Stats506: hw2_4

David Li

October 22, 2017

Stats506: Problem 4 Code
Data Used:
Submitted by: David Li

We are looking to compare coverage probabilities and width of confidence intervals for the median with 4 different distributions and 3 different sample sizes.
Most of the computationally intensive methods are embedded in the functions, so I will just display the tables

```
# Standard Normal Distribution, for sample sizes 50, 250, and 1000 (mean = 0, sd = 1)
a1 = Compute_median_cvrg_prob(samplesize = 50, intervalrep = 200, dist = rnorm(n = 50, mean = 0, sd = 1
a2 = Compute_median_cvrg_prob(samplesize = 250, intervalrep = 200, dist = rnorm(n = 250, mean = 0, sd =
a3 = Compute_median_cvrg_prob(samplesize = 1000, intervalrep = 200, dist = rnorm(n = 1000, mean = 0, sd

a4 = Compute_median_95CI_width(samplesize = 50, intervalrep = 200, bigdist = rnorm(n = 50*200, mean = 0
a5 = Compute_median_95CI_width(samplesize = 250, intervalrep = 200, bigdist = rnorm(n = 250*200, mean =
a6 = Compute_median_95CI_width(samplesize = 1000, intervalrep = 200, bigdist = rnorm(n = 1000*200, mean

a7 = Compute_robust_cvrg_prob(samplesize = 50, intervalrep = 200, dist = rnorm(n = 50, mean = 0, sd = 1
a8 = Compute_robust_cvrg_prob(samplesize = 250, intervalrep = 200, dist = rnorm(n = 250, mean = 0, sd =
a9 = Compute_robust_cvrg_prob(samplesize = 1000, intervalrep = 200, dist = rnorm(n = 1000, mean = 0, sd

a10 = Compute_robust_estimator_CI_width(samplesize = 50, intervalrep = 200, bigdist = rnorm(n = 50*200,
a11 = Compute_robust_estimator_CI_width(samplesize = 250, intervalrep = 200, bigdist = rnorm(n = 250*20
a12 = Compute_robust_estimator_CI_width(samplesize = 1000, intervalrep = 200, bigdist = rnorm(n = 1000*
#Table
normtable = createtable(a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11,a12, "Standard Normal Dist", "mean = 0, sd =
normtable
```

```
##                                Convergence Probability
##                                "Standard Normal Dist"
## Bootstrap, Size = 50           "0.855"
## Bootstrap, Size = 250          "0.015"
## Bootstrap, Size = 1000         "0.925"
## Robust Estimation, Size = 50   "0.885"
## Robust Estimation, Size = 250  "0.55"
## Robust Estimation, Size = 1000 "0.855"
##                                Width of Confidence Interval
##                                "mean = 0, sd = 1"
## Bootstrap, Size = 50           "0.690510438776347"
## Bootstrap, Size = 250          "0.309720009618319"
## Bootstrap, Size = 1000         "0.154674604734466"
## Robust Estimation, Size = 50   "0.544950058568191"
## Robust Estimation, Size = 250  "0.249722744070916"
## Robust Estimation, Size = 1000 "0.124364775469699"
```

```
# Exponential Distribution, for sample sizes 50, 250, and 1000 (rate parameter = 1)
b1 = Compute_median_cvrg_prob(samplesize = 50, intervalrep = 200, dist = rexp(n = 50, rate = 1), bigdist
b2 = Compute_median_cvrg_prob(samplesize = 250, intervalrep = 200, dist = rexp(n = 250, rate = 1), bigdi
b3 = Compute_median_cvrg_prob(samplesize = 1000, intervalrep = 200, dist = rexp(n = 1000, rate = 1), bi
```

```r
b4 = Compute_median_95CI_width(samplesize = 50, intervalrep = 200, bigdist = rexp(n = 50*200, rate = 1))
b5 = Compute_median_95CI_width(samplesize = 250, intervalrep = 200, bigdist = rexp(n = 250*200, rate = 
