Bootstrap CIs

Jo Hardin

10/7/2020

There are many built in functions in R (and Python, Matlab, Stata, etc. for that matter) which will bootstrap a dataset and create any of a number of standard bootstrap intervals. However, this file will bootstrap using first principles in order to see the inner workings of the bootstrap process.

Example: heroin survival time

- Hesketh and Everitt (2000) report on a study by Caplehorn and Bell (1991) that investigated the times that heroin addicts remained in a clinic for methadone maintenance treatment.
- The data include the amount of time that the subjects stayed in the facility until treatment was terminated (column 4).
- For about 37% of the subjects, the study ended while they were still the in clinic (status=0).
- Their survival time has been truncated. For this reason we might not want to estimate the mean survival time, but rather some other measure of typical survival time. Below we explore using the median as well as the 25% trimmed mean. (From ISCAM Chance & Rossman, Investigation 4.5.3)

```
heroin <- readr::read_table2("http://www.rossmanchance.com/iscam2/data/heroin.txt")
names(heroin)
```

```
## [1] "id"     "clinic" "status" "times"  "prison" "dose"
```

```
head(heroin)
```

```
## # A tibble: 6 x 6
##      id clinic status times prison  dose
##   <dbl>  <dbl>  <dbl> <dbl>  <dbl> <dbl>
## 1     1      1      1   428      0    50
## 2     2      1      1   275      1    55
## 3     3      1      1   262      0    55
## 4     4      1      1   183      0    30
## 5     5      1      1   259      1    65
## 6     6      1      1   714      0    55
```

```
obs.stat <- heroin %>%
  summarize(tmeantime = mean(times, trim=0.25)) %>% pull()
obs.stat
```

```
## [1] 378.3
```

Bootstrapping the data

```
set.seed(4747)
heroin.bs<-heroin %>% sample_frac(size=1, replace=TRUE)

heroin.bs %>%
  summarize(tmeantime = mean(times, trim=0.25)) %>% pull()
```

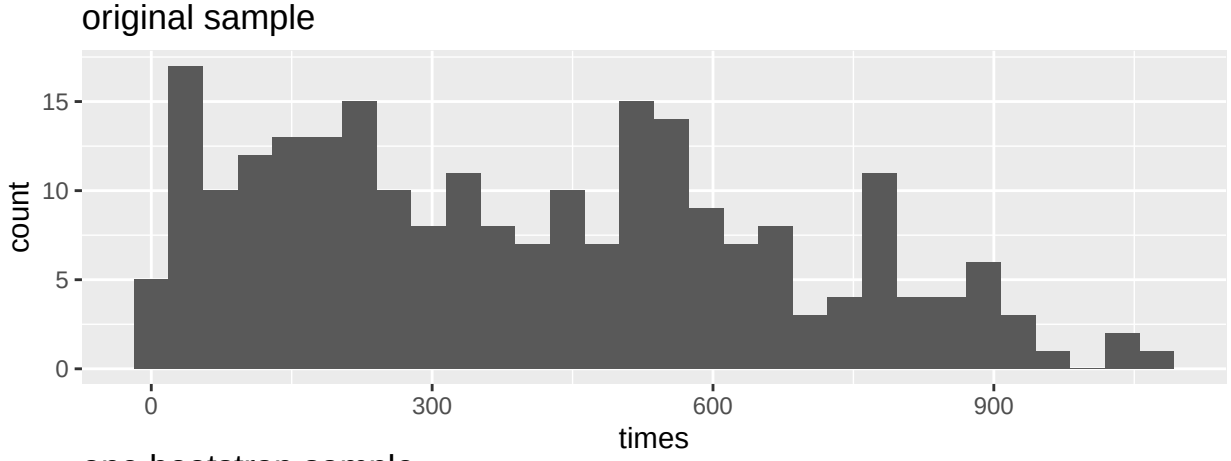
```
## [1] 372.2583
```

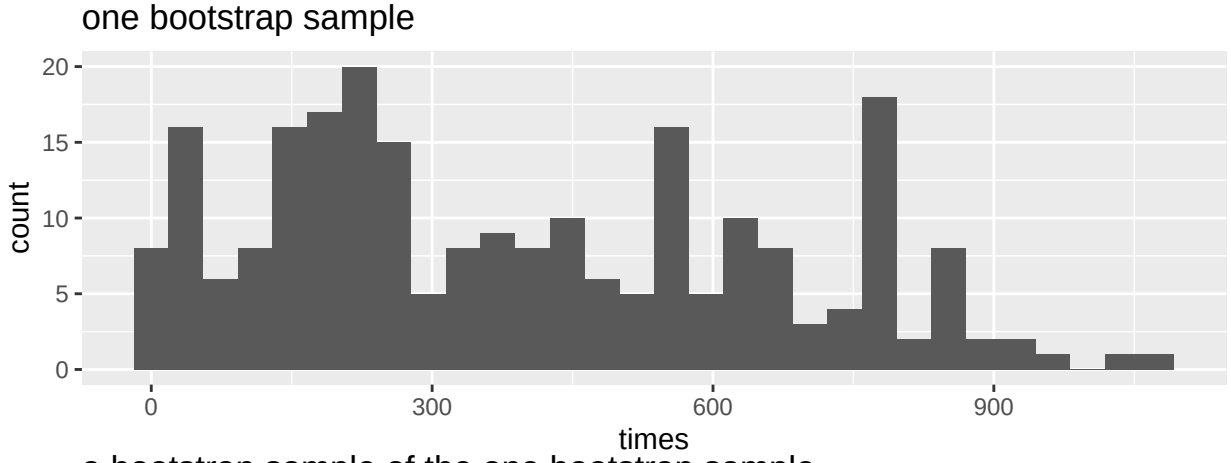
Creating a sampling distribution for the trimmed mean

