Estimating Population Size

In any war, it is always of value to one side to have good intelligence on the weapons resources of the other side. During the Second World War, for example, Allied military planners eagerly searched fro ways to accurately estimate the Axis production of tanks, aircrafts and numerous other weapons platforms. In the specific case of German tanks, a very clever way todo that was based on using either the stamped serial numbers or gearbox markings on captured Mark I and MArk V tanks, respectively. This type of problem has far wider applications.

We can experimentally see how well the formula that was used works in practice with a Monte Carlo Simulation. That is, the program first randomly picks a value for N (integer between 100 and 1000) with a value for the the sample size as a percentage of N.

The program then generates, randomly, n different integers in the interval 1 to N, the maximum of those integers is then used in the estimation formula to estimate a value for N. This estimate can be compared to the actual value of N to determine how weel the formual has performed. We investigate samples that are 2%, 5%, 10% and 20% of the population.

Function to implement Monte Carlo simulation using formula

```
runSim <- function(nSim=1000,  sample.size.perc=10/100 ) {

  #set up an array to store parameter estimates
  estArray <- array(0, dim=c(nSim,3))

  for (i in 1:nSim) {

    N <- sample(100:1000,1)  # select a population size

    n <- round(sample.size.perc*N,0) # the sample size

    sample2 <- sample(N, n) # select a sample from the population

    est <- (n+1)/n*max(sample2) - 1 # apply formula

    est <-  round(est ,0) # estimate of pop size

    estArray[i,1] <- (est - N)/ N *100  # percentage error is captured
    estArray[i,2] <- N
    estArray[i,3] <- sample.size.perc

  }

  list(estArray=estArray )

}
```

Sample size 2% of population

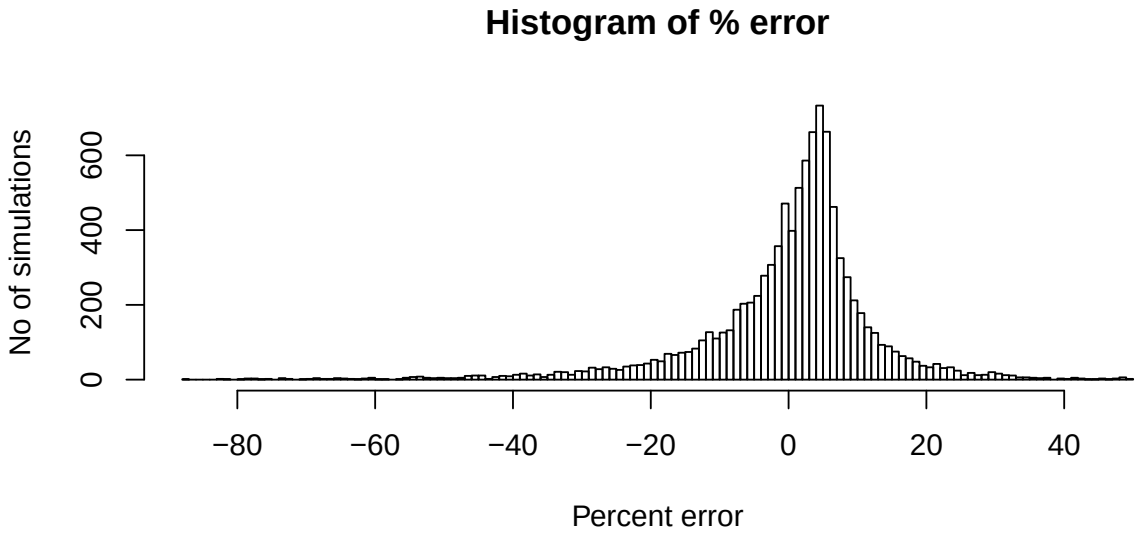
```
  x<- runSim(nSim=10000,  sample.size.perc=2/100) # run simulation
  apply(x$estArray, 2, mean, na.rm=TRUE)[1]       # mean % error
```

```
[1] 0.01385591
```

```
  quantile( x$estArray[,1] , prob=c(0.025, 0.5, 0.975))  # median and 95% confidence intervals for % er
```

```
      2.5%        50%      97.5% 
-33.333333   2.290867  22.222222 
```

```
  hist(x$estArray[,1], breaks=100, main="Histogram of % error" , xlab="Percent error", ylab="No of simul
```



