The Bootstrap - Numerical Procedures and Applications

Erin Sprünken
Humboldt-Universität zu Berlin

20.11.2020

Outline

Motivation

An introductory example: Consider you are a researcher at Charité. Your colleagues and you are conducting research regarding a new treatment for lung cancer. In order to find out whether it is effective you need to undertake a trial with test subjects. However, for reasons you are not allowed to conduct a large-sample study. This reasons include the unknown (side-)effects as well as ethical reasons. Now, you have conflicting interests: On the one hand you want to have a large sample to get valid results and inference, on the other hand you are not allowed to. Your supervisor allows you to test the treatment among 10 test subjects. How can you get valid results?

Answer: Resampling Techniques

General Applications

- Small sample sizes
- Asymptotic results
- Validation
- Machine Learning techniques
- Numerical solution to problems without analytical solution

Other resampling techniques:

- Jackknife (precedes Bootstrap)
- Cross-Validation (famous in Machine Learning)
- Permutation (famous in two-sample problems)

Bootstrap Approach

Pseudocode (General)

1. Specify number of bootstrap iterations $nboot$
2. Fix Data $\mathbf{X} = (X_1, ..., X_n)$
3. For (i in 1:nboot)
   1. Sample $\mathbf{X_i^*} = (X_1^*, ...X_n^*)$ from $\mathbf{X}$
   2. Compute Statistic of interest $\Psi(\mathbf{X_i^*})$
   3. Save Statistic of interest at position i in $\mathbf{\Psi} = (\Psi_1, ..., \Psi_i, ..., \Psi_{nboot})$
4. Use methods of statistical inference on the empirical distribution of $\mathbf{\Psi}$

Why does that work?

We compute the bootstrap distribution. Since, given the fixed $\mathbf{X}$ the statistics of interest are random variables and we compute a sequence of random variables, we obtain an empirical distribution $\hat{F}_n$ of our variable of interest. Probability Theory taught us, that

$$\hat{F}_n(\psi) \overset{\mathbb{P}-a.s.}{\to} F(\psi).$$

How do we bootstrap? My paper will focus on Nonparametric Bootstrap and (Rademacher-)Wild Bootstrap.

How does that work?

Nonparametric Bootstrap

- Fix Data $\mathbf{X} = (X_1, ..., X_n)$
- In each iteration: Sample with replacement, such that $\mathbb{P}(X_1^* = X_1) = n^{-1}$
- For valid asymptotic results this probability condition is necessary!
- Each iteration consists of a bootstrap sample $\mathbf{X}^* = (X_1^*, ..., X_n^*)$

(Rademacher-) Wild Bootstrap

- Fix Data $\mathbf{X} = (X_1, ..., X_n)$
- Center Data $\mathbf{Z} = (Z_1, ..., Z_n)$ where $Z_k = X_k - \bar{X}$
- In each iteration: Generate (sample) i.i.d. weights $\mathbf{w} = (w_1, ..., w_n)$ such that

$$\mathbb{E}[w_i] = 0, \quad Var(w_i) = 1 \qquad \forall i$$

- Generate Bootstrap sample $\mathbf{X}^* = (X_1^*, ..., X_n^*)$, where $X_i^* = w_i \cdot Z_i$
- Rademacher Distribution: $\mathbb{P}(w_i = 1) = \mathbb{P}(w_i = -1) = 0.5$

One-sample t-Test

Problem

We want to test whether a population has a specific mean infering from a sample:

$$\begin{aligned} H_0 : \quad & \mu = \mu_0 \\ H_1 : \quad & \mu \neq \mu_0 \end{aligned}$$

We infer from the sample mean.

Statistic

$$T(\mathbf{X}) = \frac{\bar{X} - \mathbb{E}[\bar{X}]}{Var(X)}\sqrt{n}$$

Bootstrapping the t-Test

Pseudocode t-Test

1. Specify number of bootstrap iterations $nboot$
2. Fix Data $\mathbf{X} = (X_1, ..., X_n)$
3. For (i in 1:nboot)
   1. Sample$\mathbf{X}^* = (X_1^*, ..., X_n^*)$ from $\mathbf{X}$
   2. Compute Statistic $T_i(X^* \mid \mathbf{X})$
4. Compute True Statistic $T(\mathbf{X})$
5. Decide whether to reject by comparing $T(\mathbf{X})$ to critical quantile of Bootstrap Distribution $\hat{T}_n$