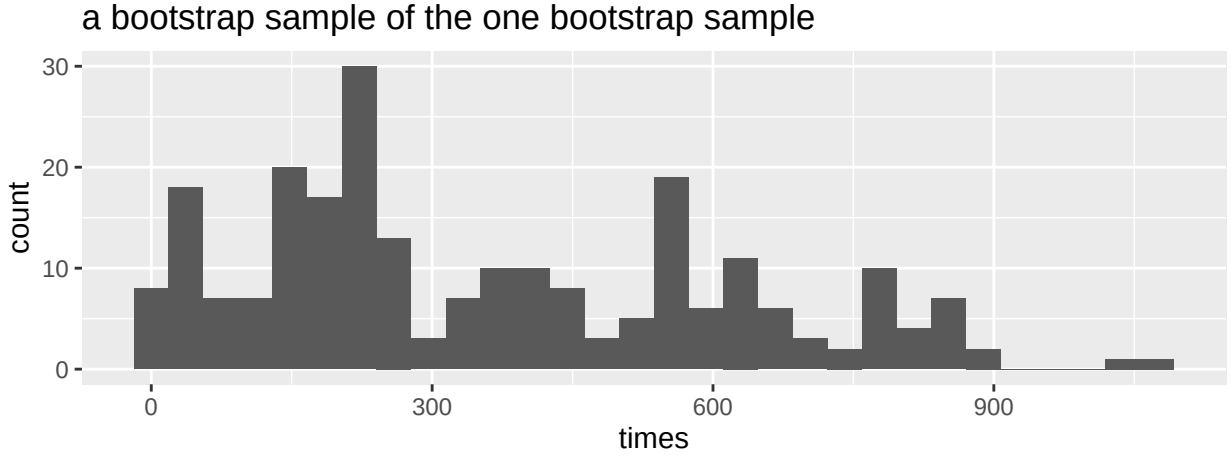
```
bs.test.stat<-c()    # will eventually be B long, check after you run everything!
bs.sd.test.stat<-c() # will eventually be B long, check after you run everything!

B <- 500
M <- 100
set.seed(4747)
```

```
for(b in 1:B){
    heroin.bs<-heroin %>% sample_frac(size=1, replace=TRUE)
    bs.test.stat<-c(bs.test.stat,
                heroin.bs %>%
                  summarize(tmeantime = mean(times, trim = 0.25)) %>% pull())

        bsbs.test.stat <- c() # refresh the rs test statistics

    for(m in 1:M){
    heroin.bsbs<-heroin.bs %>% sample_frac(size=1, replace=TRUE)
    bsbs.test.stat <- c(bsbs.test.stat,
                    heroin.bsbs %>%
                      summarize(tmeantime = mean(times, trim = 0.25)) %>% pull())
    }
    bs.sd.test.stat<-c(bs.sd.test.stat, sd(bsbs.test.stat))
}
```

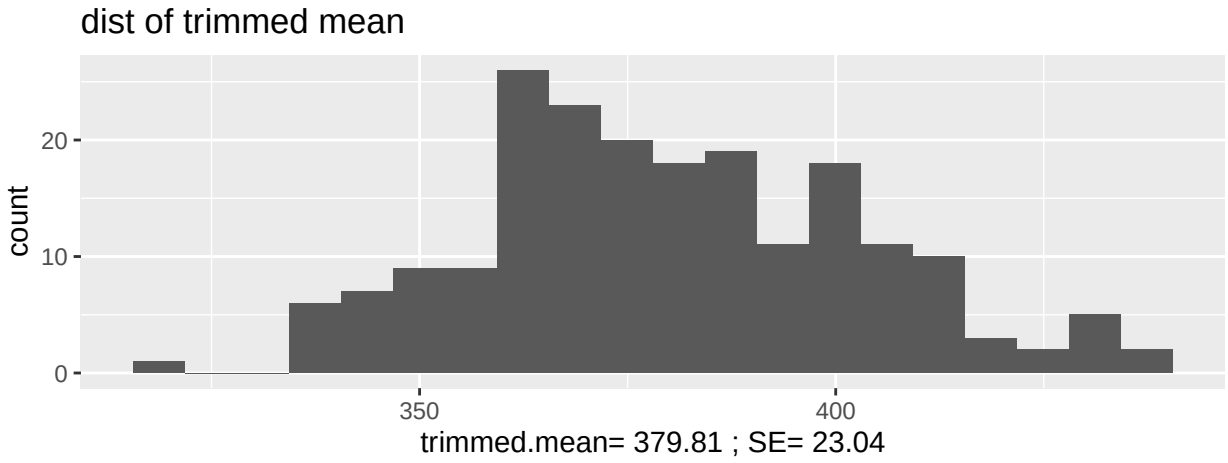
What do the data distributions look like?



