```
my.bootstrap.ci<-function(vec0, nboot, alpha) {
  n0<-length(vec0) # extract sample size
  mean0<-mean(vec0) # find the mean of the data set
  sd0<-sqrt(var(vec0)) # find the standard deviation of the data set

  bootvec<-NULL # create an empty vector to store the bootstrap

  for(i in 1:nboot) { # boostrap distribution using a for loop
    vecb<-sample(vec0,replace=T) # sampling with replacement
    meanb<-mean(vecb) # find the mean
    sdb<-sqrt(var(vecb)) # find the standard deviation
    bootvec<-c(bootvec,(meanb-mean0)/(sdb/sqrt(n0))) # overwrite the empty vector
  }

  lq<-quantile(bootvec,alpha/2) # lower quartile of the bootstrap distribution
  uq<-quantile(bootvec,1-alpha/2) # upper quartile of the bootstrap distribution

  LB<-mean0-(sd0/sqrt(n0))*uq # Find the lower bound
  UB<-mean0-(sd0/sqrt(n0))*lq # Find upper bound

  NLB<-mean0-(sd0/sqrt(n0))*qnorm(1-alpha/2) # New lower bound to normalize data
  NUB<-mean0+(sd0/sqrt(n0))*qnorm(1-alpha/2) # New upper bound to normalize data
  list(bootstrap.confidence.interval=c(LB,UB),normal.confidence.interval=c(NLB,NUB))
} # Calculating normal confidence interval

my.simulation<- function(mu.val, n, nsim) {
  cvec.boot<-NULL # Coverage indicator vector for bootstrap
  cvec.norm<-NULL # Coverage indicator vector for normal

  mulnorm<-(exp(mu.val+1/2)) #Real mean

  for(i in 1:nsim) { # run simulation in a for loop
    if((i/10)==floor(i/10)) {
      print(i)
    }
    vec.sample<-rlnorm(n,mu.val) # Sample the simulation vector
    # run bootstraps
    boot.list<-my.bootstrap.ci(vec.sample, 10000, 0.1)
    boot.conf<-boot.list$bootstrap.confidence.interval
    norm.conf<-boot.list$normal.confidence.interval

    cvec.boot<-c(cvec.boot,(boot.conf[1]<mulnorm)*(boot.conf[2]>mulnorm)) #Count up
coverage by bootstrap interval
```

```
    cvec.norm<-c(cvec.norm,(norm.conf[1]<mulnorm)*(norm.conf[2]>mulnorm)) # Coverage
by normal theory interval
  }

  list(boot.coverage=(sum(cvec.boot)/nsim),norm.coverage=(sum(cvec.norm)/nsim))
} #Output coverage probability
```

	3	10	30	100
0.1	boot: 0.838 norm: 0.656	boot: 0.854 norm: 0.752	boot: 0.88, norm: 0.817	boot: 0.875 norm: 0.857
0.05	boot: 1 norm: 0.673	boot: 0.899 norm: 0.777	boot: 0.928 norm: 0.867	boot: 0.94 norm: 0.908

alpha = 0.1

```
> my.simulation(3, 3, 1000)
$boot.coverage
[1] 0.838

$norm.coverage
[1] 0.656

> my.simulation(3, 10, 1000)
$boot.coverage
[1] 0.854

$norm.coverage
[1] 0.752

> my.simulation(3, 30, 1000)
$boot.coverage
[1] 0.88

$norm.coverage
[1] 0.817

> my.simulation(3, 100, 1000)
$boot.coverage
[1] 0.875

$norm.coverage
[1] 0.857
```

alpha = 0.05

```
> my.simulation(3, 3, 1000)
$boot.coverage
[1] 1

$norm.coverage
[1] 0.673

> my.simulation(3, 10, 1000)
$boot.coverage
[1] 0.899

$norm.coverage
[1] 0.777

> my.simulation(3, 30, 1000)
$boot.coverage
[1] 0.928

$norm.coverage
[1] 0.867

> my.simulation(3, 100, 1000)
$boot.coverage
[1] 0.94

$norm.coverage
[1] 0.908
```

Conclusion: For both alpha of 0.1 and 0.05, as we increase the number of simulation numbers (n), the difference between the outputted values from boot and norm decreases, as they become more similar.