Monte Carlo Error

Sriram Kannan

09-13-2021

Contents

Introduction **1**

Background **1**

Methods and Observations **1**

Introduction

Simulation by essence, operates through approximation, that is, there is some degree of error in a quantity estimated by the Monte Carlo simulation. Intuitively, it would seem that the degree of error is inversely proportional to the number of times a simulation is repeated. In this blog post, we attempt to investigate the relationship between then number of replicates and simulation error - Both Absolute and Relative Error.

Background

Monte Carlo simulation is a technique used to predict the probability of different outcomes when factors causing uncertainty are present. It is used across a wide range of fields such as finance, manufacturing, research and development, insurance, logistics etc in order to bolster the decision making process. It equips the user with a range of possible outcomes and probabilities of any choice of action.

This technique was first used by scientists working on the Manhattan Project during World War 2 and gains its moniker from Monte Carlo, the Monaco resort town famous for its casinos.

The Monte Carlo simulation is then run across an uncertain variable which is assigned a random value and outputs a result. This process is repeated again and again while assigning the variable in question with many different values. Once the simulation is complete, the results are averaged together to provide an estimate. It focuses on constantly repeating random samples to achieve certain results.

Methods and Observations

In order to illustrate the relationship between the number of replicates and the simulation error, we perform a 14 X 5 factorial experiment simulation that estimates the error for each combination of replicate number (4, 8, 16, 32 and so on) and probabilities (0.01, 0.05, 0.10, 0.25, 0.50).

P denotes the true underlying probability pcap denotes the probability estimated from the simulation

absolute error = |pcap - p|

and

relative error = |pcap - p|/

Our number of trials in an experiment (r) is fixed at 10000.

```
#Required library for the plots
library(tgsify)
```

```
## Loading required package: data.table
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
## Loading required package: dtplyr
```

```
## Loading required package: showtext
```

```
## Loading required package: sysfonts
```

```
## Loading required package: showtextdb
```

```
#We initialize a grid with all possible combinations of number of replicates (N which is a sequence of
op <- expand.grid(
  N = 2^c(2:15),
  P = c(0.01, 0.05, 0.10, 0.25, 0.50),
  absolute_error = NA,
  relative_error = NA,
  KEEP.OUT.ATTRS = FALSE
)
```

```
#We calculate the pcap value as well as the errors for the grid above using the below chunk of code.
r <- 10000
for(i in 1:nrow(op))
{
  p <- op$P[i]
```

```
  n <- op$N[i]
  pcap <- rbinom(r,n,p)/n
  op[i, "absolute_error"] <- mean(abs(pcap - p))
  op[i, "relative_error"] <- mean(abs(pcap - p)/p)
}
```

```
op
```

```
##        N    P absolute_error relative_error
## 1      4 0.01   0.0198410000    1.984100000
## 2      8 0.01   0.0186555000    1.865550000
## 3     16 0.01   0.0168835000    1.688350000
## 4     32 0.01   0.0143373750    1.433737500
## 5     64 0.01   0.0105142500    1.051425000
## 6    128 0.01   0.0070581875    0.705818750
## 7    256 0.01   0.0050055937    0.500559375
## 8    512 0.01   0.0034945937    0.349459375
## 9   1024 0.01   0.0024496055    0.244960547
## 10  2048 0.01   0.0017634727    0.176347266
## 11  4096 0.01   0.0012307275    0.123072754
## 12  8192 0.01   0.0008821611    0.088216113
## 13 16384 0.01   0.0006196033    0.061960327
## 14 32768 0.01   0.0004359791    0.043597913
## 15     4 0.05   0.0810000000    1.620000000
## 16     8 0.05   0.0662850000    1.325700000
## 17    16 0.05   0.0441600000    0.883200000
## 18    32 0.05   0.0316156250    0.632312500
## 19    64 0.05   0.0216290625    0.432581250
## 20   128 0.05   0.0152564063    0.305128125
## 21   256 0.05   0.0109209375    0.218418750
## 22   512 0.05   0.0077505859    0.155011719
## 23  1024 0.05   0.0054632227    0.109264453
## 24  2048 0.05   0.0038137109    0.076274219
## 25  4096 0.05   0.0027417578    0.054835156
## 26  8192 0.05   0.0019600244    0.039200488
## 27 16384 0.05   0.0013445154    0.026890308
## 28 32768 0.05   0.0009566272    0.019132544
## 29     4 0.10   0.1299350000    1.299350000
## 30     8 0.10   0.0861425000    0.861425000
## 31    16 0.10   0.0613912500    0.613912500
## 32    32 0.10   0.0424200000    0.424200000
## 33    64 0.10   0.0304443750    0.304443750
## 34   128 0.10   0.0210918750    0.210918750
## 35   256 0.10   0.0151452344    0.151452344
## 36   512 0.10   0.0107382422    0.107382422
## 37  1024 0.10   0.0074864258    0.074864258
## 38  2048 0.10   0.0053256543    0.053256543
## 39  4096 0.10   0.0037622217    0.037622217
## 40  8192 0.10   0.0026392871    0.026392871
## 41 16384 0.10   0.0018724646    0.018724646
## 42 32768 0.10   0.0012938605    0.012938605
## 43     4 0.25   0.1609250000    0.643700000
## 44     8 0.25   0.1167750000    0.467100000
```

```
## 45    16 0.25   0.0842187500    0.336875000
## 46    32 0.25   0.0603343750    0.241337500
## 47    64 0.25   0.0424875000    0.169950000
## 48   128 0.25   0.0305523437    0.122209375
## 49   256 0.25   0.0215292969    0.086117187
## 50   512 0.25   0.0154519531    0.061807813
## 51  1024 0.25   0.0107437500    0.042975000
## 52  2048 0.25   0.0076276855    0.030510742
## 53  4096 0.25   0.0054017578    0.021607031
## 54  8192 0.25   0.0038177124    0.015270850
## 55 16384 0.25   0.0026990906    0.010796362
## 56 32768 0.25   0.0018885468    0.007554187
## 57     4 0.50   0.1870250000    0.374050000
## 58     8 0.50   0.1368375000    0.273675000
## 59    16 0.50   0.0971062500    0.194212500
## 60    32 0.50   0.0707218750    0.141443750
## 61    64 0.50   0.0505218750    0.101043750
## 62   128 0.50   0.0349468750    0.069893750
## 63   256 0.50   0.0251566406    0.050313281
## 64   512 0.50   0.0177716797    0.035543359
## 65  1024 0.50   0.0126091797    0.025218359
## 66  2048 0.50   0.0088860840    0.017772168
## 67  4096 0.50   0.0061282227    0.012256445
## 68  8192 0.50   0.0044132568    0.008826514
## 69 16384 0.50   0.0031083618    0.006216724
## 70 32768 0.50   0.0022028076    0.004405615
```

