Resampling-based inference using the `mosaic` package

Daniel Kaplan*
Macalester College
St. Paul, MN

Nicholas J. Horton†
Smith College
Northampton, MA

Randall Pruim‡
Calvin College
Grand Rapids, MI

March 24, 2012

Contents

1 Introduction

This document is intended to describe relatively straightforward ways to undertake a variety of resampling-based inferences through use of the `mosaic` package within R. It is derived from a series of problems posed at USCOTS (United States Conference on Teaching Statistics) 2011 by Robin Lock and colleagues (http://www.causeweb.org/uscots/breakout/breakout3_6.php). One of the goals of the `mosaic` package is to provide elementary commands that can be easily strung together by novices without having to master the esoteric aspects of programming. More information about the package and this initiative can be found at the Project MOSAIC website: www.mosaic-web.org.

*dtkaplan@gmail.com
†nhorton@smith.edu
‡rpruim@calvin.edu

The mosaic operations allow students to implement each of the operations in what George Cobb calls the "3 Rs" of statistical inference: Randomization, Replication, and Rejection (Cobb, 2007). By putting the 3 Rs together in various ways, students learn to generalize and internalize the logic of inference, rather than just to blindly follow formulaic methods.

There's an interesting discussion of the role of simulation in Speed (2011), where he notes the changing role of simulation. It used to be:

> something that people did when they can't do the math. ... It now seems that we are heading into an era when all statistical analysis can be done by simulation.

Arguably, the most important operation in statistics is sampling: ideally, selecting a random subset from a population. Regrettably, sampling takes work and time, so instructors tend to de-emphasize the actual practice of sampling in favor of theoretical descriptions. What's more, the algebraic notation in which much of conventional textbook statistics is written does not offer an obvious notation for sampling.

With the computer, however, these efficiency and notation obstacles can be overcome. Sampling can be placed in its rightfully central place among the statistical concepts in our courses.

Resampling-based inference using permutation testing and bootstrapping are an increasingly important set of techniques for introductory statistics and beyond.

Bootstrapping and permutation testing are powerful and elegant approaches to estimation and testing, respectively that can be implemented even in many situations where asymptotic results are difficult to find or otherwise unsatisfactory (Efron and Tibshirani, 1993; Hesterberg et al 2005). Bootstrapping involves sampling *with* replacement from a population, repeatedly calculating a sample statistic of interest to empirically construct the sampling distribution. Permutation testing for a two group comparison is done by *permuting* the labels for the grouping variable, then calculating the sample statistic (e.g. difference between two groups using these new labels) to empirically construct the null distribution.

2 Background and setup

2.1 R and RStudio

R is an open-source statistical environment that has been used at a number of institutions to teach introductory statistics. Among other advantages, R makes it easy to demonstrate the concepts of statistical inference through randomization while providing a sensible path for beginners to progress to advanced and professional statistics. RStudio (http://www.rstudio.org) is an open-source integrated development environment for R which facilitates use of the system.

2.2 Setup

The mosaic package is available over the Internet and can be installed into R using the standard features of the system (this needs only be done once).

```
install.packages("mosaic")
```

Once installed, the package must be loaded so that it is available (this must be done within each R session). In addition to loading the package, we set the number of digits to display by default as well as the number of simulations to undertake.

```
require(mosaic)
options(digits = 3)
numsim <- 2000
```

This command would typically be provided in a set-up file for students so that it is executed automatically each time an R session is started.

Next we load two of the datasets that will be used in the examples.

```
mustangs <- read.csv("MustangPrice.csv")
sleep <- read.csv("SleepCaffeine.csv")
```

These datasets can also be accessed over the internet using the following `mosaic` commands:

```
mustangs <- fetchData("MustangPrice.csv")
sleep <- fetchData("SleepCaffeine.csv")
```

3 Resampling in different settings

3.1 Bootstrapping a mean (used Mustangs)

A student collected data on the selling prices for a sample of used Mustang cars being offered for sale at an internet website. The price (in $1,000's), age (in years) and miles driven (in 1,000's) for the 25 cars in the sample are given in the [file MustangPrice.csv]. Use these data to construct a 90% confidence interval for the mean price (in $1,000's) of used Mustangs.

First we start by displaying the distribution of values.

```
stem(mustangs$Price)


  The decimal point is 1 digit(s) to the right of the |

  0 | 355778899
  1 | 00223356
  2 | 1235
  3 | 238
  4 | 5
```

The mean price can be calculated as

```
mean(~Price, data = mustangs)
```

```
[1] 16
```

Even though a single trial is of little use, it's a nice idea to have students do the calculation to show that they are (usually) getting a different result than without resampling. One resampling trial can be carried out with

```
mean(~Price, data = resample(mustangs))
```

```
[1] 16.9
```

This generates a new sample (with replacement) of the same size as the original one.

