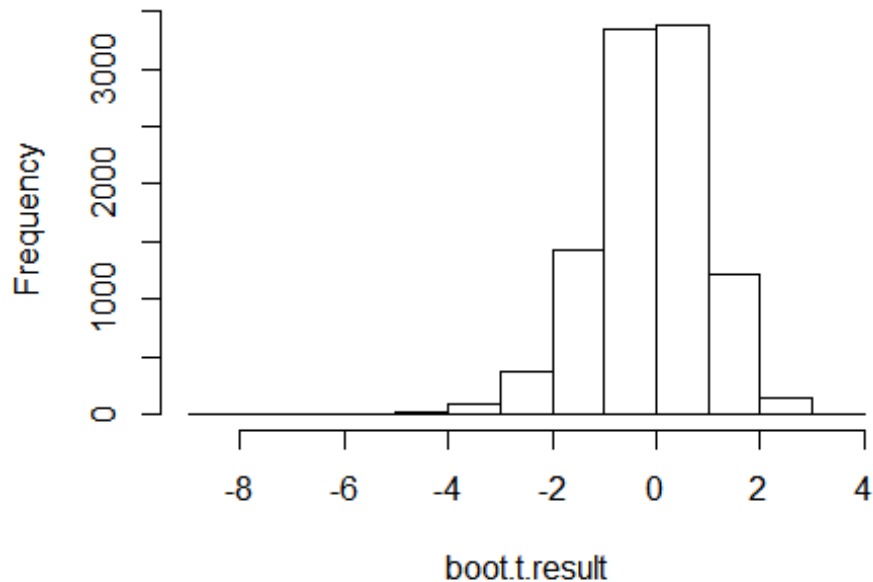
```
set.seed(4123)
x.exp <- rexp(30, rate=1/25)

n <- length(x.exp)
mean.exp <- mean(x.exp)
sd.exp <- sd(x.exp)

nboots <- 10000
boot.t.result <- numeric(nboots)
for(i in 1:nboots)
{
  boot.samp <- sample(x.exp, n, replace=TRUE)
  boot.t.result[i] <- (mean(boot.samp)-mean.exp)*sqrt(n)/sd(boot.samp)
}

hist(boot.t.result)
```

Histogram of boot.t.result



```
t.upper <- quantile(boot.t.result, 0.975)
t.lower <- quantile(boot.t.result, 0.025)

lower95.limit <- mean.exp - t.upper*sd.exp/sqrt(n)
upper95.limit <- mean.exp - t.lower*sd.exp/sqrt(n)
lower95.limit

##      97.5%
## 22.80743

upper95.limit

##      2.5%
## 44.39896
```

Quantiles of t(29) distribution compared to bootstrap distribution

```
quantile(boot.t.result, c(0.005,0.025,0.05,0.95,0.975,0.995))

##      0.5%      2.5%      5%      95%      97.5%      99.5%
## -3.515286 -2.416519 -1.977130  1.496785  1.780026  2.367328

qt(c(0.005,0.025,0.05,0.95,0.975,0.995),n-1)

## -2.756386 -2.045230 -1.699127  1.699127  2.045230  2.756386
```

“Theory” behind T-pivot interval

Suppose we wish to estimate the mean of a population. The pivotal method can be used, assuming we can find a statistic whose distribution does not depend on the parameters to be estimated.

Ex: $t = \frac{\bar{X} - \mu}{s/\sqrt{n}}$. If $X \sim N(\mu, \sigma^2)$, then $t \sim t(n-1)$. The distribution of t does not depend on

either μ or σ^2 , and thus t is a pivotal quantity. Then since

$$P\left(t_{\alpha/2} \leq \frac{\bar{X} - \mu}{\frac{s}{\sqrt{n}}} \leq t_{1-\alpha/2}\right) = 1 - \alpha, \text{ we have}$$

$$t_{\alpha/2} \left(\frac{s}{\sqrt{n}}\right) \leq \bar{X} - \mu \leq t_{1-\alpha/2} \left(\frac{s}{\sqrt{n}}\right)$$

$$-\bar{X} + t_{\alpha/2} \left(\frac{s}{\sqrt{n}}\right) \leq -\mu \leq -\bar{X} + t_{1-\alpha/2} \left(\frac{s}{\sqrt{n}}\right)$$

$$\bar{X} - t_{\alpha/2} \left(\frac{s}{\sqrt{n}}\right) \geq \mu \geq \bar{X} - t_{1-\alpha/2} \left(\frac{s}{\sqrt{n}}\right),$$

$$\Rightarrow \bar{X} - t_{1-\alpha/2} \left(\frac{s}{\sqrt{n}}\right) \leq \mu \leq \bar{X} - t_{\alpha/2} \left(\frac{s}{\sqrt{n}}\right)$$

is a $100(1 - \alpha)\%$ confidence interval for μ .

Now, since $t \sim t(n-1)$ where $t(n-1)$ is symmetric, we have $t_{\alpha/2} = -t_{1-\alpha/2}$. We can write

$$\bar{X} - t_{1-\alpha/2} \left(\frac{s}{\sqrt{n}}\right) \leq \mu \leq \bar{X} + t_{1-\alpha/2} \left(\frac{s}{\sqrt{n}}\right),$$

where if $\alpha = .05, 1 - \alpha/2 = .975$.

If the sample does not come from a normal population, t is still a pivotal quantity so we can still write

$$P\left(t_{\alpha/2} \leq \frac{\bar{X} - \mu}{\frac{s}{\sqrt{n}}} \leq t_{1-\alpha/2}\right) = 1 - \alpha$$

$$\Rightarrow \bar{X} - t_{1-\alpha/2} \left(\frac{s}{\sqrt{n}} \right) \leq \mu \leq \bar{X} + t_{\alpha/2} \left(\frac{s}{\sqrt{n}} \right) \quad (1)$$

But we can not say that $t \sim t(n-1)$!