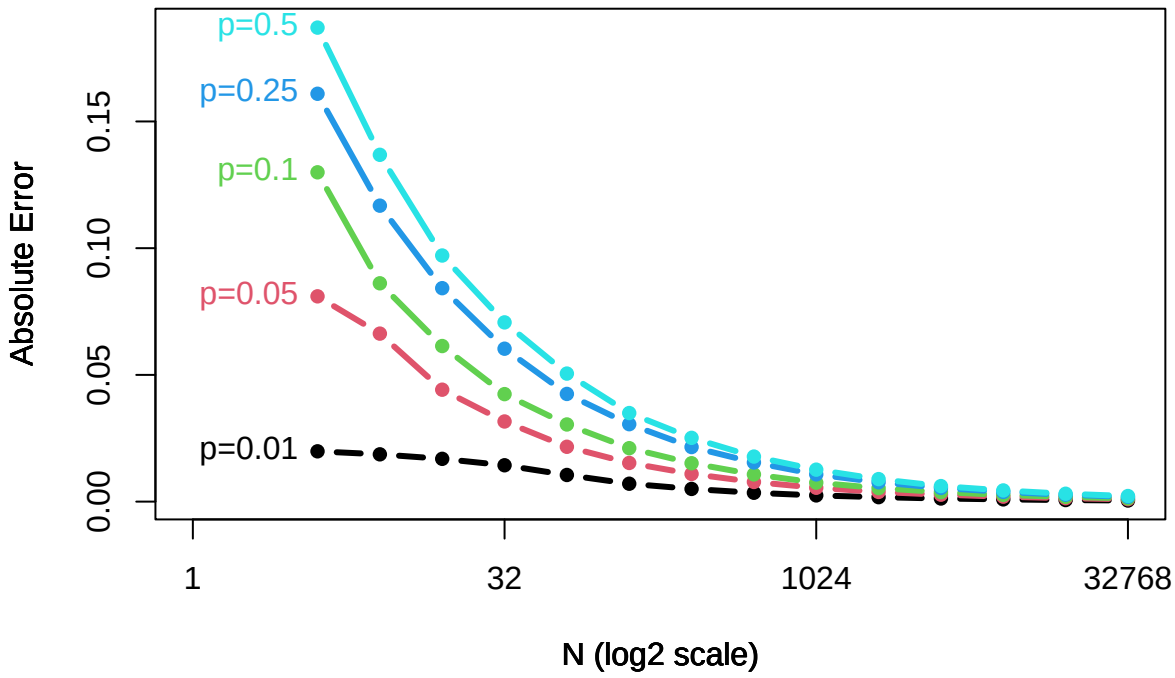
The grid above shows the both error values for all combinations of replicates and probabilities in our experiment.

```
op %>%
  mutate(x = log2(N)) %>%
  mutate(col = as.factor(P) %>% as.numeric) %>%
  #dev.new(width=5, height=4, unit="in") %>%
  plot_setup(absolute_error ~ x, c(0,15)) %>%
  split(.$P) %>%
  lwith({
    lines(x, absolute_error, col = col[1], lwd = 3, type = "b", pch = 16)
    text(x[1], absolute_error[1], "p="%|%P[1], pos = 2, col = col[1])
    title(xlab = "N (log2 scale)", ylab = "Absolute Error")
  })
```

```
## $`0.01`
## NULL
##
## $`0.05`
## NULL
##
## $`0.1`
## NULL
##
## $`0.25`
## NULL
```

```
##
## $`0.5`
## NULL
```

```
axis(2)
axis(1, at = axTicks(1), labels = 2^axTicks(1))
box()
```



