DATA609 HW4

Bin Lin

2017-9-24

Page 191: #3. Using Monte Carlo simulation, write an algorithm to calculate an approximation to ?? by considering the number of random points selected inside the quarter circle

$Q: x^2 + y^2 = 1, \quad x \geq 0, \quad y \geq 0$

where the quarter circle is taken to be inside the square

$S: 0 \leq x \leq 1 \quad and \quad 0 \leq y \leq 1$

Use the equation ?? / 4 = area Q/area S

```
set.seed(888)
sim_pi <- function(n)
{
    x <- runif(n)
    y <- runif(n)
    count = 0
    for (i in 1:n)
      {
      if (x[i] ^ 2 + y[i] ^ 2 <=1)
        count = count + 1
      }
    return (count / n * 4)
}

points <- c(10, 100, 1000, 10000, 100000)
results <- lapply(points, sim_pi)

results
```

```
## [[1]]
## [1] 3.2
##
## [[2]]
## [1] 3.12
##
## [[3]]
## [1] 3.176
##
## [[4]]
## [1] 3.1164
##
## [[5]]
## [1] 3.1476
```

Page 194: #1. Use the middle-square method to generate

```
middle_square <- function (x, n)
{
  if (x == 0)
    return (0)

  seed <- x
  a <- seed ^ 2
  s <- toString(a)

  len = floor(log10(a)) + 1

  while (len < 2 * n)
    {
    s <- paste("0", s, sep = "")
    len = len + 1
    }
  start <- (len / 2 - n / 2 + 1)
  end <- (len / 2 + n / 2)
  s <- substr(s, start, end)

  return (as.numeric(s))

}
```

a. 10 random numbers using x0 = 1009.

```
results <- c()
rand_num <- middle_square(1009, 4)
results <- rbind(results, rand_num)


iteration <- 1
while (iteration < 10)
  {
  rand_num <- middle_square(rand_num, 4)
  results <- rbind(results, rand_num)
  iteration <- iteration + 1
  }

print (results)
```

```
##          [,1]
## rand_num  180
## rand_num  324
## rand_num 1049
## rand_num 1004
## rand_num   80
## rand_num   64
## rand_num   40
## rand_num   16
## rand_num    2
## rand_num    0
```

b. 20 random numbers using x0 = 653217.

```
results <- c()
rand_num <- middle_square(653217, 6)
results <- rbind(results, rand_num)


iteration <- 1
while (iteration < 20)
  {
  rand_num <- middle_square(rand_num, 6)
  results <- rbind(results, rand_num)
  iteration <- iteration + 1
  }

print (results)
```

```
##             [,1]
## rand_num 692449
## rand_num 485617
## rand_num 823870
## rand_num 761776
## rand_num 302674
## rand_num 611550
## rand_num 993402
## rand_num 847533
## rand_num 312186
## rand_num 460098
## rand_num 690169
## rand_num 333248
## rand_num  54229
## rand_num 940784
## rand_num  74534
## rand_num 555317
## rand_num 376970
## rand_num 106380
## rand_num 316704
## rand_num 301423
```

c. 15 random numbers using x0 = 3043.

```
results <- c()
rand_num <- middle_square(3043, 4)
results <- rbind(results, rand_num)


iteration <- 1
while (iteration < 15)
  {
  rand_num <- middle_square(rand_num, 4)
  results <- rbind(results, rand_num)
  iteration <- iteration + 1
  }

print (results)
```

```
##          [,1]
## rand_num 2598
## rand_num 7496
## rand_num 1900
## rand_num 6100
## rand_num 2100
## rand_num 4100
## rand_num 8100
## rand_num 6100
## rand_num 2100
## rand_num 4100
## rand_num 8100
## rand_num 6100
## rand_num 2100
## rand_num 4100
## rand_num 8100
```

d. Comment about the results of each sequence. Was there cycling? Did each sequence degenerate rapidly?

For A, there is no cycling, however, the sequence degenerate rapidly.

For B, no cycling, no degeneration.

For C, it does not degenerate, but it cycles around every 5 values.

4. Horse Race-Construct and perform a Monte Carlo simulation of a horse race. You can be creative and use odds from the newspaper, or simulate the Mathematical Derby with the entries and odds shown in following table.

Construct and perform a Monte Carlo simulation of 1000 horse races. Which horse won the most races? Which horse won the fewest races? Do these results surprise you? Provide the tallies of how many races each horse won with your output.

Based on the sample table, Dancin’ Dantzig won the most races. L’Hopital won the fewest races. The result did not suppise me, because the simmulated probability of each horse winning is about same as the theretically winning odds for each horse.

```
set.seed(888)
odds <- c(1/8, 1/6, 1/10, 1/13, 1/5, 1/36, 1/16, 1/5)
entry_name <- c("Euler's Folly", "Leapin Leibniz", "Newton Lobell", "Count Cauchy", "Pumped up P
oisson", "Loping L'Hopital", "Steamin' Stokes", "Dancin' Dantzig")
df1 <- data.frame(entry_name, odds)

sum(df1$odds)
```

```
## [1] 0.9588675
```

```
df1$odds <- df1$odds / sum(odds)

x <- runif(1000, sum(odds))
sim_horse <- sample(entry_name, 1000, replace = TRUE, prob = odds)
table(sim_horse)
```

```
## sim_horse
##      Count Cauchy   Dancin' Dantzig     Euler's Folly    Leapin Leibniz
##                77               224               129               197
##  Loping L'Hopital     Newton Lobell Pumped up Poisson   Steamin' Stokes
##                26               105               187                55
```

```
df2 <- as.data.frame(table(sim_horse))
colnames(df2) <- c("entry_name", "Number of Wins")

df3 <- as.data.frame(prop.table(table(sim_horse)))
colnames(df3) <- c("entry_name", "Simmulated Odds")

df <- merge(df1, df2, by = "entry_name")
df <- merge(df, df3, by = "entry_name")

df
```

```
##          entry_name       odds Number of Wins Simmulated Odds
## 1      Count Cauchy 0.08022284             77           0.077
## 2   Dancin' Dantzig 0.20857939            224           0.224
## 3     Euler's Folly 0.13036212            129           0.129
## 4    Leapin Leibniz 0.17381616            197           0.197
## 5  Loping L'Hopital 0.02896936             26           0.026
## 6     Newton Lobell 0.10428969            105           0.105
## 7 Pumped up Poisson 0.20857939            187           0.187
## 8   Steamin' Stokes 0.06518106             55           0.055
```