Resampling in R: a data-driven example

Instructor: Nicholas G Reich

Overview

The goal of this example is to demonstrate the basics of resampling a dataset using R. Resampling can be performed in many different settings. Common reasons for implementing a resampling strategy include (a) small sample sizes that make reliance on parametric assumptions tenuous, (b) highly irregular or non-normal data, or (c) no closed-form solution for an estimate of a standard error or test statistic. Resampling must be implemented with care, and in situations with complex sampling designs, can be tricky to implement. However, it is a powerful tool, with a robust theoretical foundation and, in many situations, can be easily implemented with few assumptions needed about your data.

We will present the code for a bootstrap analysis and a permutation test.

The data

Our data is a sample of 45 observations. Twenty of these are independent and identically distributed counts from one distribution, f_1 with mean θ_1, and 25 are independent and identically distributed counts from another distribution, f_2 with mean θ_2. We wish to learn about whether the θ_1 and θ_2 are significantly different from one another. Specifically, we wish to ask the following specific questions:

1. with a type-I error rate of .05, do the two distributions have the same mean?
2. what is our uncertainty in our estimate of the difference in means between the two distributions?

The data is as follows

```
y1 <- c(22, 20, 9, 19, 5, 15, 18, 23, 5, 14, 13, 12, 15, 18, 20, 22, 31, 23,
    19, 31)
y2 <- c(27, 7, 18, 15, 1, 16, 42, 8, 1, 22, 27, 4, 9, 5, 5, 2, 4, 5, 14, 9,
    28, 11, 11, 7, 10)
```

We can quickly look at our data (since the number of observations is small, we will use a stem and leaf plot) and obtain our estimate of the mean and standard deviation

```
stem(y1)

##
```

```
##   The decimal point is 1 digit(s) to the right of the |
##
##   0 | 559
##   1 | 234558899
##   2 | 002233
##   3 | 11
```

```
stem(y2)
```

```
##
##   The decimal point is 1 digit(s) to the right of the |
##
##   0 | 1124455577899
##   1 | 0114568
##   2 | 2778
##   3 |
##   4 | 2
```

```
(mean.1 <- mean(y1))
```

```
## [1] 17.7
```

```
(mean.2 <- mean(y2))
```

```
## [1] 12.32
```

```
sd(y1)
```

```
## [1] 7.042
```

```
sd(y2)
```

```
## [1] 10.16
```

A permutation test for the difference in means

One simple method to compare the difference in means of the distributions is just to look at the difference in the observed means. Using the original data, this is calculated as

```
(obs.diff <- mean.1 - mean.2)
```

```
## [1] 5.38
```

(Note that putting parentheses around a command prints the new object. Usually, this is performed 'silently', but sometimes it is useful to print the result.)

However, we are interested in weighing the evidence for/against the null hypothesis that the mean of f_1 and f_2 are the same, i.e. $\theta_1 - \theta_2 = 0$. If the null hypothesis were true, then group assignments shouldn't matter. If we picked 25 of our 45 observations randomly and said they came from f_2 and the rest belonged to f_1, then our observed mean difference should be close to zero. Our observed difference, 5.38, appears to be fairly close to zero, but is it a significant difference? A permutation test can help us answer that question.

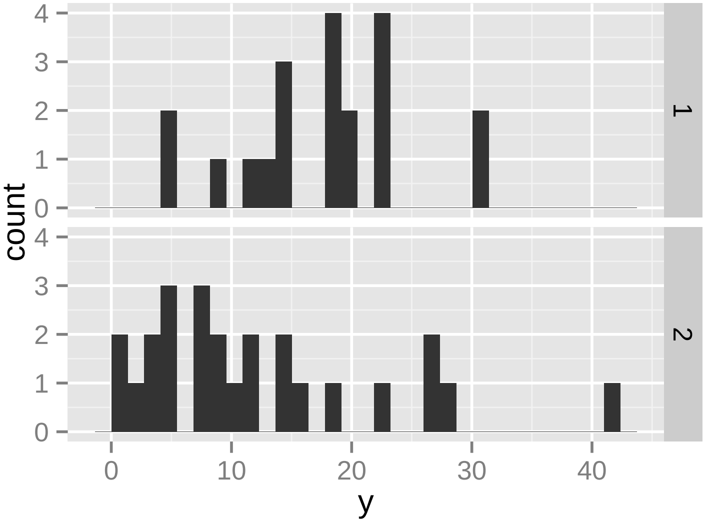
We will randomly permute the group assignments of our data, calculating many possible different groupings of the data. In all of those many groupings, how much of an outlier is our dataset?