One Solution: Estimate the distribution of t using bootstrap sampling.

*For each bootstrap sample of size n from the data, compute $t_b = \frac{\bar{X}_b - \bar{X}}{\left(\frac{s_b}{\sqrt{n}} \right)}$, then find $t_{b,\alpha/2}$ and

$t_{b,1-\alpha/2}$ and substitute into (1) above.

2.2.3 Using bootstrap samples to estimate standard error.

The “original” or often called “standard” bootstrap method. This method is motivated by the “Wald” interval which assumes that many statistics are approximately normally distributed for large sample sizes, and creates the interval as

$$\text{Estimator} \pm Z_{\alpha/2} * SE .$$

If a formula for the SE is not available, then SE can be estimated using bootstrapping. For example, to estimate the standard error of the mean:

1. Compute the mean squared error, $MSE = \frac{1}{n} \sum_{i=1}^n (\bar{X}_{b,i} - \bar{X})^2$, where \bar{X} is the mean of the original sample and $\bar{X}_{b,i}$ is the mean of the i th bootstrap sample.
2. Compute $SE = \sqrt{MSE}$
3. The confidence interval is $\bar{X} \pm Z_{\alpha/2} * SE$

```
set.seed(4123)
x.exp <- rexp(30, rate=1/25)

n <- length(x.exp)
mean.exp <- mean(x.exp)

nboots <- 10000
boot.MSE.result <- numeric(nboots)
for(i in 1:nboots)
{
  boot.samp <- sample(x.exp, n, replace=TRUE)
  boot.MSE.result[i] <- (mean(boot.samp)-mean.exp)^2
}

SE <- sqrt(mean(boot.MSE.result))

lower95.limit <- mean.exp - qnorm(0.975)*SE
upper95.limit <- mean.exp + qnorm(0.975)*SE
lower95.limit
## [1] 22.07738
upper95.limit
```

```
## [1] 41.85421
```

3. Randomization/Permutation tests--Comparing two or more groups

A company is trying to decide whether to augment its traditional instruction for new employees with computer assisted instruction. Seven new employees are selected. Four are assigned at random to the new method and the remaining three to the traditional method. A test is given at the end of instruction for all employees, and the scores are given below.

New method	Smith (37), Lin (49), O'Neal (55), Zedillo (57)
Traditional method	Johnson (23), Green (31), Zook (46)

Suppose we would like to test for evidence that the new method tends to produce higher scores. We might test $H_0 : \mu_N = \mu_T$ vs. $H_a : \mu_N > \mu_T$, using a t-test.

- $t = 2.08$, p -value = 0.046. Statistically significant?
- Independent random samples from normally distributed populations?

Permutation/Randomization test.

- Sampling distribution based upon all possible assignments of the experimental units to treatments.
- Important assumption: Each assignment is equally likely under the null hypothesis—guaranteed by random assignment.

3.1 Creating a random permutation using R.

```
set.seed(4123)
score <- c(37,49,55,57,23,31,46)
perm <- sample(score, replace=F)
perm
## [1] 46 55 37 31 23 57 49
```

Now, compute the mean of the first 4 entries, the last 3 and compute the difference

```
mean.new <- mean(perm[1:4])
mean.new
## [1] 42.25
mean.trad <- mean(perm[5:7])
mean.trad
```

```
## [1] 43
diff <- mean.new-mean.trad
diff
## [1] -0.75
```

Another way is to create an index vector, and sample for just one group.

```
set.seed(4123)
N <- length(score)
index <- sample(N, size=4, replace=F)
index
## [1] 7 3 1 6
score[index]
## [1] 46 55 37 31
score[-index]
## [1] 49 57 23
diff2 <- mean(score[index]) - mean(score[-index])
diff2
## [1] -0.75
```

3.2 Comparing groups

Steps of the permutation test using mean difference as the test statistic:

1. Compute the test statistic: $D_{obs} = \bar{X}_N - \bar{X}_T$ on the observed data
2. Randomly assign units to treatments, and recompute D
3. Repeat #2 for all possible random assignments of units to treatments
4. The p -value is $P(D \geq D_{obs}) = (\# D \text{ values at least as large as } D_{obs}) / (\# \text{ randomizations})$.

3.2.1. Exact randomization distribution.

The table below lists all 35 possible randomizations and their corresponding mean differences. Note that the permutation we created is that listed in row 19.

Randomization	New1	New2	New3	New4	Trad5	Trad6	Trad7	$\bar{X}_{New} - \bar{X}_{Trad}$	Sum New
1	46	49	55	57	23	31	37	21.4167	207
*2	37	49	55	57	46	23	31	16.1667	198
3	37	46	55	57	31	49	23	14.4167	195
4	31	49	55	57	46	23	37	12.6667	192
5	31	46	55	57	37	49	23	10.9167	189
6	37	46	49	57	31	23	55	10.9167	189
7	37	46	49	55	31	23	57	9.7500	187
8	23	49	55	57	46	31	37	8.0000	184
9	31	46	49	57	23	37	55	7.4167	183
10	23	46	55	57	37	49	31	6.2500	181
11	31	46	49	55	23	37	57	6.2500	181
12	31	37	55	57	46	49	23	5.6667	180
13	23	46	49	57	31	37	55	2.7500	175
14	31	37	49	57	46	23	55	2.1667	174
15	23	46	49	55	31	37	57	1.5833	173
16	23	37	55	57	46	49	31	1.0000	172
17	31	37	49	55	46	23	57	1.0000	172
18	31	37	46	57	23	49	55	0.4167	171
19	31	37	46	55	23	49	57	-0.7500	169
20	23	31	55	57	46	49	37	-2.5000	166
21	23	37	49	57	46	31	55	-2.5000	166
22	23	37	49	55	46	31	57	-3.6667	164
23	23	37	46	57	31	49	55	-4.2500	163
24	31	37	46	49	23	55	57	-4.2500	163
25	23	37	46	55	31	49	57	-5.4167	161
26	23	31	49	57	46	37	55	-6.0000	160
27	23	31	49	55	46	37	57	-7.1667	158
28	23	31	46	57	37	49	55	-7.7500	157
29	23	31	46	55	37	49	57	-8.9167	155
30	23	37	46	49	31	55	57	-8.9167	155
31	23	31	46	49	37	55	57	-12.4167	149
32	23	31	37	57	46	49	55	-13.0000	148
33	23	31	37	55	46	49	57	-14.1667	146
34	23	31	37	49	46	55	57	-17.6667	140
35	23	31	37	46	49	55	57	-19.4167	137