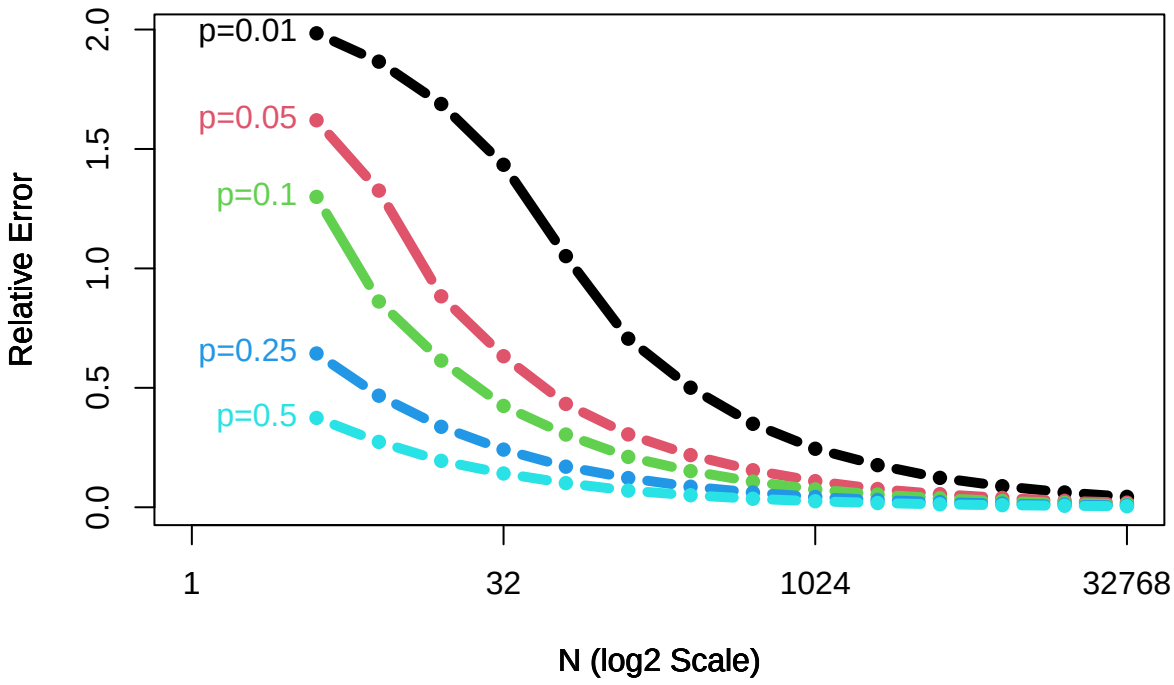
The plot above shows the relationship between absolute errors and the number of replicates. We can see that as the number of replicates increases, the absolute error approaches 0 irrespective of the probability. A second observation is that true probability value is directly proportional to the absolute error - this gets neutered over time as replicates increase.

```
op %>%
  mutate(x = log2(N)) %>%
  mutate(col = as.factor(P) %>% as.numeric) %>%
  plot_setup(relative_error ~ x, c(0,15)) %>%
  split(.$P) %>%
  lwith({
    lines(x, relative_error, col = col[1], lwd = 5, type = "b", pch = 16)
    text(x[1], relative_error[1], "p="%|%P[1], pos = 2, col = col[1])
    title(xlab = "N (log2 Scale)", ylab = "Relative Error")
  })
```

```
## $`0.01`
## NULL
```

```
##
## $`0.05`
## NULL
##
## $`0.1`
## NULL
##
## $`0.25`
## NULL
##
## $`0.5`
## NULL
```

```
axis(2)
axis(1, at = axTicks(1), labels = 2^axTicks(1))
box()
```