Bootstrapping the t-Test

Resampling Test Statistic

$$
\begin{aligned}
T(X^* \mid \mathbf{X}) &= \frac{\bar{X}^* - \mathbb{E}[\bar{X}^* \mid \mathbf{X}]}{\hat{\sigma}^*}\sqrt{n} \\
\bar{X}^* &= \frac{1}{n}\sum_{k=1}^{n} X_k^* \\
(\hat{\sigma}^*)^2 &= \frac{1}{n-1}\sum_{k=1}^{n}(X_k^* - \bar{X}^*)^2 \\
\mathbb{E}[\bar{X}^* \mid \mathbf{X}] &= \mathbb{E}\left[\frac{1}{n}\sum_{k=1}^{n} X_k^* \middle| \mathbf{X}\right] \\
\mathbb{E}[X_k^* \mid \mathbf{X}] &= \bar{X}
\end{aligned}
$$

Simulation and Computation

What is of interest?

- Accuracy
- Speed and Complexity
- Type I and (some) Type II Error

Simulation Settings

- Different distributions such as Normal, χ^2, Exponential, ...
- Different Datasets (Regression)
- Different Implementations (R, C++ (with Rcpp), C)
- Different Number of Bootstrap Iterations
- Different Bootstrap Versions (Nonparametric vs. Wild)

What is left out?

- No Parametric Bootstrap
- Only one version of Wild Bootstrap (Rademacher)
- Regression Problem not available for Wild Bootstrap

What is of interest?

Accuracy

$$\begin{aligned} Bias &= \mathbb{E}[\hat{\theta} - \theta] \\ Variance &= \mathbb{E}\left[(\hat{\theta} - \mathbb{E}[\theta])^2\right] \\ \Rightarrow MSE &= Variance + Bias^2 \end{aligned}$$

Speed and Complexity

- Complexity is qualitative: How hard is it to write or read the code
- Speed is quantitative: How long do the computations take

Type I and Type II Error

$$\begin{aligned} \alpha &= \mathbb{P}(H_1 \mid H_0) \\ 1 - \beta &= \mathbb{P}(H_0 \mid H_1) \end{aligned}$$

Computational Issues and Implementation

Best Practice and it's importance in R:
R is a very handy language regarding statistics. Why should we even stick to best practices such as matrix algebra, if the code works it works, right?

Wrong!

R is a very-high-level language. What does that mean? From low to high we have:

1. Machine Code
2. Assembler
3. C, FORTRAN
4. R, Python, ...

Implementations in R which do not follow Best Practice can lead to very inefficient programs which can cost a huge amount of time. This is, because the computer has to go all the steps from R to machine Code. See the following examples for a simple bootstrap version of the one-sample t-Test.

Example: Not Best Practice R

Why is this not best practice? We make use of for loops and ignore the matrix-structure of R

Code main part of t-Test

```
T.x     = c()
t.true = (mean(x) - mu.0) / sqrt(var(x)) * sqrt(length(x))

for(i in 1:nboot){
    x.boot = sample(x, size = length(x), replace = T)
    t.boot = (mean(x.boot) - mean(x)) /
              sqrt(var(x.boot))*sqrt(length(x))
    T.x = c(T.x, t.boot)
}
```