Two of 35 possible assignments of units to observations were as large or larger than the observed value of $D_{obs} = 16.2$ (one of these is $D_{obs} = 16.2$). Thus the p -value is $P(D \geq D_{obs}) = 2/35 \approx 0.057$.

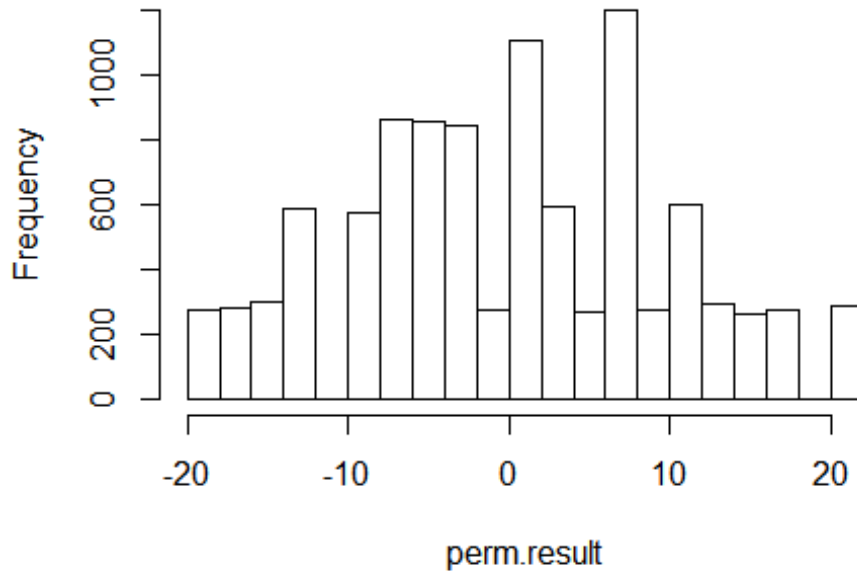
- Exact p -value—does not depend upon unverifiable assumptions
- The p -value (0.046) from the t -test can be viewed as an approximation to the exact permutation p -value.

3.2.2 Using R to generate many randomizations and compute p -value.

Now, we use a **for** loop as before to create many random permutations and corresponding mean differences, store all the mean differences and then compute the p -value.

```
score <- c(37,49,55,57,23,31,46)
observed.diff <- mean(score[1:4]) - mean(score[5:7])
N <- length(score)
set.seed(4132)
nperms <- 9999
perm.result <- numeric(nperms)      # vector to save the random differences
for(i in 1:nperms)
{
  index <- sample(N, size=4, replace=FALSE) #Choose 4 values w/o replacement
  perm.result[i] <- mean(score[index]) - mean(score[-index])
}
hist(perm.result)
```

Histogram of perm.result



```
(sum(perm.result >= observed.diff)+1)/(nperms + 1)      #P-value
## [1] 0.0562
```

Note that this p-value is not exact.

When sample sizes are moderate to large, enumerating all possible arrangements may be very time consuming at best and practically impossible at worst. As the table below illustrates, with two samples of 25, there are over 100 trillion arrangements to consider!

Total sample size	Sample 1	Sample 2	Permutations
10	5	5	252
20	10	10	184,756
30	15	15	155,117,520
40	20	20	137,846,528,820
50	25	25	126,410,606,437,752

Solution:

Randomly sample the population of permutations. While enumerating 1 trillion permutations may be computationally time-prohibitive, in many cases enumerating 10,000 or even 100,000 is not. When estimating an exact p -value based on a random

sample of R permutations, the exact p -value would be expected to be within $2\sqrt{\frac{p(1-p)}{R}}$ with 95% confidence. For example, if the true p -value is $p = 0.05$, the estimate would be expected to be within the margins of error below:

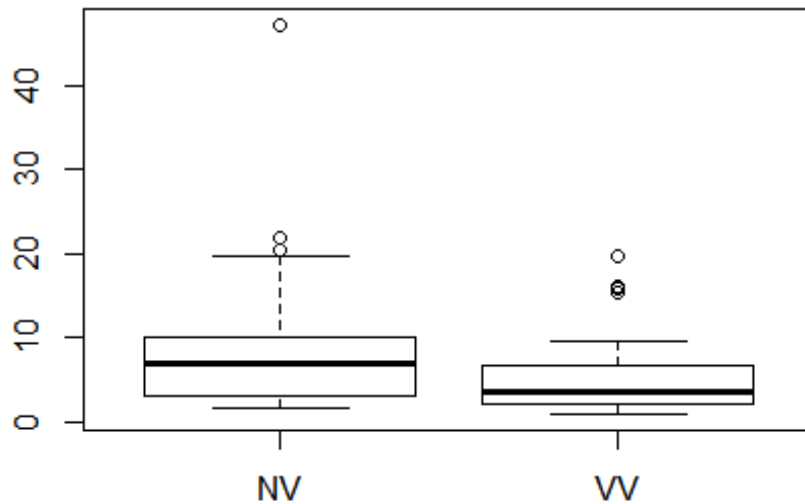
R	95% margin of error
1000	0.013784
5000	0.006164
10000	0.004359
100000	0.001378