Another trial can be carried out with the command:

```
mean(~Price, data = resample(mustangs))
```

```
[1] 18.3
```

Let's generate five more:

```
trials <- do(5) * mean(~Price, data = resample(mustangs))
trials
```

```
  result
1   16.5
2   19.0
3   12.9
4   18.5
5   13.0
```

Now conduct 2000 resampling trials, saving the results in an object called `trials`:

```
numsim
```

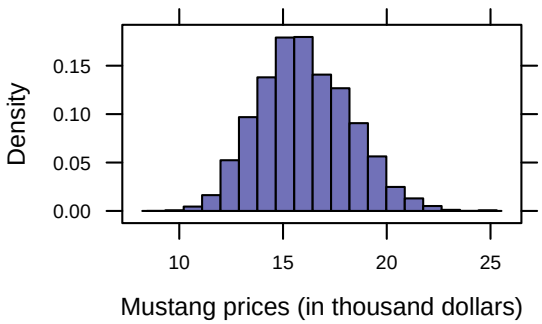
```
[1] 2000
```

```
trials <- do(numsim) * mean(~Price, resample(mustangs))
```

This creates a new set of data with the result from each of the `numsim` = 2000 trials.

Plots of distributions are straightforward, e.g.:

```
xhistogram(~result, data = trials, xlab = "Mustang prices (in thousand dollars)")
```



Calculation of the 90% confidence interval can be done directly.

```
qdata(c(0.05, 0.95), trials$result)
```

```
  5%  95%
12.6 19.7
```

Alternatively, the standard error from this distribution can be used to estimate the 90% margin of error. First calculate the critical value t_* (or z_*) for the appropriate degrees of freedom:

```
tstar <- qt(0.95, df = 24)
zstar <- qnorm(0.95)
```

The resulting margin of error will be

```
tstar * sd(trials$result)
```

```
[1] 3.72
```

```
zstar * sd(trials$result)
```

```
[1] 3.58
```

3.2 Testing a proportion (NFL Overtimes)

The National Football League (NFL) uses an overtime period to determine a winner for games that are tied at the end of regulation time. The first team to score in the overtime wins the game and a coin flip is used to determine which team gets the ball first. Is there an advantage to winning the coin flip? Data from the 1974 through 2009 seasons show that the coin flip winner won 240 of the 428 games where a winner was determined in overtime. Treat these as a sample of NFL games to test whether there is sufficient evidence to show that the proportion of overtime games won by the coin flip winner is more than one half.

If the coin flip result were unrelated to the outcome of the game, the observed 240 game wins out of 428 events would itself be a plausible outcome of a coin flip.

Style 1 Using the built-in binomial distribution operators.

Generate a simulation where each trial is a random sample of 428 games from a world in which the null hypothesis holds true.

```
prop(rbinom(1e+05, prob = 0.5, size = 428) >=
    240)
```

```
   TRUE
0.00739
```

It's very unlikely, if the null were true, that the coin flip winner would win 240 or more times.

Of course, such a calculation can be done directly, but that raises issues such as which tail `pbinom()` is calculating (R always does the left tail) and adjusting the cut-off appropriately

```
pbinom(239, prob = 0.5, size = 428)
```

```
[1] 0.993
```

Style 2 Explicitly simulating a coin flip.

Recognizing that coin flips are a staple of statistics courses, the `mosaic` package offers a random flip operator that does the tabulation for you. Here is one trial involving flipping 428 coins:

```
do(1) * rflip(428)
```

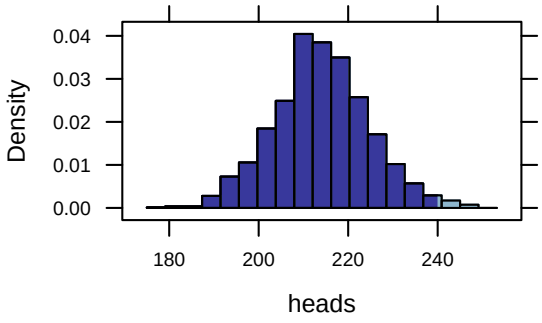
```
    n heads tails
1 428   204   224
```

We'll do 2000 trials, and count what fraction of the trials the coin toss winner (say, "heads") wins 240 or more of the 428 attempts:

```
trials <- do(numsim) * rflip(428)
prop(trials$heads >= 240)
```

```
 TRUE
0.012
```

```
xhistogram(~heads, groups = (heads >= 240),
    data = trials)
```



The observed pattern of 240 wins is not a likely outcome under the null hypothesis. The shading added through the `groups =` option helps to visually reinforce the result.

