Resampling Methods
A Practical Guide to Data Analysis
Phillip I. Good
- short review of chapter 1 and 2 -

Magda Mielczarek
16.12.2015
CHAPTER 1: Software for Resampling

Two approaches:

• using of a menu-driven program such as
  • Excel, S-Plus, Stata, StatXact, Testimate, etc.

• programming in a computer language such as:
  • C++, R, Resampling Stats, SAS, etc.
CHAPTER 1: Software for Resampling

• using of a menu-driven program such as:
  • Excel, S-Plus, Stata, StatXact, Testimate, etc.

1.9 S-PLUS®

S-Plus is a menu-driven variant of R aimed at corporate users. It’s expensive but includes more built-in functions, the ability to design and analyze sequential trials, a huge library of resampling methods, plus a great deal of technical support. Check it out at http://www.insightful.com/contactus/request_cd.asp
CHAPTER 1: Software for Resampling

• programming in a computer language such as:
  • C++, R, Resampling Stats, SAS, etc.

1.6 R

R is a do-it-yourself programming language specifically designed for use by statisticians. This means that functions like mean(), quantiles(), binom(), glm(), plot(), sample(), and tree() are precompiled and ready to add your own statistical functions. You can visit http://www.cran.r-project.org/ and obtain many add-on boards from the same Internet address. Many resampling functions are available for download.

1.8 SAS®

SAS is a highly expensive menu-driven program best known for its ability to build customized tables and graphs. Widely available among corporate users, resampling methods can be added via a macro language. The cumbersome nature of this language, a giant step backward from the C++ in which SAS is written, makes it extremely difficult and time consuming to write and debug your own resampling methods. SAS Proc MULTTEST is recommended for obtaining accurate significance levels for multiple tests.
CHAPTER 2: Estimating Population Parameters

Content:

• usage of a bootstrap to estimate the precision of an estimate and to obtain confidence intervals for population parameters and statistics

• computer code is provided
  ➔ usage of resampling methods in practice
CHAPTER 2: Estimating Population Parameters

• **Samples and Populations**

• It is not possible to measure traits in the entire population

• … the method of examination is prohibitively expensive or time-consuming or both

• … the method of examination is destructive

• Treat the sample as if it were the population
CHAPTER 2: Estimating Population Parameters

• Bootstrap

• treat the original sample of values as a stand-in for the population and to resample from it repeatedly, with replacement, computing the desired estimate each time
(primitive) bootstrap example

• 22 observations on the **heights** of sixth-grade students
  137.0 138.5 140.0 141.0 142.0 143.5 145.0 147.0 148.5 150.0 153.0
  154.0 155.0 156.5 157.0 158.0 158.5 159.0 160.5 161.0 162.0 167.5

• Sampling with replacement
  138.5 138.5 140.0 141.0 141.0 143.5 145.0 147.0 148.5 150.0 153.0
  154.0 155.0 156.5 157.0 158.5 159.0 159.0 159.0 160.5 161.0 162.0
  … repeat 100 times
(primitive) bootstrap example

- Sample var = 76.7
- 100 bootstrap var = 47.4 - 115.6 (mean = 71.4)
- "They provide a feel for what might have been had we sampled repeatedly from the original population"

→ the bootstrap can be used to determine the precision of any estimator

Fig. 2.3. Boxplot and strip chart of variances of 100 bootstrap samples.
### Programming the bootstrap in R

It selects 100 bootstrap samples from the classroom data and then produces a boxplot and stripchart of their variances.

```r
class = c(141, 156.5, 162, 159, 157, 143.5, 154, 158, 140, 142, 150, 148.5, 138.5, 161, 153, 145, 147, 158.5, 160.5, 167.5, 155, 137)

n = length(class)  # record group size
N = 100             # set number of bootstrap samples
stat = numeric(N)  # create a vector in which to store the results

# set up a loop to generate a series of bootstrap samples
for (i in 1:N) {
  classB = sample(class, n, replace=T)  # bootstrap sample counterparts to observed samples are denoted with "B"
  stat[i] = var(classB)
}

boxplot(stat)
stripchart(stat)
```
Estimating Bias