Example. This dataset contains results from an experiment in visual perception using random dot stereograms, such as that shown below. Both images appear to be composed entirely of random dots. However, they are constructed so that a 3D image (of a diamond) will be seen, if the images are viewed with a stereo viewer, causing the separate images to fuse. Another way to fuse the images is to fixate on a point between them and defocus they eyes, but this technique takes some effort and practice. An experiment was performed to determine whether knowledge of the form of the embedded image affected the time required for subjects to fuse the images. One group of subjects (group NV-43 subjects) received either no information or just verbal information about the shape of the embedded object. A second group (group VV-35 subjects) received both verbal information and visual information (e.g., a drawing of the object).

[Cleveland, W. S. (1993). *Visualizing Data*. Original source: Frisby, J. P. and Clatworthy, J.L., "Learning to see complex random-dot stereograms," *Perception*, 4, (1975), pp. 173-178]

A randomization test on these data involves over $\binom{78}{43} > 1.8 \times 10^{22}$ permutations. In the following code, the data are read and a boxplot created that shows several outliers in each group. Thus we consider using the median difference as the test statistic instead of the mean difference. The p-value is estimated based on 10,000 randomly sampled randomizations.

```
fusion=read.table("C:/Users/sjricht2/Documents/DataSets/Independent samples T
-test/Fusion_data.txt", header=TRUE)
boxplot(fusion$time~fusion$treatment)
```



```

names(fusion)
## [1] "time"      "treatment"

N <- length(fusion$time)
Treat1 <- subset(fusion, Select=time, treatment=="NV", drop=T)
Treat2 <- subset(fusion, Select=time, treatment=="VV", drop=T)
N1 <- length(Treat1)

head(Treat1,5)

##      time treatment
## 1 47.20001      NV
## 2 21.99998      NV
## 3 20.39999      NV
## 4 19.70001      NV
## 5 17.40000      NV

head(Treat2,5)

##      time treatment
## 44 19.70001      VV
## 45 16.19998      VV
## 46 15.90000      VV
## 47 15.40002      VV
## 48  9.70000      VV

```

```
observed.fusion <- median(Treat1$time)-median(Treat2$time)
observed.fusion

## [1] 3.3

nperms <- 9999
set.seed(4132)

result <- numeric(nperms)
for(i in 1:nperms)
{
  index <- sample(N, size=N1, replace = FALSE)
  result[i] <- median(fusion$time[index]) - median(fusion$time[-index])
}

(sum(result >= observed.fusion)+1)/(nperms + 1)

## [1] 0.3213
```

What hypotheses are being tested by the permutation test?

No population distribution is assumed, and thus it does not make sense to test parameters (e.g., equality of means).

Null hypothesis: Population distributions are identical

Alternative hypothesis: Observations tend to be larger in one of the populations

3.2.3 Choice of test statistic

Another advantage of the permutation test is that the function of the sample that is best suited to address the research question may be used. For a test of location difference, we may use

- Difference of means
- Difference of medians
- others (e.g., trimmed means, ratios)

3.3 Wilcoxon Rank sum test: Permutation test on rank transformed data

- Requires only ordinal level data
- Reduces effect of outliers for interval level data
- Hypotheses are the same as permutation test on raw data

Rank transformation: replace data with their respective ranks

- For WRS test, data from both samples are combined, then assigned ranks
- Tied observations generally all receive the “average rank”

Steps for WRS test.

1. Combine observations and assign ranks, with tied observations receiving the average rank
2. Perform permutation test on ranks (mean difference or sum of ranks in sample 1 can be used as test statistic)

```
fusion=read.table("C:/Users/sjricht2/Documents/DataSets/Independent samples T
-test/Fusion_data.txt", header=TRUE)

fusion$ranks.time <- rank(fusion$time)
head(fusion,5)

##      time treatment ranks.time
## 1 47.20001      NV      78.0
## 2 21.99998      NV      77.0
## 3 20.39999      NV      76.0
## 4 19.70001      NV      74.5
## 5 17.40000      NV      73.0

N <- length(fusion$time)
Treat1 <- subset(fusion, Select=ranks.time, treatment=="NV", drop=T)
Treat2 <- subset(fusion, Select=ranks.time, treatment=="VV", drop=T)
N1 <- length(Treat1)

observed.fusion <- mean(Treat1$ranks.time)-mean(Treat2$ranks.time)
observed.fusion

## [1] 11.42791

nperms <- 9999
set.seed(4132)

result <- numeric(nperms)
```

```

for(i in 1:nperms)
{
  index <- sample(N, size=N1, replace = FALSE)
  result[i] <- mean(fusion$ranks.time[index]) -
               mean(fusion$ranks.time[-index])
}

(sum(result >= observed.fusion)+1)/(nperms + 1)

## [1] 0.2064

```

3.4 Selecting Among Two-sample tests

- 1) *t*-test—If selecting independent random sample from normal populations, is optimal for detecting location difference
- 2) Permutation test using means—Gives exact *p*-value regardless of distribution of population. Power will be similar to *t*-test
- 3) Permutation test using medians—Can have higher power than tests on means, especially for skewed and heavy-tailed distributions
- 4) Permutation test using ranks (Wilcoxon rank-sum test)-- Can have higher power than tests on means, especially for skewed and heavy-tailed distributions

The WRS test has been studied extensively in relation to the *t*-test. The *t*-test tends to have higher power for symmetric distributions, especially for lighter tailed distributions and smaller sample sizes. The WRS test generally has higher power for heavier-tailed distributions and moderate to large sample sizes.