3.3 Permutation test of means from two groups (sleep and memory)

In an experiment on memory, students were given lists of 24 words to memorize. After hearing the words they were assigned at random to different groups. One group of 12 students took a nap for 1.5 hours while a second group of 12 students stayed awake and was given a caffeine pill. The results below display the number of words each participant was able to recall after the break. Test whether the data indicate a difference in mean number of words recalled between the two treatments.

```
sleep
```

```
      Group Words
1     Sleep    14
2     Sleep    18
3     Sleep    11
4     Sleep    13
5     Sleep    18
6     Sleep    17
7     Sleep    21
8     Sleep     9
9     Sleep    16
10    Sleep    17
11    Sleep    14
12    Sleep    15
13 Caffeine    12
14 Caffeine    12
15 Caffeine    14
16 Caffeine    13
17 Caffeine     6
18 Caffeine    18
19 Caffeine    14
20 Caffeine    16
21 Caffeine    10
22 Caffeine     7
23 Caffeine    15
24 Caffeine    10
```

The Sleep group seems to have remembered somewhat more words on average:

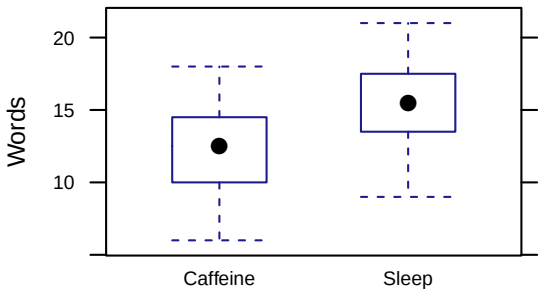
```
mean(Words ~ Group, data = sleep)
```

```
Caffeine    Sleep
    12.2     15.2
```

```
obs <- diff(mean(Words ~ Group, data = sleep))
obs
```

```
Sleep
    3
```

```
bwplot(Words ~ Group, data = sleep)
```



To implement the null hypothesis, scramble the Group with respect to the outcome, Words:

```
diff(mean(Words ~ shuffle(Group), data = sleep))
```

```
Sleep
 1.17
```

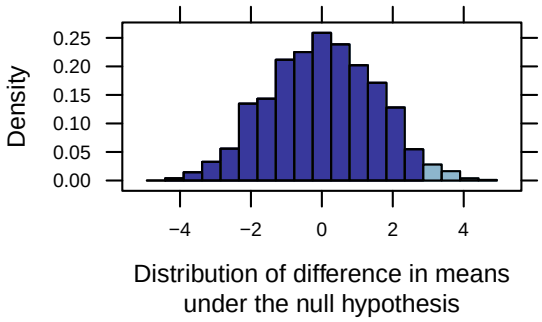
That's just one trial. Let's try again:

```
diff(mean(Words ~ shuffle(Group), data = sleep))
```

```
Sleep
    0
```

To get the distribution under the null hypothesis, we carry out many trials:

```
trials <- do(numsim) * diff(mean(Words ~ shuffle(Group),
    data = sleep))
xhistogram(~Sleep, groups = Sleep >= obs,
    data = trials, xlab = "Distribution of difference in means\nunder the null
hypothesis")
```



We can calculate the one-sided p-value for this test by summing up the number of trials which yielded a result as extreme or more extreme as the observed difference. Here only 0 of the permutations were that large, so the p-value is equal to 0. We conclude that it's unlikely that the two

groups have the same mean word recall back in their respective populations.

3.4 Permutation test of proportions from two groups (HELP RCT)

We can undertake a test of difference in two proportions, in this case, the proportion homeless by gender in the HELP randomized clinical trial.

```
tally(~ homeless | sex, data=HELPrct)
```

```
          sex
homeless   female  male
  homeless  0.374 0.488
  housed    0.626 0.512
  Total     1.000 1.000
```

```
prop(~ homeless | sex, data=HELPrct)
```

```
homeless:female   homeless:male
          0.374           0.488
```

```
obs <- diff(prop(~ homeless | sex, data=HELPrct))
obs  # observed value
```

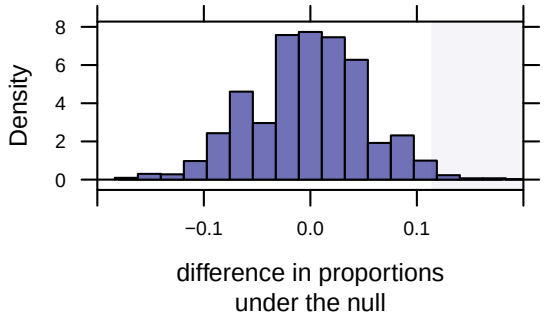
```
homeless:male
        0.115
```

We will use the same general approach to empirically calculate the null distribution by permuting the labels for the grouping variable.

```
# compute permutation distribution
nulldist <- do(numsim) * diff(prop(homeless ~ shuffle(sex), data=HELPrct))
```

```
statTally(obs, nulldist, center = 0, xlab = "difference in proportions\nunder the null")
```

```
Null distribution appears to be asymmetric. (p = 0.0144)


Test statistic applied to sample data =  0.1146


Quantiles of test statistic applied to random data:

    50%     90%     95%     99%
0.00448 0.06566 0.07790 0.11461



Of the random samples


18 ( 0.9 % ) had test stats = 0.1146


17 ( 0.85 % ) had test stats > 0.1146
```



