Homework 3

View it online: http://acsweb.ucsd.edu/~djc035/Assignment3.html (http://acsweb.ucsd.edu/~djc035/Assignment3.html)

Objective

How do we find clusters of palindromes? How do we determine whether a cluster is just a chance occurrence or a potential replication site?

1.

Locations: Here the goal is to graphically compare your sample palindrome locations to random uniform scatter. To do this, you can visualize the distribution of your sample, the distributions of random uniform scatter instances, and the theoretical uniform distribution. You can visualize the distributions using either histograms or empirical cdfs. Be sure to simulate the random uniform scatter several times (at least 5 times). This is sufficient for the [Random Scatter] section in the homework prompt.

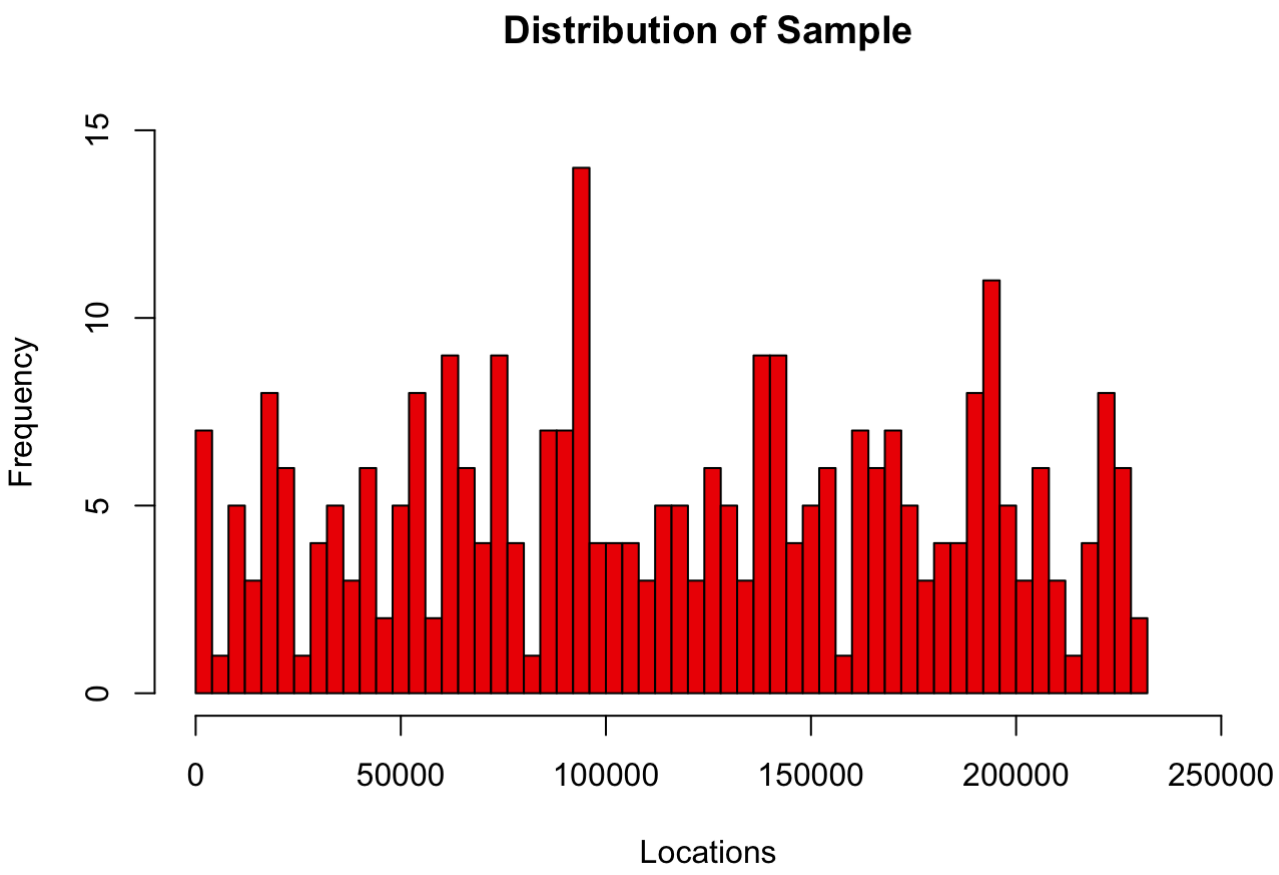
```
#read in "hcmv.txt" into data
data <- read.table("hcmv.txt", header=TRUE)
head(data)
```

```
##   location
## 1      177
## 2     1321
## 3     1433
## 4     1477
## 5     3248
## 6     3255
```