Let's begin by getting our data into a format that will make the resampling easy.

```
dat <- data.frame(y = c(y1, y2), grp = rep(1:2, times = c(length(y1), length(y2))))
```

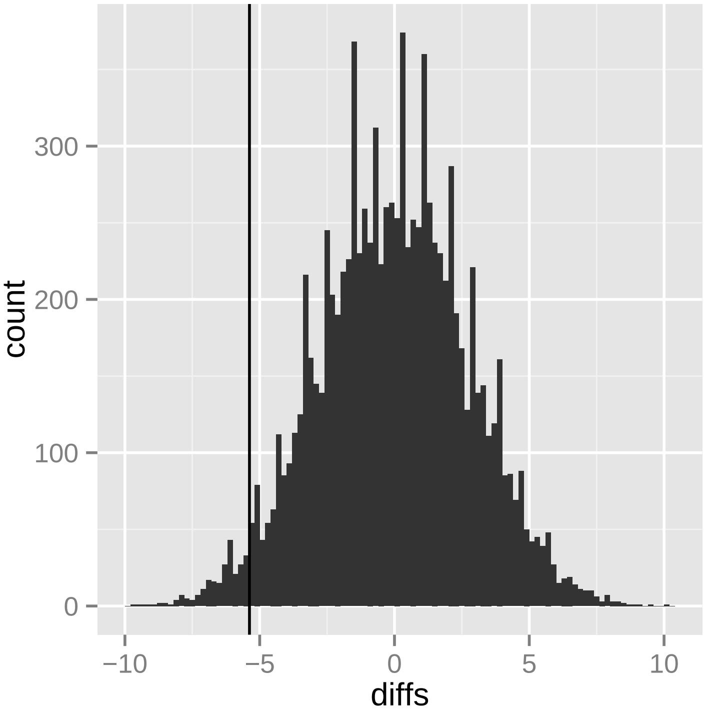
In this format, we can use the `tapply()` function to get a mean by group, and the difference of mean by group. It also lends itself to easy side-by-side graphical comparison.

```
with(dat, tapply(y, grp, mean))

##     1     2
## 17.70 12.32

(obs.diff <- diff(with(dat, tapply(y, grp, mean))))

##     2
## -5.38

require(ggplot2)

## Loading required package:  ggplot2

qplot(y, data = dat, facets = grp ~ .)

## stat_bin:  binwidth defaulted to range/30.  Use 'binwidth = x' to adjust this.

## stat_bin:  binwidth defaulted to range/30.  Use 'binwidth = x' to adjust this.
```



Now, we create a loop to resample the data and a storage vector to store all of the calculated mean differences. We will resample the data 10,000 times and plot the results.

```
n.samps <- 10000
diffs <- rep(NA, n.samps)
for (i in 1:n.samps) {
    grp.sampled <- sample(dat$grp, size = 45, replace = FALSE)
    diffs[i] <- diff(with(dat, tapply(y, grp.sampled, mean)))
}
qplot(diffs, binwidth = 0.2) + geom_vline(xintercept = obs.diff)
```



Next, we wish to calculate the p-value for our permutation test. The one-sided p-value corresponds to the probability that an resampled mean difference was less than the observed difference. The two-sided p-value corresponds to the probability that the absolute difference in means was greater than the absolute value of the observed difference (for more explanation on the two-sided test, see the wikipedia page on permutation tests).

```
(p.val.one.side <- sum(diffs <= obs.diff)/n.samps)

## [1] 0.0268

(p.val.two.side <- sum(abs(diffs) >= abs(obs.diff))/n.samps)

## [1] 0.0508
```

We can also run a parametric model for the difference in means using R's version of the familiar t-test. This test assumes normality, which may or may not be a valid assumption for this data:

```
(ttest.results <- t.test(y ~ grp, data = dat))

##
##  Welch Two Sample t-test
##
## data:  y by grp
## t = 2.093, df = 42.24, p-value = 0.04236
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
```

```
##   0.1943 10.5657
## sample estimates:
## mean in group 1 mean in group 2
##           17.70           12.32
```

This test gives similar results as the permutation test, although it appears to be a little less conservative (i.e. has a lower p-value).