- **bias** = amount by which its expected value differs from the quantity to be estimated
- (sixth-graders’ heights) **bias** = 76.7 - 71.4 = 5.3

- **E(X)** - expected or mean value of a random variable X
- An estimate θ[X] based on a sample is also a random variable
- θ is the population parameter we are trying to estimate

- **Bias** of θ[X] is \( b = E(\theta[X] - \theta) \)
- A **bootstrap estimate** for the bias \( b \) of θ[X] is given by \( b^* = \sum_i (\theta_i^* - \theta[X]) / k \)
- \( \theta_i^* \) is the \( i \)th bootstrap sample estimate of \( \theta \) for \( 1 \leq i \leq k \).
Biased sample example - small-scale clinical trial

• A new hormone medicine is the “equivalent” of the new homone medicine
• Crossover trial, each of eight patients received in random order each of the following:
  → new medicine
  → old medicine
  → placebo (manufactured at the new medicine)
• To establish equivalence $|\theta/\mu| \leq 0.20$ where $\theta = E(\text{new}) - E(\text{old})$ and $\mu = E(\text{old}) - E(\text{placebo})$

… but an estimate is biased, both because it is:
• a ratio
• the same factor $E(\text{old})$ appears in both the numerator and the denominator
Biased sample example - small-scale clinical trial

• How large is the bias in the present case?

• \( \theta/\mu = -0.07 \) (FDA’s criteria of 0.20) … but it is **biased**!

• Efron and Tibshirani (1993) generated 400 bootstrap samples and found the bootstrap estimate of bias to be only **0.0043**

• … and still < 0.20

• „Regardless, the bootstrap is notoriously unreliable for small samples and we would be ill advised to draw conclusions without additional data”

---

**Table 2.1. Patch Data Summary**

<table>
<thead>
<tr>
<th>Subject</th>
<th>Old-Placebo</th>
<th>New-Old</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8406</td>
<td>-1200</td>
</tr>
<tr>
<td>2</td>
<td>2342</td>
<td>2601</td>
</tr>
<tr>
<td>3</td>
<td>8187</td>
<td>-2705</td>
</tr>
<tr>
<td>4</td>
<td>8459</td>
<td>1982</td>
</tr>
<tr>
<td>5</td>
<td>4795</td>
<td>-1290</td>
</tr>
<tr>
<td>6</td>
<td>3516</td>
<td>351</td>
</tr>
<tr>
<td>7</td>
<td>4796</td>
<td>-638</td>
</tr>
<tr>
<td>8</td>
<td>10238</td>
<td>-2719</td>
</tr>
<tr>
<td>average</td>
<td>6342</td>
<td>-452.3</td>
</tr>
</tbody>
</table>
Confidence Intervals
Percentile Bootstrap

• … we will always be in error (unless we can sample the entire population)

• confidence interval - estimate lies between some minimum and some maximum value with a pre-specified probability

• constructing confidence intervals from the bootstrap distribution

• 90% confidence interval for the variance of sixth-graders’ heights, we might exclude 5% of the bootstrap values from each end of the boxplot

• confidence interval is [52,95]

“Note that our original point estimate of 76.7 is neither more nor less likely than any other value in the interval [52,95]”
Bias-Corrected Bootstrap Confidence Interval

- huge improvement over the percentile bootstrap
- „The idea behind these intervals comes from the observation that percentile bootstrap intervals are most accurate when the estimate is symmetrically distributed about the true value of the parameter and the tails of the estimate’s distribution drop off rapidly to zero. In other words, when the estimate has an almost-normal distribution.”

- ”Suppose $\theta$ is the parameter we are trying to estimate, $\hat{\theta}$ is the estimate, and we are able to come up with a monotone increasing transformation $t$ such that $t(\hat{\theta})$ is normally distributed about $t(\theta)$. We could use this normal distribution to obtain an unbiased confidence interval, and then apply a back-transformation to obtain an almost-unbiased confidence interval.”

- ”The formula is complicated (…) already incorporated in several computer programs”
install.packages("boot")  # only once

library(boot)
f.median <- function(y, id) {
  median(y[id])
}

boot.ci(boot(class, f.median, 400), conf = 0.90)