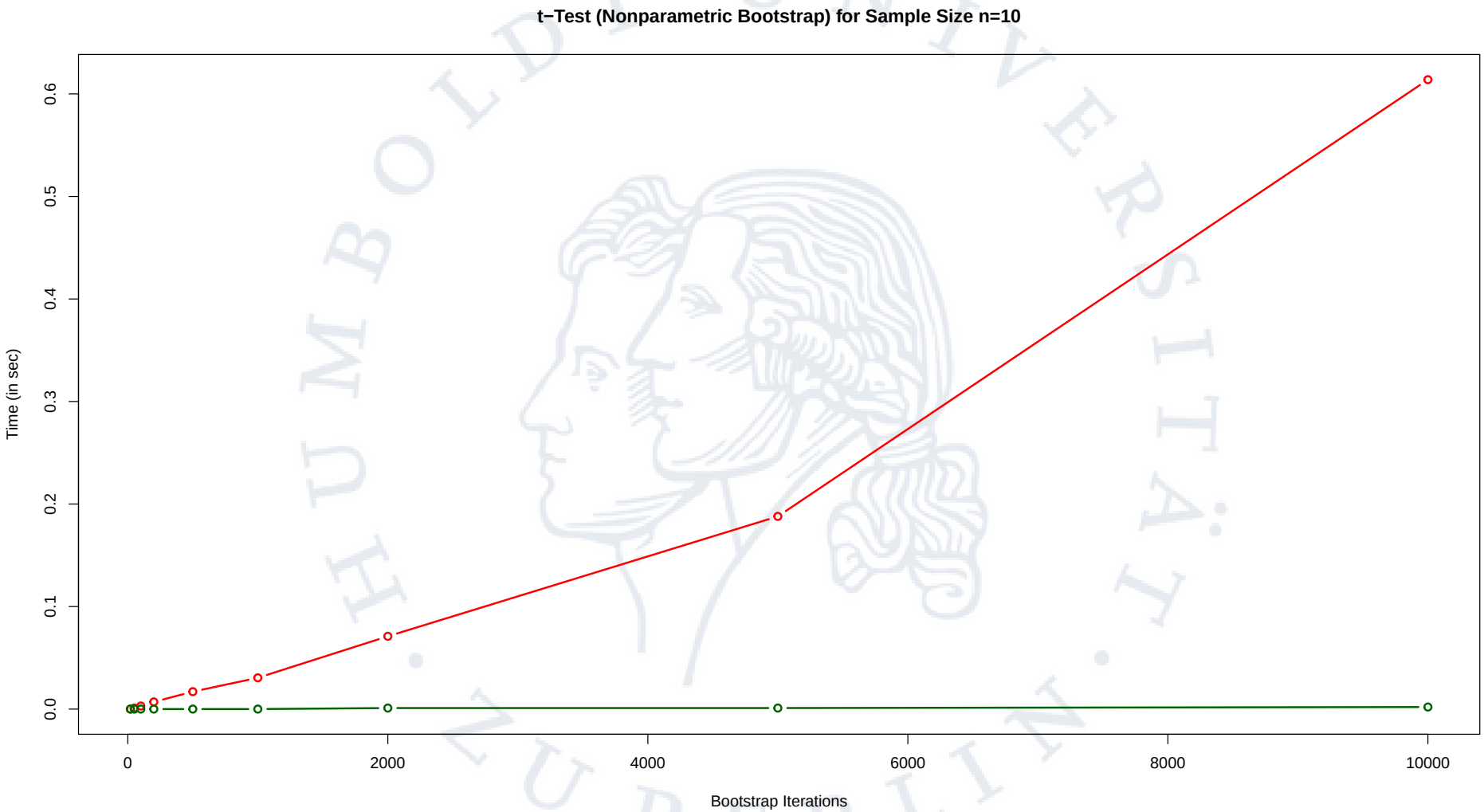
Example: Best Practice R

In this example we exploit the matrix-structure of R resulting in a far more efficient computation. Some of you may know the usage of the apply-family to avoid for-loops, but matrix-algebra will usually be superior to apply in terms of efficency. The reason is, that apply is yet another for-loop in R. Matrix-functions in R are computed in C and thus highly competetive when it comes to speed.

Code main part of t-Test - Best Practice

```
T.x     = numeric(nboot)
n       = length(x)
t.true  = (mean(x) - mu.0) / sqrt(var(x)) * sqrt(n)
x.boot  = matrix(sample(x, size = n*nboot, replace = T),
          ncol = nboot)
x.mean  = colMeans(x.boot)
x.var   = (colSums(x.boot^2) - n*x.mean^2)/(n-1)
T.x     = sqrt(n) * (x.mean - mean(x)) / sqrt(x.var)
```

Results: Best Practice vs. Not Best Practice



Red Line: Loop-Version, Green Line: Matrix-Version

What can be done beyond my seminar paper?

- Multiple Contrast Tests
- Effect-Size Tests
- Multivariate-Behrens-Fisher Problem
- Other Tests (Anderson-Darling, Shapiro-Wilk, ...)
- Timeseries Bootstrapping
- Bootstrapping in Portfolio Analysis

Literature

- Bootstrap Methods: Another Look at the Jackknife (Efron, 1979)
- Estimated Sampling Distributions: The Bootstrap and Competitors (Beran, 1982)
- Jackknife, Bootstrap and Other Resampling Methods in Regression Analysis (Wu, 1986)
- Bootstrap, wild bootstrap and asymptotic normality (Mammen, 1992)
- Bootstrapping and permuting paired t-test type statistics (Konietschke and Pauly, 2014)