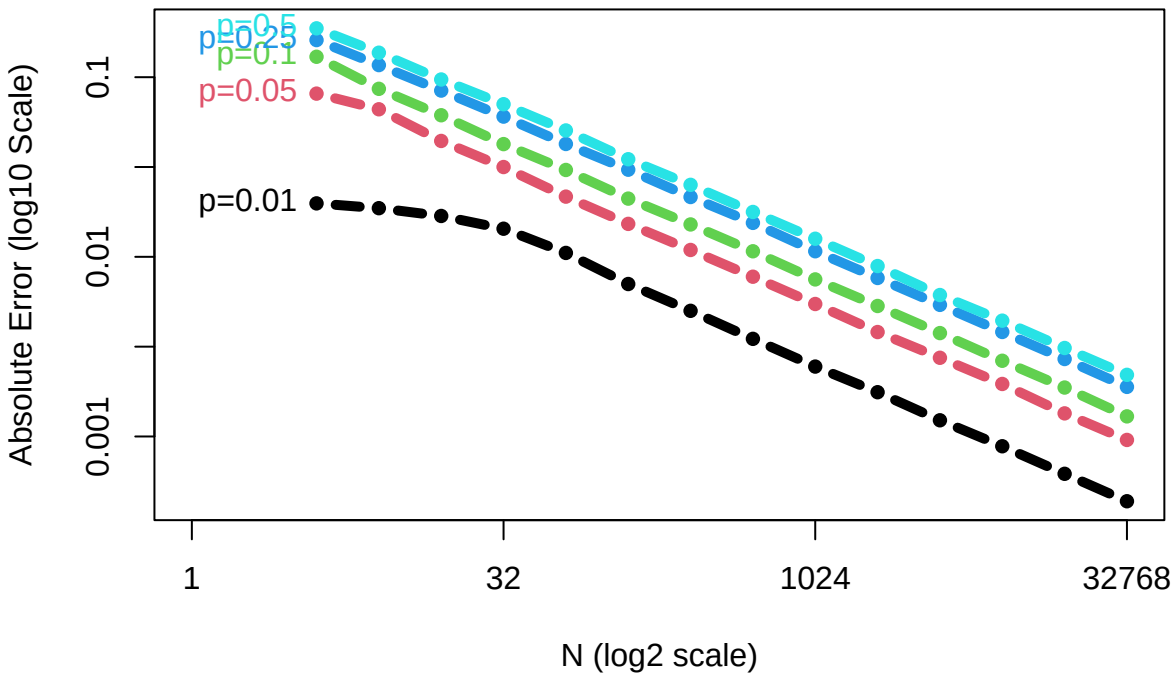
The plot above shows the relationship between relative errors and the number of replicates. We can see that as the number of replicates increases, the relative error also approaches 0 irrespective of the probability. A second observation is that true probability value is inversely proportional to the relative error unlike the absolute error - this also however, gets neutered over time as replicates increase.

```
op %>%
  mutate(x = log2(N)) %>%
  mutate(absolute_error = log10(absolute_error)) %>%
  mutate(col = as.factor(P) %>% as.numeric) %>%
```

```
  plot_setup(absolute_error ~ x, c(0,15)) %>%
  split(.$P) %>%
  lwith({
    lines(x, absolute_error, col = col[1], lwd = 5, type = "b", pch = 16)
    text(x[1], absolute_error[1], "p="%|%P[1], pos = 2, col = col[1])
  })
```

```
## $`0.01`
## NULL
##
## $`0.05`
## NULL
##
## $`0.1`
## NULL
##
## $`0.25`
## NULL
##
## $`0.5`
## NULL
```

```
axis(2, at = axTicks(2), labels = 10^axTicks(2))
axis(1, at = axTicks(1), labels = 2^axTicks(1))
title(xlab = "N (log2 scale)", ylab = "Absolute Error (log10 Scale)")
box()
```