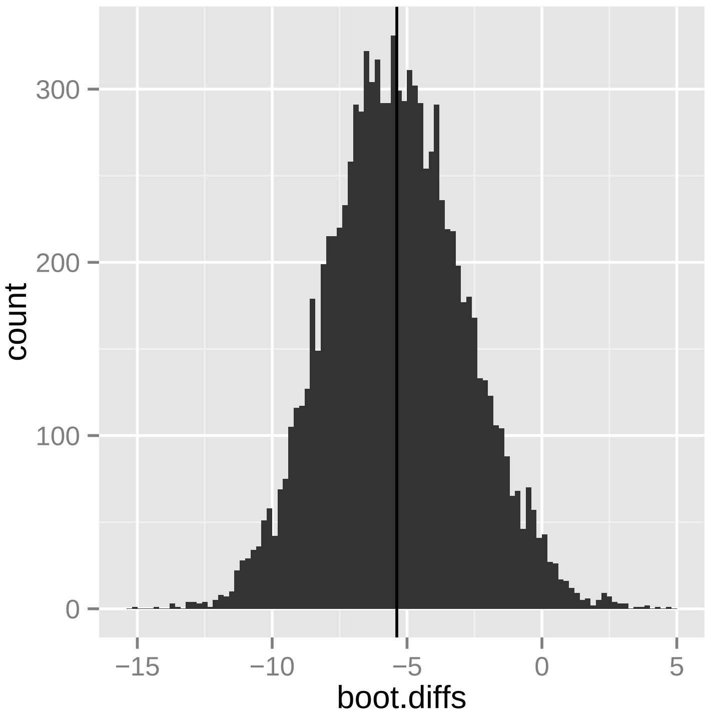
Based on the permutation test results, we conclude that there is marginal statistical evidence (p=0.0508) to suggest that $\theta_1 \neq \theta_2$.

Bootstrap confidence interval for the difference in means

The mechanics of a bootstrap algorithm are, in many cases, very similar to a permutation test, in that they both utilize resampling. The key conceptual difference between the two methods is that a permutation test can help you test a hypothesis where a bootstrap algorithm can help you characterize your uncertainty in the estimate of a parameter. Additionally, from an operational perspective, a bootstrap sample of your data samples with replacement while a permutation test just shuffles the labels.

Rather than using the test statistics to calculate a p-value, as we did in the permutation test, we will use the estimates of the differences of means ($\theta_1 - \theta_2$) and use the middle p% of those numbers as the p% confidence interval.

```
boot.diffs <- rep(NA, n.samps)
for (i in 1:n.samps) {
    ## note how I sample to ensure that the 20/25 split is maintained
    boot.sample <- c(sample(20, replace = TRUE), sample(21:45, replace = TRUE))
    boot.dat <- dat[boot.sample, ]
    boot.diffs[i] <- diff(with(boot.dat, tapply(y, grp, mean)))
}
qplot(boot.diffs, binwidth = 0.2) + geom_vline(xintercept = obs.diff)
```



```
boot.quants <- quantile(boot.diffs, c(0.01, 0.025, 0.05, 0.1, 0.5, 0.9, 0.95,
    0.975, 0.99))
round(boot.quants, 2)

##      1%   2.5%     5%    10%    50%    90%    95%  97.5%    99%
## -11.02 -10.21  -9.40  -8.57  -5.45  -2.06  -1.07  -0.23   0.65
```

Note that bootstrapping does not change our point estimate of the estimated difference in means. Based on the quantiles shown above, the bootstrap 95% confidence interval for the difference in means would be -10.21 to -0.23. As we saw in the permutation test analysis, these results suggest that we have only marginal evidence to suggest that the θ_1 is different from θ_2, or that the two groups have different means.