b6 = Compute_median_95CI_width(samplesize = 1000, intervalrep = 200, bigdist = rexp(n = 1000*200, rate =

b7 = Compute_robust_cvrg_prob(samplesize = 50, intervalrep = 200, dist = rexp(n = 50, rate = 1), bigdist
b8 = Compute_robust_cvrg_prob(samplesize = 250, intervalrep = 200, dist = rexp(n = 250, rate = 1), bigdi
b9 = Compute_robust_cvrg_prob(samplesize = 1000, intervalrep = 200, dist = rexp(n = 1000, rate = 1), big

b10 = Compute_robust_estimator_CI_width(samplesize = 50, intervalrep = 200, bigdist = rexp(n = 50*200, 
b11 = Compute_robust_estimator_CI_width(samplesize = 250, intervalrep = 200, bigdist = rexp(n = 250*200
b12 = Compute_robust_estimator_CI_width(samplesize = 1000, intervalrep = 200, bigdist = rexp(n = 1000*2
# Table
exptable = createtable(b1,b2,b3,b4,b5,b6,b7,b8,b9,b10,b11,b12, "Exponential Dist", "rate = 1")
exptable
```

```
##                                Convergence Probability
##                                "Exponential Dist"
## Bootstrap, Size = 50           "0.76"
## Bootstrap, Size = 250          "0.905"
## Bootstrap, Size = 1000         "0.565"
## Robust Estimation, Size = 50   "0.835"
## Robust Estimation, Size = 250  "0.17"
## Robust Estimation, Size = 1000 "0.75"
##                                Width of Confidence Interval
##                                "rate = 1"
## Bootstrap, Size = 50           "0.551515351637972"
## Bootstrap, Size = 250          "0.250987104622502"
## Bootstrap, Size = 1000         "0.123590769596971"
## Robust Estimation, Size = 50   "0.393610442104561"
## Robust Estimation, Size = 250  "0.176215055464033"
## Robust Estimation, Size = 1000 "0.089021504395167"
```

```r
# Uniform Distribution, for sample sizes 50, 250, and 1000 (min = 0, max = 1)
c1 = Compute_median_cvrg_prob(samplesize = 50, intervalrep = 200, dist = runif(n = 50, min = 0, max = 1
c2 = Compute_median_cvrg_prob(samplesize = 250, intervalrep = 200, dist = runif(n = 250, min = 0, max =
c3 = Compute_median_cvrg_prob(samplesize = 1000, intervalrep = 200, dist = runif(n = 1000, min = 0, max

c4 = Compute_median_95CI_width(samplesize = 50, intervalrep = 200, bigdist = runif(n = 50*200, min = 0,
c5 = Compute_median_95CI_width(samplesize = 250, intervalrep = 200, bigdist = runif(n = 250*200, min = 
c6 = Compute_median_95CI_width(samplesize = 1000, intervalrep = 200, bigdist = runif(n = 1000*200, min =

c7 = Compute_robust_cvrg_prob(samplesize = 50, intervalrep = 200, dist = runif(n = 50, min = 0, max = 1
c8 = Compute_robust_cvrg_prob(samplesize = 250, intervalrep = 200, dist = runif(n = 250, min = 0, max =
c9 = Compute_robust_cvrg_prob(samplesize = 1000, intervalrep = 200, dist = runif(n = 1000, min = 0, max

c10 = Compute_robust_estimator_CI_width(samplesize = 50, intervalrep = 200, bigdist = runif(n = 50*200,
c11 = Compute_robust_estimator_CI_width(samplesize = 250, intervalrep = 200, bigdist = runif(n = 250*20
c12 = Compute_robust_estimator_CI_width(samplesize = 1000, intervalrep = 200, bigdist = runif(n = 1000*
# Table
uniformtable = createtable(c1,c2,c3,c4,c5,c6,c7,c8,c9,c10,c11,c12, "Uniform Dist", "min = 0, max = 1")
uniformtable
```

```
##                                Convergence Probability
##                                "Uniform Dist"
```

```