3.5 More than two groups

The methods of this section can be extended to more than two groups. The ANOVA F-statistic can be used as the test statistic if using the raw, numeric data. The randomization test based on ranks is known as the Kruskal-Wallis test.

3.6 Randomization tests for contingency tables

Example. Seven patients are included in a study to compare two methods of relieving postoperative pain. Three are allowed to control the amount of pain-relief medicine themselves, while the other four are given a physician prescribed level of medicine according to standard practice. Afterward, the patients evaluate their satisfaction as either “not satisfied (NS)”, “somewhat satisfied (SS)” or “very satisfied (VS)”.

	NS	SS	VS
Physician prescribed	2	2	0
Self-administered	0	1	2

Is there an association between the method of pain-relief and satisfaction?

Null hypothesis—Method and satisfaction are independent (or, the distributions of patients among the satisfaction categories are identical)

Alternative hypothesis—There is an association between method and satisfaction (or, the distributions of patients among the satisfaction categories are different).

Typically a chi-squared test would be considered to assess these hypotheses, using the test statistic

$$\chi^2 = \sum_{\text{all cells}} \frac{(\text{observed} - \text{expected})^2}{\text{expected}},$$

which has an approximate $\chi^2((\text{rows} - 1)(\text{cols} - 1))$ distribution for large sample sizes.

However, the approximation can be poor when there are small expected counts.

The randomization test can be viewed as identical to the two-group permutation test with a quantitative response, i.e.,

Physician prescribed	NS ₁ , NS ₂ , SS ₃ , SS ₄
Self-administered	SS ₅ , VS ₆ , VS ₇

Then a randomization test can be performed by randomly assigning observations to groups, as follows:

1. Compute the test statistic: $X_{obs}^2 = \sum_{all\ cells} \frac{(observed - expected)^2}{expected}$ on the observed data
2. Randomly assign units to treatments, construct the corresponding table and compute X_{perm}^2
3. Repeat #2 for all possible random assignments of observations to groups
4. The p -value is $P(X_{perm}^2 \geq X_{obs}^2) = (\# X_{perm}^2 \text{ values at least as large as } X_{obs}^2) / (\# \text{ randomizations})$.

Example. The expected frequencies are computed below as (row total)*(column total)/n.

```
table <- matrix(
  c(2, 2, 0,
    0, 1, 2),
  nrow = 2, byrow = TRUE
)
table

##      [,1] [,2] [,3]
## [1,]    2    2    0
## [2,]    0    1    2

chitest <- chisq.test(as.table(table))

## Warning in chisq.test(as.table(table)): Chi-squared approximation may be
## incorrect

chitest
```



```
##
## Pearson's Chi-squared test
##
## data: as.table(table)
## X-squared = 4.2778, df = 2, p-value = 0.1178

chitest$expected

##           A           B           C
## A 1.1428571 1.714286 1.1428571
## B 0.8571429 1.285714 0.8571429
```

For the observed table, the chi-squared statistic is 4.28.

One particular randomization would be the following

Physician prescribed	SS ₅ , NS ₂ , SS ₃ , SS ₄
Self-administered	NS ₁ , VS ₆ , VS ₇

and the resulting table would be

	NS	SS	VS
Physician prescribed	1	3	0
Self-administered	1	0	2

The chi-squared statistic for this table is 4.96.

This process would be somewhat cumbersome to code. Luckily, R has built-in functions that can do this.

Approximate p-value based on a random sample of permutations

```
table <- matrix(
  c(2, 2, 0,
    0, 1, 2),
  nrow = 2, byrow = TRUE
)
```

```

chisq.test(as.table(table),simulate.p.value=T, B=10000)

##
## Pearson's Chi-squared test with simulated p-value (based on 10000
## replicates)
##
## data: as.table(table)
## X-squared = 4.2778, df = NA, p-value = 0.3247

```

Exact p-value based on all possible permutations

```

table <- matrix(
  c(2, 2, 0,
    0, 1, 2),
  nrow = 2, byrow = TRUE
)

fisher.test(as.table(table))

##
## Fisher's Exact Test for Count Data
##
## data: as.table(table)
## p-value = 0.3143
## alternative hypothesis: two.sided

```

Approximate p-value based on chi-squared distribution

```

table <- matrix(
  c(2, 2, 0,
    0, 1, 2),
  nrow = 2, byrow = TRUE
)

chisq.test(as.table(table))

## Warning in chisq.test(as.table(table)): Chi-squared approximation may be
## incorrect

##
## Pearson's Chi-squared test
##
## data: as.table(table)
## X-squared = 4.2778, df = 2, p-value = 0.1178

```


4. Methods for correlation and regression

4.1. Randomization test for linear relation between quantitative variables

Data are either

- (a) a random sample of bivariate pairs (X_i, Y_i) , or
- (b) values of Y_i are obtained for fixed values of X_i .

1(a): There is usually no a priori reason to assign either X or Y as independent or dependent.

Thus use correlation as a measure of strength of linear relationship.

$$\text{Correlation -- } \rho = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y} \quad (\text{population correlation})$$

$$\text{Estimate of } \rho \text{ -- } r = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2 \sum_{i=1}^n (Y_i - \bar{Y})^2}}$$

To test $H_0 : \rho = 0$:

The test statistic $t = \left(\sqrt{\frac{n-2}{1-r^2}} \right) r \sim t(n-2)$ can be used if the (X, Y) pairs are a random sample

from a bivariate normal population.

1(b): Fit the model $Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$ where ε_i 's are *iid* with mean 0 and variance σ^2 . β_1 is the slope of the regression line.

$$\text{Estimator of } \beta_1: \hat{\beta}_1 = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sum (X_i - \bar{X})^2}$$

To test $H_0: \beta_1 = 0$, the test statistic is $t = \sqrt{\frac{\sum (X_i - \bar{X})^2}{MSE}} \hat{\beta}_1 \sim t(n-2)$, if ε_i 's are normally distributed.