We can also calculate a two-sided p-value:

```
sum(abs(nulldist) >= abs(obs))/numsim
```

```
[1] 0.0385
```

3.5 Bootstrapping a correlation

The data on Mustang prices in Problem #1 also contains the number of miles each car had been driven (in thousands). Find a 95% confidence interval for the correlation between price and mileage.

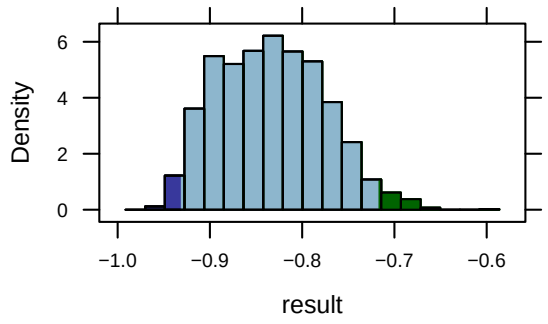
```
with(mustangs, cor(Price, Miles))
```

```
[1] -0.825
```

```
trials <- do(numsim) * with(resample(mustangs),
    cor(Price, Miles))
quantiles <- qdata(c(0.025, 0.975), trials$result)
quantiles
```

```
  2.5%  97.5%
-0.929 -0.718
```

```
xhistogram(~result, data = trials, groups = cut(result,
    c(-Inf, quantiles, Inf)), nbin = 30)
```



3.6 Bootstrapping a regression model

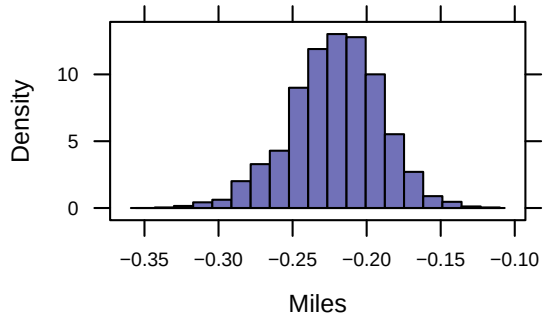
But there's no reason to restrict oneself to the correlation: we can also fit the linear model and consider the coefficients themselves:

```
do(1) * lm(Price ~ Miles, data = mustangs)
```

```
  Intercept  Miles sigma r-squared
1      30.5 -0.219  6.42      0.68
```

```
trials <- do(numsim) * lm(Price ~ Miles, data = resample(mustangs))
xhistogram(~Miles, data = trials)
sd(trials)  # standard errors
```

```
Intercept     Miles     sigma r-squared
   2.9600    0.0306    1.5818    0.0915
```



The predicted average price goes down by 22 ± 6 cents per mile driven.

Using simulations in other ways The basic technology of resampling and shuffling can be used to demonstrate many other concepts in statistics than the generation of confidence intervals and p-values. For example, it is very useful for showing the origins of distributions such as t and F. Similarly, it can be helpful to show students the distribution of p-values under the null hypothesis — students are surprised to see that it's uniform. Seeing this helps them to understand the sense in which the "significance level" refers to a false rejection of the null in a world in which the null

is true.

4 Acknowledgments

Thanks to Sarah Anoke for comments on an earlier draft, as well as to Robin Lock of St. Lawrence University for organizing the session at USCOTS. Project MOSAIC is supported by the US National Science Foundation (DUE-0920350). More information about the package and this initiative can be found at the Project MOSAIC website: www.mosaic-web.org.

5 References

- G. W. Cobb, The introductory statistics course: a Ptolemaic curriculum?, *Technology Innovations in Statistics Education*, 2007, 1(1).
- B. Efron & R. J. Tibshirani, *An introduction to the bootstrap*, 1993, Chapman & Hall, New York.
- T Hesterberg, D. S. Moore, S. Monaghan, A. Clipson & R. Epstein. *Bootstrap methods and permutation tests (2nd edition)*, (2005), W.H. Freeman, New York.
- D. Kaplan, *Statistical modeling: A fresh approach*, http://www.macalester.edu/~kaplan/ISM.
- T. Speed, Simulation, *IMS Bulletin*, 2011, 40(3):18.