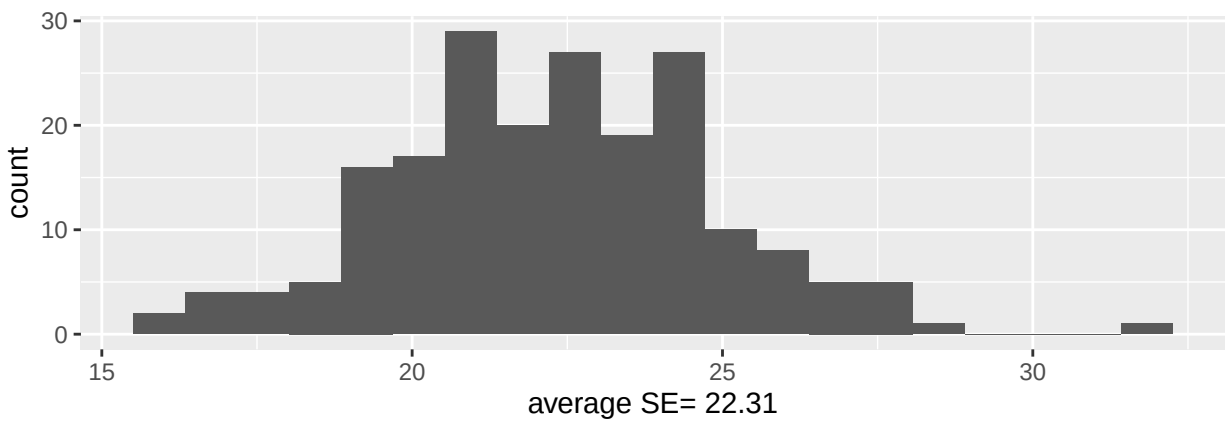




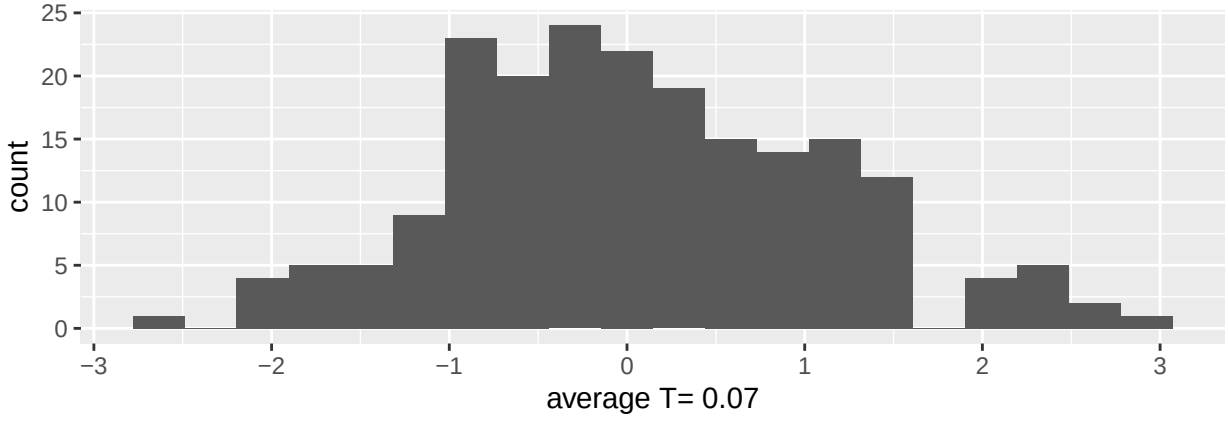
What do the sampling distributions look like?



What is the distribution of the SE of the statistic?



What is the distribution of the T statistics?



95% normal CI with BS SE

```
obs.stat +
  qnorm(c(.025,.975))*
  sd(bs.test.stat)
```

```
## [1] 333.1345 423.4655
```

95% Bootstrap-t CI

Note that the t-value is needed (which requires a different SE for each bootstrap sample).

```
bs.t.hat<-(bs.test.stat - obs.stat)/bs.sd.test.stat

bs.t.hat.95 = quantile(bs.t.hat, c(.975,.025))

obs.stat - bs.t.hat.95*sd(bs.test.stat)
```

```
##    97.5%     2.5%
## 336.5108 426.8502
```

95% Percentile CI

```
quantile(bs.test.stat, c(.025, .975))
```

```
##     2.5%    97.5%
## 339.3323 431.0694
```