It can be shown that $\hat{\beta}_1 = r \left(\frac{S_Y}{S_X} \right)$ and also that $t_{corr} = t_{slope}$.

So, to test for a linear relation between X & Y , we can use either statistic.

4.1.1 Pearson correlation or slope of regression line

A permutation test may be used to obtain an exact *p-value*, regardless of the form of the distribution of ε_i 's. Under $H_0: \beta_1 = 0$ or $H_0: \rho = 0$, X does not affect the value of Y , so an observed Y is just as likely to occur with any X . Thus, the permutation distribution is derived from all possible assignments of the observed Y s to the observed X s.

The procedure is exactly the same as for comparing groups.

Steps of the permutation test using r or $\hat{\beta}_1$ as the test statistic:

1. Compute the test statistic, e.g., r , on the observed data
2. Randomly assign observations (Y 's) to treatments (X 's), and recompute the test statistic, r_{perm} .

3. Repeat #2 for all possible random assignments of observations to treatments

4. The p -value is

$$P(r_{perm} \geq r_{obs}) = (\# r_{perm} \text{ values at least as large as } r_{obs}) / (\# \text{ permutations}).$$

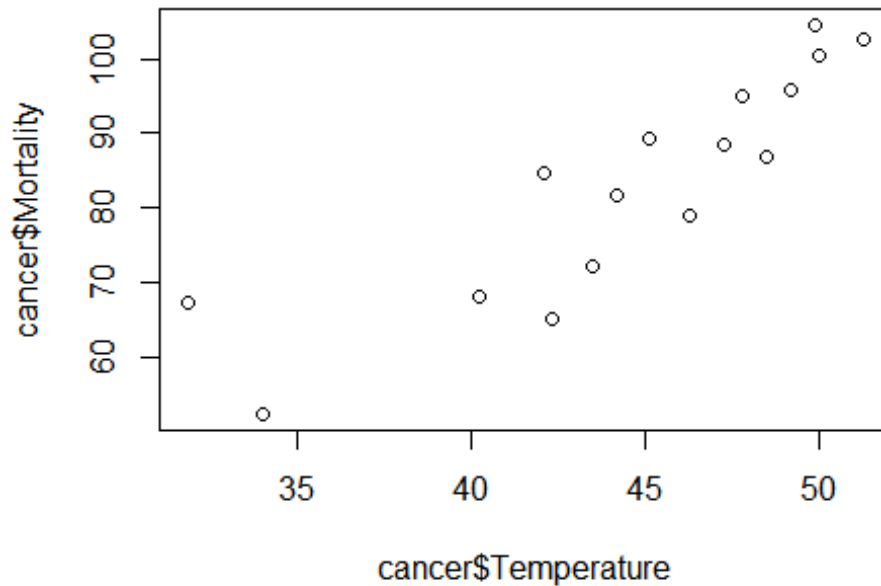
Example. Lea (1965) discussed the relationship between mean annual temperature and the mortality rate for a type of breast cancer in women. The subjects were residents of certain regions of Great Britain, Norway, and Sweden.

Randomization test for linear association

```
set.seed(4123)
cancer <- read.table('C:/Users/sjricht2/Documents/DataSets/Regression/Breast
Cancer Data.txt', header=T)
cancer

##      Mortality Temperature
## 1      102.5         51.3
## 2      104.5         49.9
## 3      100.4         50.0
## 4       95.9         49.2
## 5       87.0         48.5
## 6       95.0         47.8
## 7       88.6         47.3
## 8       89.2         45.1
## 9       78.9         46.3
## 10      84.6         42.1
## 11      81.7         44.2
## 12      72.2         43.5
## 13      65.1         42.3
## 14      68.1         40.2
## 15      67.3         31.8
## 16      52.5         34.0

plot(cancer$Temperature, cancer$Mortality)
```



```

r.obs <- cor(cancer$Temperature,cancer$Mortality)
r.obs

## [1] 0.8748544

slope.obs <- lm(cancer$Mortality~cancer$Temperature)$coeff[2]
slope.obs

## cancer$Temperature
##          2.357695

n <- length(cancer$Mortality)
nperms <- 9999 #set number of times to repeat this process

result.r <- numeric(nperms)
result.slope <- numeric(nperms)
for(i in 1:nperms)
{
  index <- sample(n, size=n, replace = FALSE)
  result.r[i] <-
    cor(cancer$Temperature,cancer$Mortality[index])
  result.slope[i] <-
    lm(cancer$Mortality[index]~cancer$Temperature)$coeff[2]
}
'Permutation test p-value'

```

```
## [1] "Permutation test p-value"
(sum(result.r >= r.obs)+1)/(nperms + 1)
## [1] 1e-04
(sum(result.slope >= slope.obs)+1)/(nperms + 1)
## [1] 1e-04
```