## Bootstrap, Size = 50           "0.725"
## Bootstrap, Size = 250          "0.915"
## Bootstrap, Size = 1000         "0.775"
## Robust Estimation, Size = 50   "0.82"
## Robust Estimation, Size = 250  "0.745"
## Robust Estimation, Size = 1000 "0.795"
##                                Width of Confidence Interval
##                                "min = 0, max = 1"
## Bootstrap, Size = 50           "0.268622345614218"
## Bootstrap, Size = 250          "0.12186597735282"
## Bootstrap, Size = 1000         "0.0614908542177582"
## Robust Estimation, Size = 50   "0.197751426757243"
## Robust Estimation, Size = 250  "0.0911275472376818"
## Robust Estimation, Size = 1000 "0.0459218959596955"
```

```
# Gamma Distribution, for sample sizes 50, 250, and 1000 (shape parameter = 3, rate = 1)
d1 = Compute_median_cvrg_prob(samplesize = 50, intervalrep = 200, dist = rgamma(n = 50, shape = 3, rate
d2 = Compute_median_cvrg_prob(samplesize = 250, intervalrep = 200, dist = rgamma(n = 250, shape = 3, ra
d3 = Compute_median_cvrg_prob(samplesize = 1000, intervalrep = 200, dist = rgamma(n = 1000, shape = 3, 

d4 = Compute_median_95CI_width(samplesize = 50, intervalrep = 200, bigdist = rgamma(n = 50*200, shape =
d5 = Compute_median_95CI_width(samplesize = 250, intervalrep = 200, bigdist = rgamma(n = 250*200, shape
d6 = Compute_median_95CI_width(samplesize = 1000, intervalrep = 200, bigdist = rgamma(n = 1000*200, sha

d7 = Compute_robust_cvrg_prob(samplesize = 50, intervalrep = 200, dist = rgamma(n = 50, shape = 3, rate
d8 = Compute_robust_cvrg_prob(samplesize = 250, intervalrep = 200, dist = rgamma(n = 250, shape = 3, ra
d9 = Compute_robust_cvrg_prob(samplesize = 1000, intervalrep = 200, dist = rgamma(n = 1000, shape = 3, 

d10 = Compute_robust_estimator_CI_width(samplesize = 50, intervalrep = 200, bigdist = rgamma(n = 50*200
d11 = Compute_robust_estimator_CI_width(samplesize = 250, intervalrep = 200, bigdist = rgamma(n = 250*2
d12 = Compute_robust_estimator_CI_width(samplesize = 1000, intervalrep = 200, bigdist = rgamma(n = 1000
# Table
gammatable = createtable(d1,d2,d3,d4,d5,d6,d7,d8,d9,d10,d11,d12, "Gamma Dist", "shape = 3, rate = 1")
gammatable
```

```
##                                Convergence Probability
##                                "Gamma Dist"
## Bootstrap, Size = 50           "0.93"
## Bootstrap, Size = 250          "0.935"
## Bootstrap, Size = 1000         "0.86"
## Robust Estimation, Size = 50   "0.775"
## Robust Estimation, Size = 250  "0.385"
## Robust Estimation, Size = 1000 "0.815"
##                                Width of Confidence Interval
##                                "shape = 3, rate = 1"
## Bootstrap, Size = 50           "1.12943965287951"
## Bootstrap, Size = 250          "0.498649060936842"
## Bootstrap, Size = 1000         "0.252522627334669"
## Robust Estimation, Size = 50   "0.858623341344718"
## Robust Estimation, Size = 250  "0.390588073609076"
## Robust Estimation, Size = 1000 "0.195685839087385"
```

The tables are fairly straightforward in showing that the CI widths generally decrease as we increase sample size. In addition, we can compare the coverage probabilities
by their trends as we alter the sample size. Based on the characteristics of the distribution at hand, the

estimator from bootstrap and robust estimation