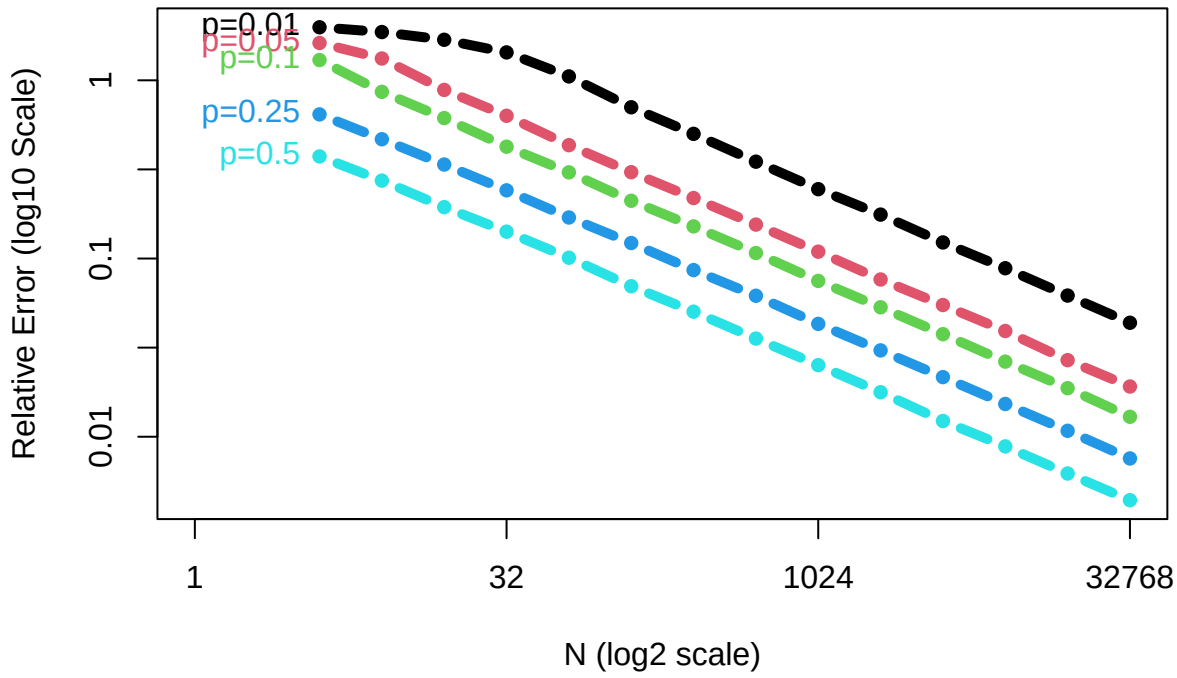
The plot above shows the relationship between absolute errors and the number of replicates with the absolute error being in the log10 scale. We see a more precise view of the magnitude of the absolute error and notice a gradual decay in errors over time as replicates increase irrespective of probability. True probability value is directly proportional to the absolute error as expected.

```
op %>%
  mutate(x = log2(N)) %>%
  mutate(relative_error = log10(relative_error)) %>%
  mutate(col = as.factor(P) %>% as.numeric) %>%
  plot_setup(relative_error ~ x, c(0,15)) %>%
  split(.$P) %>%
  lwith({
    lines(x, relative_error, col = col[1], lwd = 5, type = "b", pch = 16)
    text(x[1], relative_error[1], "p="%|%P[1], pos = 2, col = col[1])
  })
```

```
## $`0.01`
## NULL
##
## $`0.05`
## NULL
##
## $`0.1`
## NULL
##
## $`0.25`
## NULL
```

```
##
## $`0.5`
## NULL
```

```
axis(2, at = axTicks(2), labels = 10^axTicks(2))
axis(1, at = axTicks(1), labels = 2^axTicks(1))
title(xlab = "N (log2 scale)", ylab = "Relative Error (log10 Scale)")
box()
```



The plot above shows the relationship between relative errors and the number of replicates with the relative error being in the log10 scale. We see a more precise view of the magnitude of the relative error and notice a gradual decay in errors over time as replicates increase irrespective of probability. True probability value is inversely proportional to the relative error as expected.