# Mini Project4

## Section 1:

first, compute the true median of the gamma distribution with shape = 3, rate = 5. Use the boot package in r to compute the CI for normal, basic and percentile bootstrap with R= 2000. Then, do this 1000 times, if the true median is in the CI, set the corresponding value of a record vector to 1, otherwise 0. Then we get a vector of 1000 values, which is either 1 or 0. And the portion of 1’s is the coverage probility. And do it for n = 5, 10, 30, 100.

## Section 2:

The result can be seen as follows:

|  |
| --- |
| > normal.coverage(1000, median, 5, 2000)  [1] 0.911  There were 50 or more warnings (use warnings() to see the first 50)  > normal.coverage(1000, median, 10, 2000)  [1] 0.896  There were 50 or more warnings (use warnings() to see the first 50)  > normal.coverage(1000, median, 30, 2000)  [1] 0.917  There were 50 or more warnings (use warnings() to see the first 50)  > normal.coverage(1000, median, 100, 2000)  [1] 0.911  There were 50 or more warnings (use warnings() to see the first 50)  >  > basic.coverage(1000, median, 5, 2000)  [1] 0.765  There were 50 or more warnings (use warnings() to see the first 50)  > basic.coverage(1000, median, 10, 2000)  [1] 0.829  There were 50 or more warnings (use warnings() to see the first 50)  > basic.coverage(1000, median, 30, 2000)  [1] 0.822  There were 50 or more warnings (use warnings() to see the first 50)  > basic.coverage(1000, median, 100, 2000)  [1] 0.888  There were 50 or more warnings (use warnings() to see the first 50)  >  > percent.coverage(1000, median, 5, 2000)  [1] 0.946  There were 50 or more warnings (use warnings() to see the first 50)  > percent.coverage(1000, median, 10, 2000)  [1] 0.945  There were 50 or more warnings (use warnings() to see the first 50)  > percent.coverage(1000, median, 30, 2000)  [1] 0.938  There were 50 or more warnings (use warnings() to see the first 50)  > percent.coverage(1000, median, 100, 2000)  [1] 0.951  There were 50 or more warnings (use warnings() to see the first 50) |

Generally speaking, as the number of data increase( in this simulation, n = 5, 10, 30, 100), the coverage does not change a lot. However, basic bootstrap CI coverage increases a lot ( from 0.765, 0.829, 0.822, 0.888). And from the result, we can see that in general, percentile performs best, follows by normal. The basic one performs worst

## Section 3:

library(boot)

median = qgamma(0.5,shape = 3, rate = 5) **# get true median**

ci.boot = function(dataNum, r){

data = rgamma(dataNum, shape = 3, rate = 5) **# get data from the distribution with shape =3, rate =5**

median.npar <- function(x, indices){

result <- median(x[indices])

return(result)

}

median.npar.boot <- boot(data, median.npar, R = r, sim = "ordinary",stype="i") **# get bootstrap**

result = boot.ci(median.npar.boot) **# get ci from bootstrap**

return(result)

}

normal.coverage = function( n, median, dataNum, r){ **#get the coverage probability**

record = rep( 0, n ) **# use a vector to store result of each experiments**

for ( i in 1:n ){ # for each experiment

ci = ci.boot(dataNum, r)$normal[1,2:3]

if( median >= ci[1] && median <= ci[2]){ **# check if it covers the true value**

record[ i ] = 1 # if covers, set the value in record to 1

}

}

result = sum( record ) / n **# get the portion of experiments that covers real p**

return ( result )

}

basic.coverage = function( n, median, dataNum, r){ **#get the coverage probability**

record = rep( 0, n ) **# use a vector to store result of each experiments**

for ( i in 1:n ){ # for each experiment

ci = ci.boot(dataNum, r)$basic[1,4:5]

if( median >= ci[1] && median <= ci[2]){ **# check if it covers the true value**

record[ i ] = 1 **# if covers, set the value in record to 1**

}

}

result = sum( record ) / n **# get the portion of experiments that covers real p**

return ( result )

}

percent.coverage = function( n, median, dataNum, r){ **#get the coverage probability**

record = rep( 0, n ) **# use a vector to store result of each experiments**

for ( i in 1:n ){ **# for each experiment**

ci = ci.boot(dataNum, r)$percent[1,4:5]

if( median >= ci[1] && median <= ci[2]){ **# check if it covers the true value**

record[ i ] = 1 **# if covers, set the value in record to 1**

}

}

result = sum( record ) / n **# get the portion of experiments that covers real p**

return ( result )

}

normal.coverage(1000, median, 5, 2000)

normal.coverage(1000, median, 10, 2000)

normal.coverage(1000, median, 30, 2000)

normal.coverage(1000, median, 100, 2000)

basic.coverage(1000, median, 5, 2000)

basic.coverage(1000, median, 10, 2000)

basic.coverage(1000, median, 30, 2000)

basic.coverage(1000, median, 100, 2000)

percent.coverage(1000, median, 5, 2000)

percent.coverage(1000, median, 10, 2000)

percent.coverage(1000, median, 30, 2000)

percent.coverage(1000, median, 100, 2000)