can vary from little to wildly

```
## Appendix ##
## Packages Utilized ##
library("dplyr", lib.loc="~/R/x86_64-pc-linux-gnu-library/3.4")
library("ggplot2", lib.loc="~/R/x86_64-pc-linux-gnu-library/3.4")
library("knitr", lib.loc="~/R/x86_64-pc-linux-gnu-library/3.4")
library("rmarkdown", lib.loc="~/R/x86_64-pc-linux-gnu-library/3.4")
library("stringdist", lib.loc="~/R/x86_64-pc-linux-gnu-library/3.4")
library("stringr", lib.loc="~/R/x86_64-pc-linux-gnu-library/3.4")
library("tidyr", lib.loc="~/R/x86_64-pc-linux-gnu-library/3.4")

# Begin Functions Section #
Boot_median_95CI_function = function(x, B = 1000){ # Use the bootstrapping technique with 1000 bootstra
  n = length(x)
  boot_samples = sample(x, n*B, replace = TRUE)  # Sample with replacement
  dim(boot_samples) = c(B, n) # each row is a dataset
  boot_norm_median = apply(boot_samples, 1, function(b) median(b)) # Apply median to each bootstrap sam
  norm_lower_bound = unname(quantile(boot_norm_median, .025))
  norm_upper_bound = unname(quantile(boot_norm_median, .975)) # Find the 95% range of values and bounds
  return(c(norm_lower_bound, norm_upper_bound))
}

Compute_median_cvrg_prob = function(samplesize, intervalrep, dist, bigdist){ # Calculate convergence pr
  norm_median_est = median(dist) # theoretical value
  dim(bigdist) = c(intervalrep, samplesize)
  AllCI = apply(bigdist, 1, Boot_median_95CI_function) # Bootstrap the repetitions of samples to obtain
  cvrg_prob = mean(((AllCI[1,]) < norm_median_est) & ((AllCI[2,]) > norm_median_est)) # Calculate the p
  return(cvrg_prob)
}

Compute_median_95CI_width = function(samplesize, intervalrep, bigdist){ # Calculating the average width
  dim(bigdist) = c(intervalrep, samplesize)
  AllCI = apply(bigdist, 1, Boot_median_95CI_function) # Bootstrapping again
  width = mean(AllCI[2,] - AllCI[1,])
  return(width)
}

Compute_robust_cvrg_prob = function(samplesize, intervalrep, dist, bigdist){ # Using a robust method, w
  norm_median_est = median(dist) # Theoretical value
  dim(bigdist) = c(intervalrep, samplesize)
  # Calculation of estimated parameters
  med = apply(bigdist, 1, median)
  absol = abs(bigdist - med)
  mad = apply(absol, 1, median)
  mul = qnorm(1-{1-.95}/2) # 95% CI Level
  error = (mul * (1.49 * mad)) / (sqrt(dim(bigdist)[2]))
  ubd = med + error
  lbd = med - error
  cvrg_prob = mean(((lbd) < norm_median_est) & ((ubd) > norm_median_est)) # Calculate proportion of CIs
  return(cvrg_prob)
}
```

```r
Compute_robust_estimator_CI_width = function(samplesize, intervalrep, bigdist){ # Calculating average w
  dim(bigdist) = c(intervalrep, samplesize)
  # Calculation of estimated parameters
  med = apply(bigdist, 1, median)
  absol = abs(bigdist - med)
  mad = apply(absol, 1, median)
  mul = qnorm(1-({1-.95}/2)) # 95% CI Level
  error = (mul * (1.49 * mad)) / (sqrt(dim(bigdist)[2]))
  ubd = med + error
  lbd = med - error
  return(mean(ubd - lbd))
}

createtable = function(a,b,c,d,e,f,g,h,i,j,k,l,m,n){ # Create a table for viewing and reporting purpose
  values = c(m,a,b,c,g,h,i,n,d,e,f,j,k,l)
  listdimnames = list(c("", "Bootstrap, Size = 50", "Bootstrap, Size = 250", "Bootstrap, Size = 1000",
  table = matrix(data = values, nrow = 7, ncol = 2, dimnames = listdimnames)
  return(table)
}
# End Functions Section #
```