4.1.2. Rank correlation

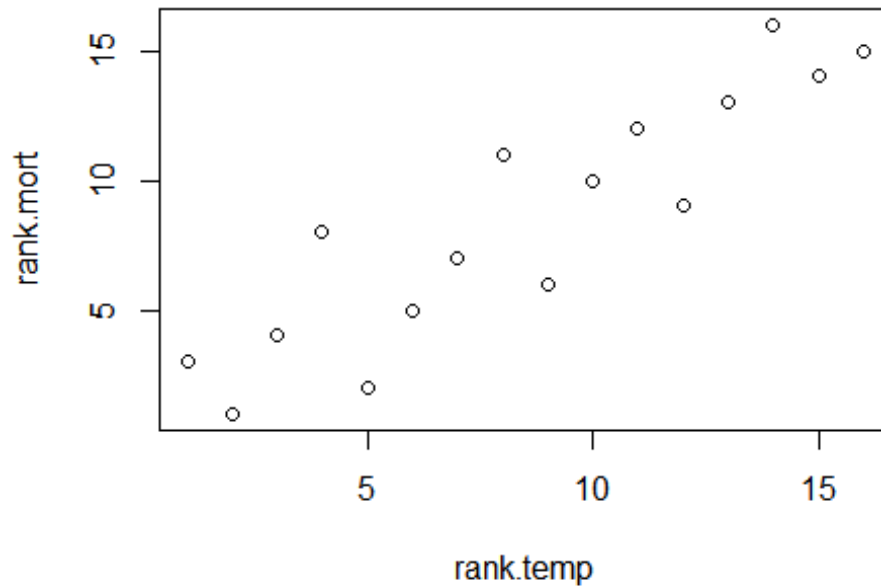
The randomization test can be carried out on rank transformed data. The X and Y values are ranked separately, and Pearson correlation calculated on the rank-transformed data. Pearson correlation calculated on rank transformed data is called *Spearman* correlation. The alternative hypothesis for this test is that the rank-transformed data have a linear association, or that the original data have a monotonic (strictly increasing or decreasing) relation.

Randomization test for rank correlation

```
set.seed(4123)
cancer <- read.table('C:/Users/sjricht2/Documents/DataSets/Regression/Breast
Cancer Data.txt', header=T)

rank.temp <- rank(cancer$Temperature)
rank.mort <- rank(cancer$Mortality)

plot(rank.temp,rank.mort)
```

```

r.obs <- cor(rank.temp,rank.mort)
r.obs

## [1] 0.9029412

n <- length(rank.mort)
nperms <- 9999 #set number of times to repeat this process

result.r <- numeric(nperms)

for(i in 1:nperms)
{
  index <- sample(n, size=n, replace = FALSE)
  result.r[i] <-
    cor(rank.temp,rank.mort[index])
}
'Permutation test p-value'

## [1] "Permutation test p-value"

(sum(result.r >= r.obs)+1)/(nperms + 1)

## [1] 1e-04

```

4.2 Bootstrap confidence intervals for correlation and slope

Intervals for ρ

Suppose we have a random sample of ordered pairs, (X_i, Y_i) , $i = 1, 2, \dots, n$. If the distribution of (X_i, Y_i) is bivariate normal, then it can be shown that:

$$Z = \frac{1}{2} \ln \left[\frac{1+r}{1-r} \right] \sim N \left[\frac{1}{2} \ln \left(\frac{1+\rho}{1-\rho} \right), \frac{1}{n-3} \right],$$

which can be used to construct an interval estimator for ρ . This interval is not robust to departures from normality, however. Notice also that the distribution of Z depends on the population correlation, ρ , and thus Z is not a pivot quantity.

Bootstrap interval.

- 1) Draw a specified number of **bivariate** bootstrap samples of size n , i.e., sample **pairs** of observations.
- 2) Compute r_{boot} , the sample correlation for each bootstrap sample.
- 3) Use the percentile method to construct confidence interval. (No pivot quantity exists).

4.2.1 Bivariate bootstrap sampling

```
boot.index <- sample(1:nrow(cancer), replace = TRUE)
boot.index
## [1] 8 1 16 13 13 16 1 3 15 3 6 14 7 2 14 12
cancer
## Mortality Temperature
## 1 102.5 51.3
## 2 104.5 49.9
## 3 100.4 50.0
## 4 95.9 49.2
## 5 87.0 48.5
## 6 95.0 47.8
## 7 88.6 47.3
```

```
## 8      89.2      45.1
## 9      78.9      46.3
## 10     84.6      42.1
## 11     81.7      44.2
## 12     72.2      43.5
## 13     65.1      42.3
## 14     68.1      40.2
## 15     67.3      31.8
## 16     52.5      34.0
```

```
boot.data <- cancer[boot.index,]
boot.data
```

```
##      Mortality Temperature
## 8      89.2      45.1
## 1     102.5      51.3
## 16     52.5      34.0
## 13     65.1      42.3
## 13.1   65.1      42.3
## 16.1   52.5      34.0
## 1.1    102.5      51.3
## 3      100.4      50.0
## 15     67.3      31.8
## 3.1    100.4      50.0
## 6      95.0      47.8
## 14     68.1      40.2
## 7      88.6      47.3
## 2     104.5      49.9
## 14.1   68.1      40.2
## 12     72.2      43.5
```

4.2.2 Bootstrap confidence interval for correlation, using bivariate sampling

```
set.seed(4123)
cancer <- read.table('C:/Users/sjricht2/Documents/DataSets/Regression/Breast
Cancer Data.txt', header=T)

nboot <- 10000
r.boot <- numeric(nboot)
slope.boot <- numeric(nboot)

for (i in 1:nboot) {
  boot.index <- sample(1:nrow(cancer), replace = TRUE)
  boot.data <- cancer[boot.index,]
  r.boot[i]=cor(boot.data$Mortality, boot.data$Temperature)
}
```

```
cor(cancer$Mortality,cancer$Temperature)
## [1] 0.8748544

quantile(r.boot,c(0.01,.025,.05,.10,.90,.95,.975,0.99))
##          1%          2.5%          5%          10%          90%          95%          97.5%
## 0.7428583 0.7683708 0.7915637 0.8150605 0.9461222 0.9569523 0.9647395
##          99%
## 0.9722582
```

The observed correlation is 0.875, and a 95% confidence interval is (0.768, 0.965).

Extra.

Bootstrap C.I for slope

Case 1: Data consists of a sample of bivariate pairs (X_i, Y_i) .