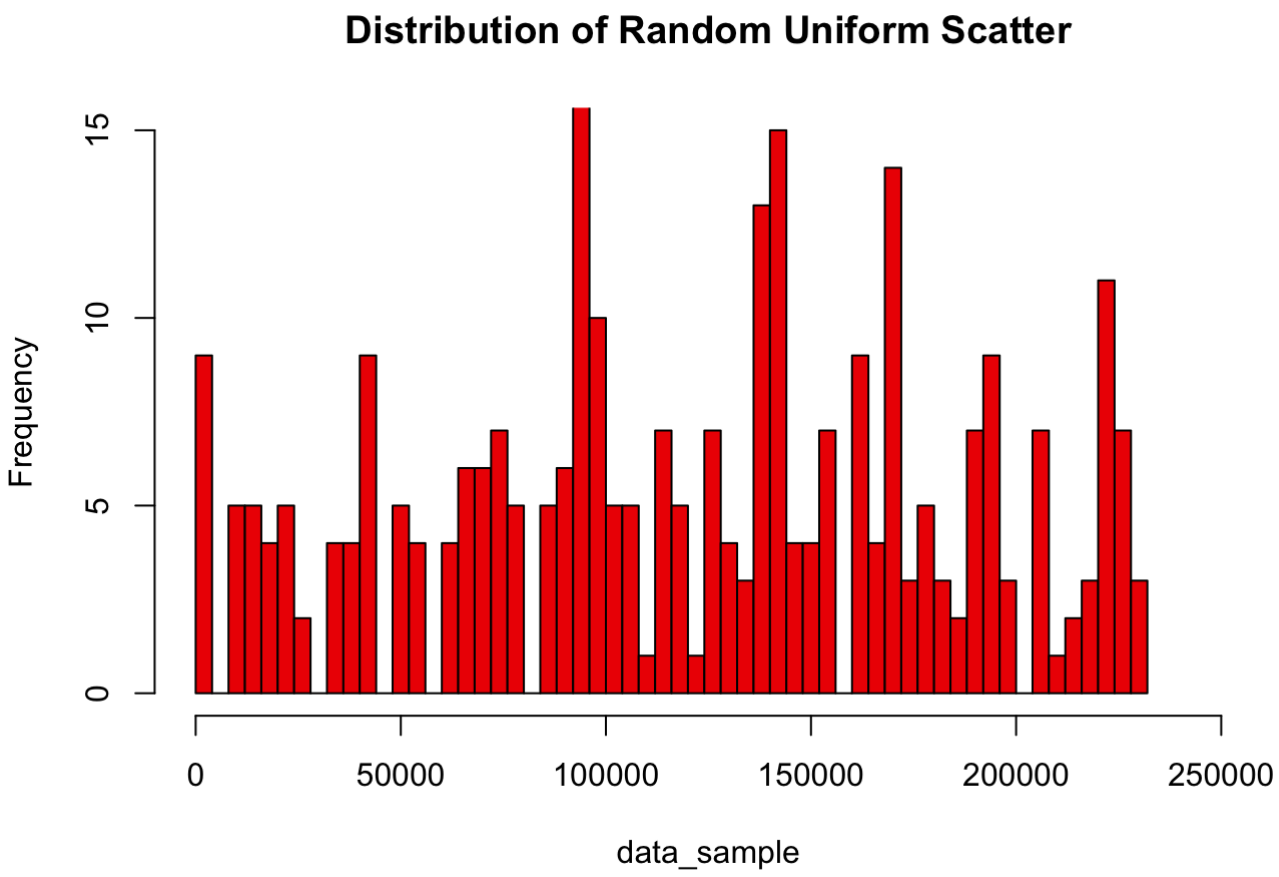
```
nrow(data)
```

```
## [1] 296
```

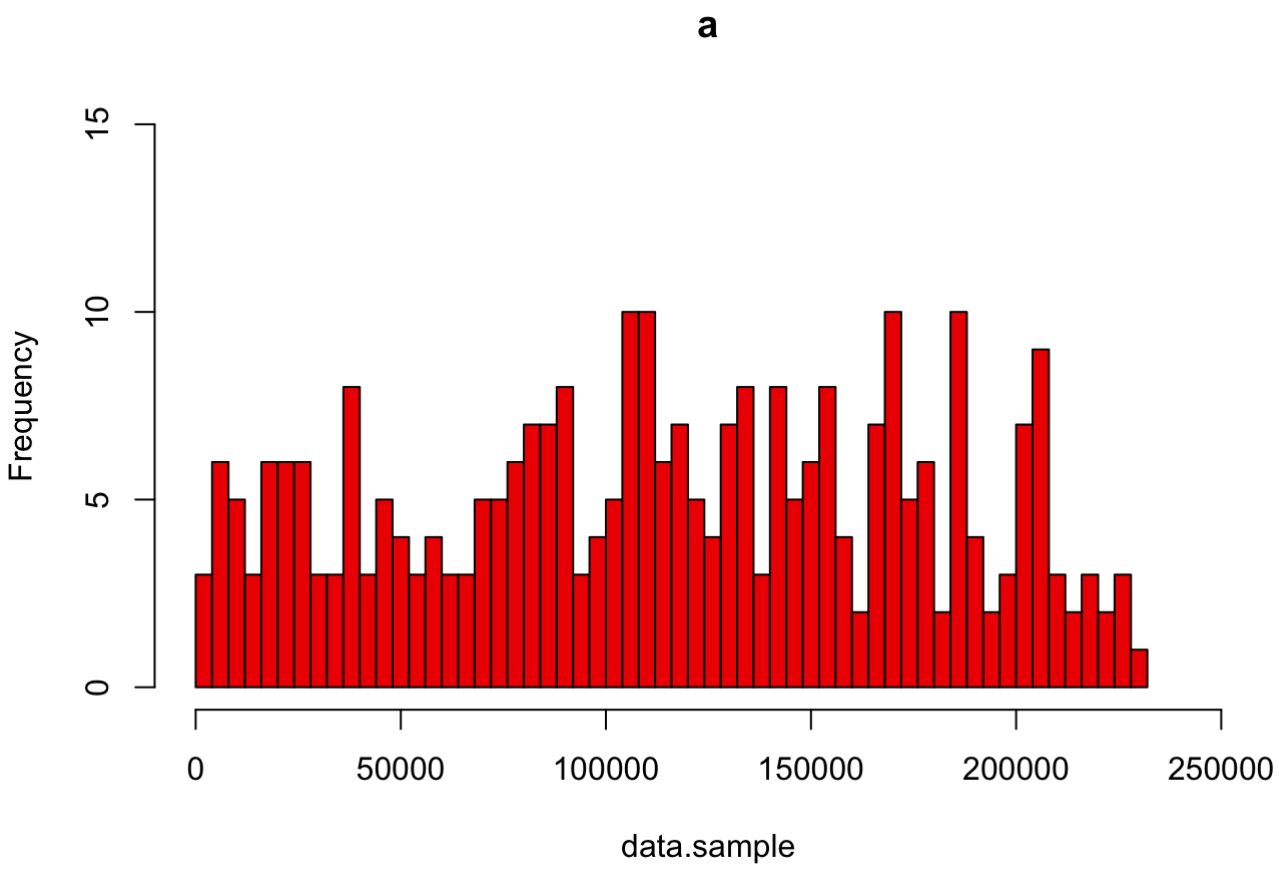
```
hist(data$location, xlim=c(0,250000), ylim=c(0,15), breaks=seq(0,232000, by=4000), col=
'red2', main="Distribution of Sample", xlab="Locations")
```