Sample size 5% of population

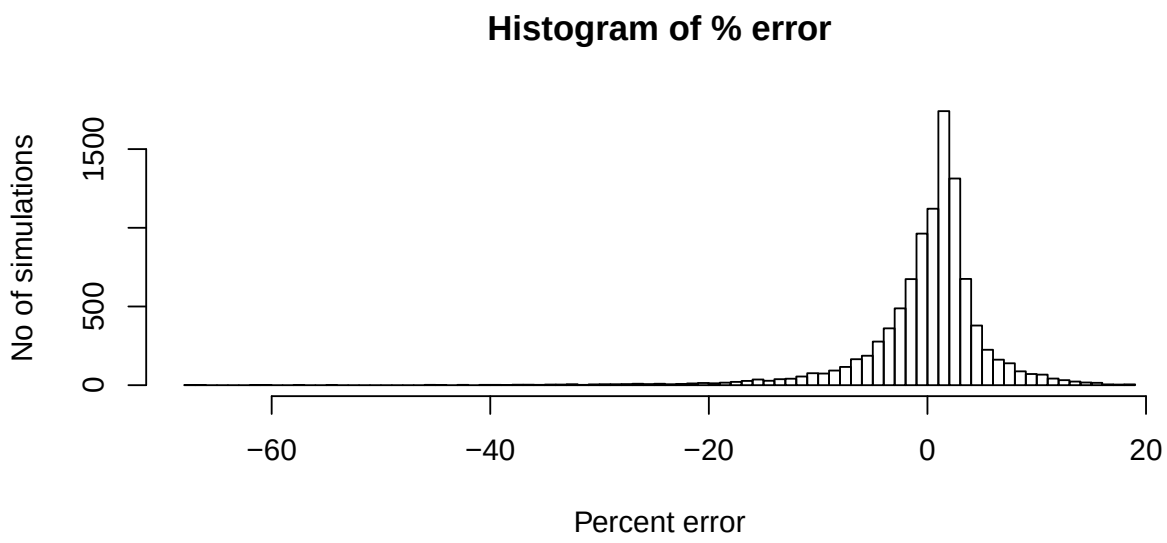
```
  x<- runSim(nSim=10000,  sample.size.perc=5/100) # run simulation
  apply(x$estArray, 2, mean, na.rm=TRUE)[1]       # mean % error
```

```
[1] -0.0187242
```

```
  quantile( x$estArray[,1] , prob=c(0.025, 0.5, 0.975))  # median and 95% confidence intervals for % er
```

```
      2.5%        50%      97.5% 
-14.525259   1.003344   9.524408 
```

```
  hist(x$estArray[,1], breaks=100, main="Histogram of % error" , xlab="Percent error", ylab="No of simul
```



Sample size 10% of population

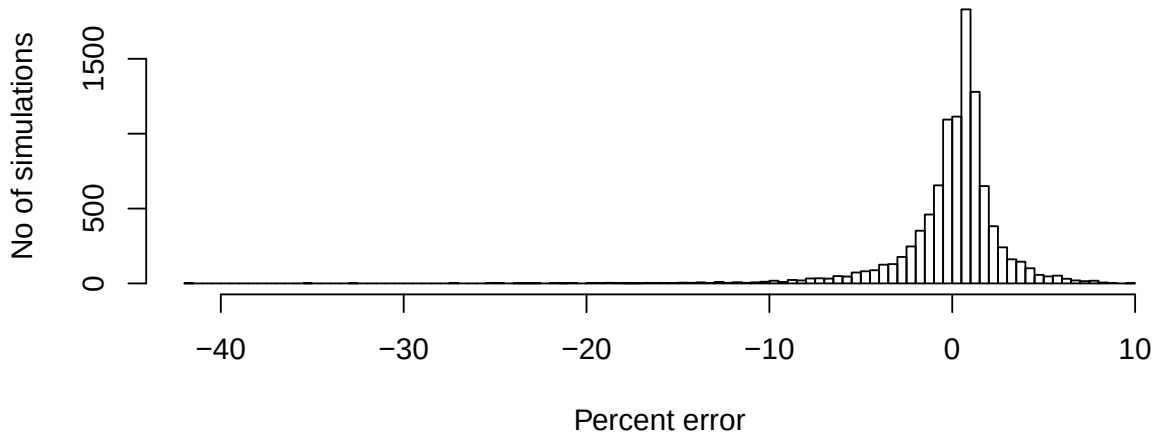
```
  x<- runSim(nSim=10000,  sample.size.perc=10/100) # run simulation
  apply(x$estArray, 2, mean, na.rm=TRUE)[1]       # mean % error
```

```
[1] 0.04915648
```

```
  quantile( x$estArray[,1] , prob=c(0.025, 0.5, 0.975))  # median and 95% confidence intervals for % er
```

```
     2.5%        50%      97.5%
-6.8185347  0.5107252  4.5005653
```

```
    hist(x$estArray[,1], breaks=100, main="Histogram of % error" , xlab="Percent error", ylab="No of si
```