- 1) Draw a specified number of bivariate bootstrap samples of size n , i.e., sample pairs of observations.
- 2) Compute $\hat{\beta}_{1,b}$, the slope of the regression line, for each bootstrap sample.
- 3) Use the percentile method to construct confidence interval.

Since a pivot quantity exists for the sample slope, a t -pivot interval may also be computed.

The statistic $t = \frac{\hat{\beta}_1 - \beta}{SE(\hat{\beta}_1)}$, where $SE(\hat{\beta}_1) = \sqrt{\frac{MSE}{(n-1)S_X^2}}$, is a pivotal quantity.

Thus:

- 1) Draw a specified number of bivariate bootstrap samples of size n , i.e., sample pairs of observations.

2) Compute $t_b = \frac{\hat{\beta}_{1,b} - \hat{\beta}}{SE(\hat{\beta}_{1,b})}$, where $SE(\hat{\beta}_{1,b}) = \sqrt{\frac{MSE_b}{(n-1)S_{X,b}^2}}$;

- 3) For a given confidence level, $1 - \alpha$, determine the quantiles $t_{b,\alpha/2}$ and $t_{b,1-\alpha/2}$.

- 4) The confidence interval is given by:

$$\hat{\beta}_1 - t_{b,1-\alpha/2}SE(\hat{\beta}_1) < \beta_1 < \hat{\beta}_1 - t_{b,\alpha/2}SE(\hat{\beta}_1).$$

4.2.3 Fixed X sampling for the slope

- May be used for inferences on slope of regression line.

- Assumes $Y = h(X) + \varepsilon$, where $h(X)$ is some function, say a linear function, ε are independent, identically distributed with mean 0, variance σ^2 .
- Idea: Sample errors and then add to the mean.

Steps:

- 1) Compute $\hat{h}(X)$ from the observed sample.
- 2) Compute the errors, $e_i = Y_i - \hat{h}(X_i)$.
- 3) Select a bootstrap sample of n errors (e_i) and compute $Y_i = \hat{h}(X_i) + e_{i,b}, i = 1, \dots, n$.
- 4) Repeat (3) N_b times.

This yields N_b bootstrap samples, each consisting of n pairs, $(X_i, Y_{i,b})$.

- 1) For each bootstrap sample compute

$t_e = \frac{\hat{\beta}_{1,e}}{SE(\hat{\beta}_{1,e})}$, where $\hat{\beta}_{1,e}$ is the slope of the pairs (X_i, e_i) for each bootstrap sample, and

$$SE(\hat{\beta}_{1,e}) = \sqrt{\frac{MSE_e}{(n-1)S_X^2}} = \sqrt{\frac{\sum (e_i - e_{i,b})^2}{(n-1)S_X^2}};$$

- 2) Then for a given confidence level, $1 - \alpha$, determine the quantiles $t_{e,\alpha/2}$ and $t_{e,1-\alpha/2}$.

- 3) The confidence interval is given by $\hat{\beta}_1 - t_{e,1-\alpha/2}SE(\hat{\beta}_1) < \beta_1 < \hat{\beta}_1 - t_{e,\alpha/2}SE(\hat{\beta}_1)$.

Fixed X sampling: Assumes

- Correct function being fit.

- X-values **constants**
- Constant error variance

If these assumptions are questionable, use bivariate sampling. Bivariate sampling will usually result in a larger standard error if assumptions for fixed-X sampling appear violated.

*See example 8.4.1

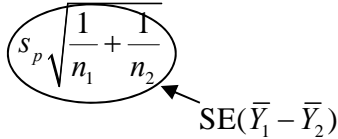
5. Two-Sample Confidence Intervals

Two cases, as in parametric case: Model: $Y_{ij} = \mu_i + \varepsilon_{ij}$

- 1) $\varepsilon_{ij} \sim F(\varepsilon)$ (common error distribution)

Pivot quantity:

$$t = \frac{(\bar{Y}_1 - \bar{Y}_2) - (\mu_1 - \mu_2)}{s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$



Bootstrap interval:

- i) Compute $e_{ij} = Y_{ij} - \bar{Y}_i$ for each observation.
- ii) Select n_1 errors, with replacement, from the set of all errors and assign to 1st sample. Then select n_2 in similar fashion and assign to second sample.

iii) Compute $t_\varepsilon = \frac{\bar{e}_{1,b} - \bar{e}_{2,b}}{s_{\varepsilon p} \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$, $s_{\varepsilon p}^2 = \frac{\sum \sum (\bar{e}_{ij} - \bar{e}_i)^2}{(n_1 + n_2 - 2)}$

- iv) Interval is $(\bar{Y}_1 - \bar{Y}_2) - t_{e,1-\alpha/2} SE(\bar{Y}_1 - \bar{Y}_2) \leq \mu_1 - \mu_2 \leq (\bar{Y}_1 - \bar{Y}_2) - t_{e,\alpha/2} SE(\bar{Y}_1 - \bar{Y}_2)$.

- 2) Unequal error distributions: Select errors within samples.

$$SE(\bar{Y}_1 - \bar{Y}_2) = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}. \text{ Compute } t'_\varepsilon = \frac{\bar{e}_1 - \bar{e}_2}{\sqrt{\frac{s_{\varepsilon 1}^2}{n_1} + \frac{s_{\varepsilon 2}^2}{n_2}}}, \text{ where } s_E^2 = \sum \frac{(e_{ij} - \bar{e}_i)^2}{n_i - 1}$$

Interval is: $(\bar{Y}_1 - \bar{Y}_2) - t_{e,1-\alpha/2} SE(\bar{Y}_1 - \bar{Y}_2) \leq \mu_1 - \mu_2 \leq (\bar{Y}_1 - \bar{Y}_2) - t_{e,\alpha/2} SE(\bar{Y}_1 - \bar{Y}_2)$.