Sample size 20% of population

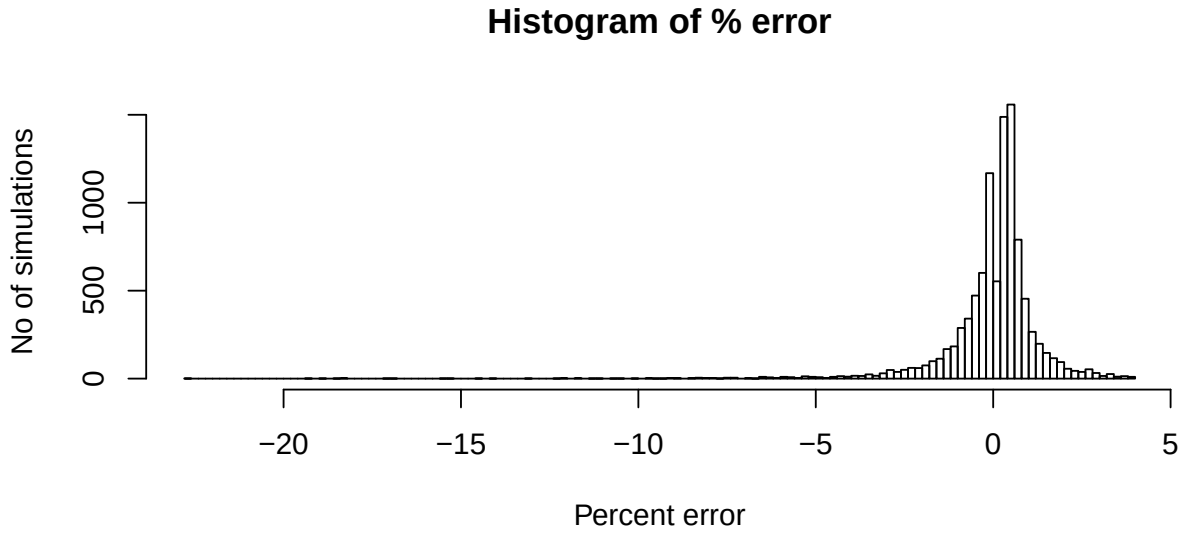
```
  x<- runSim(nSim=10000,  sample.size.perc=20/100) # run simulation
  apply(x$estArray, 2, mean, na.rm=TRUE)[1]       # mean % error
```

```
[1] 0.01389191
```

```
  quantile( x$estArray[,1] , prob=c(0.025, 0.5, 0.975))  # median and 95% confidence intervals for % er
```

```
     2.5%        50%      97.5%
-3.1392783  0.2453988  2.1739130
```

```
  hist(x$estArray[,1], breaks=100, main="Histogram of % error" , xlab="Percent error", ylab="No of simul
```



Computing Environment

```
sessionInfo()
```

```
R version 3.6.0 (2019-04-26)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 17134)

Matrix products: default

locale:
[1] LC_COLLATE=English_United Kingdom.1252
[2] LC_CTYPE=English_United Kingdom.1252
[3] LC_MONETARY=English_United Kingdom.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United Kingdom.1252

attached base packages:
[1] stats     graphics  grDevices utils     datasets
[6] methods   base
```

```
other attached packages:
[1] knitr_1.23

loaded via a namespace (and not attached):
 [1] compiler_3.6.0  magrittr_1.5    tools_3.6.0
 [4] htmltools_0.3.6 yaml_2.2.0      Rcpp_1.0.1
 [7] stringi_1.4.3   rmarkdown_1.13  stringr_1.4.0
[10] xfun_0.7        digest_0.6.19   evaluate_0.14
```

This took 0.87 seconds to execute.