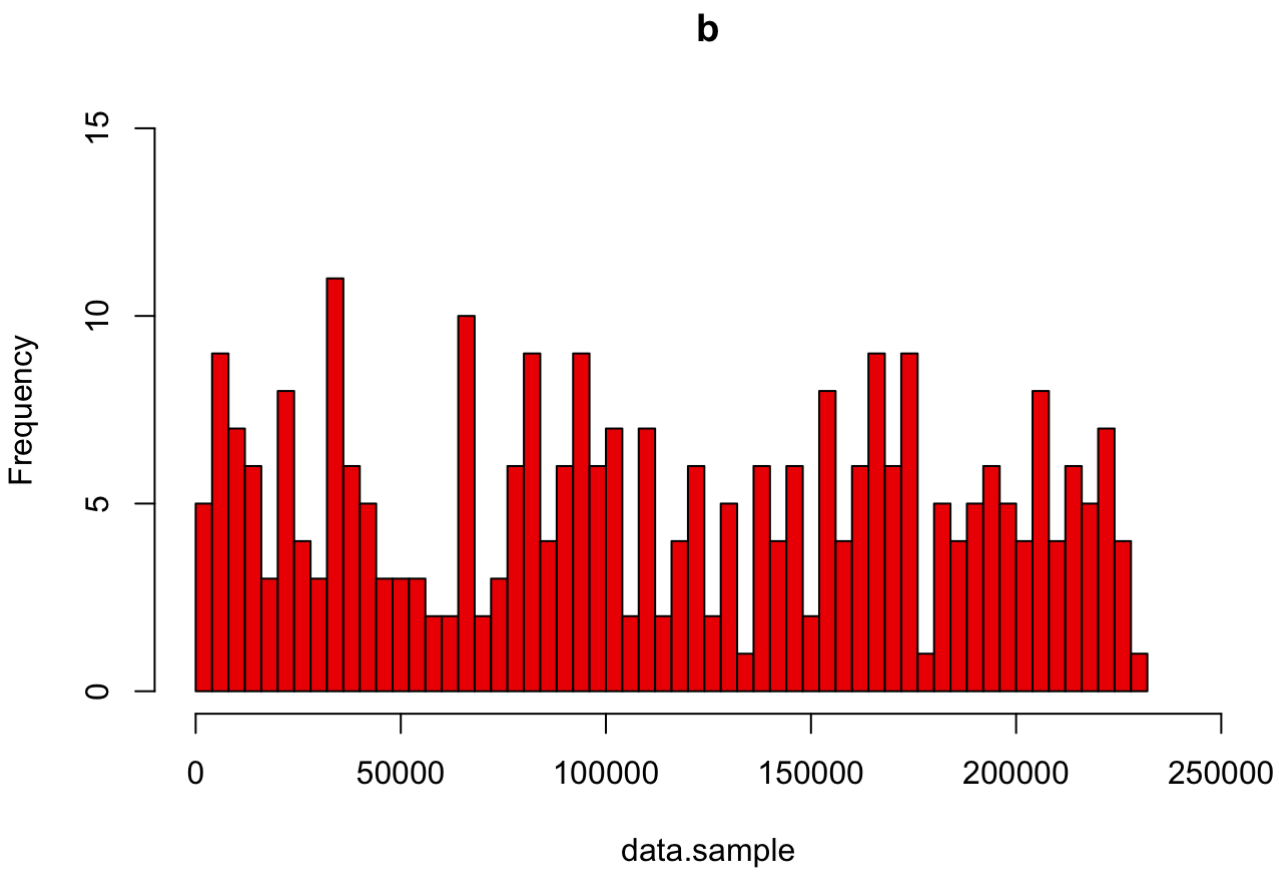

```
set.seed(100)
data_sample = sample(x=data$location, size=296, replace=TRUE)
hist(data_sample, xlim=c(0,250000), ylim=c(0,15), breaks=seq(0,232000, by=4000), col='re
d2', main="Distribution of Random Uniform Scatter")
```



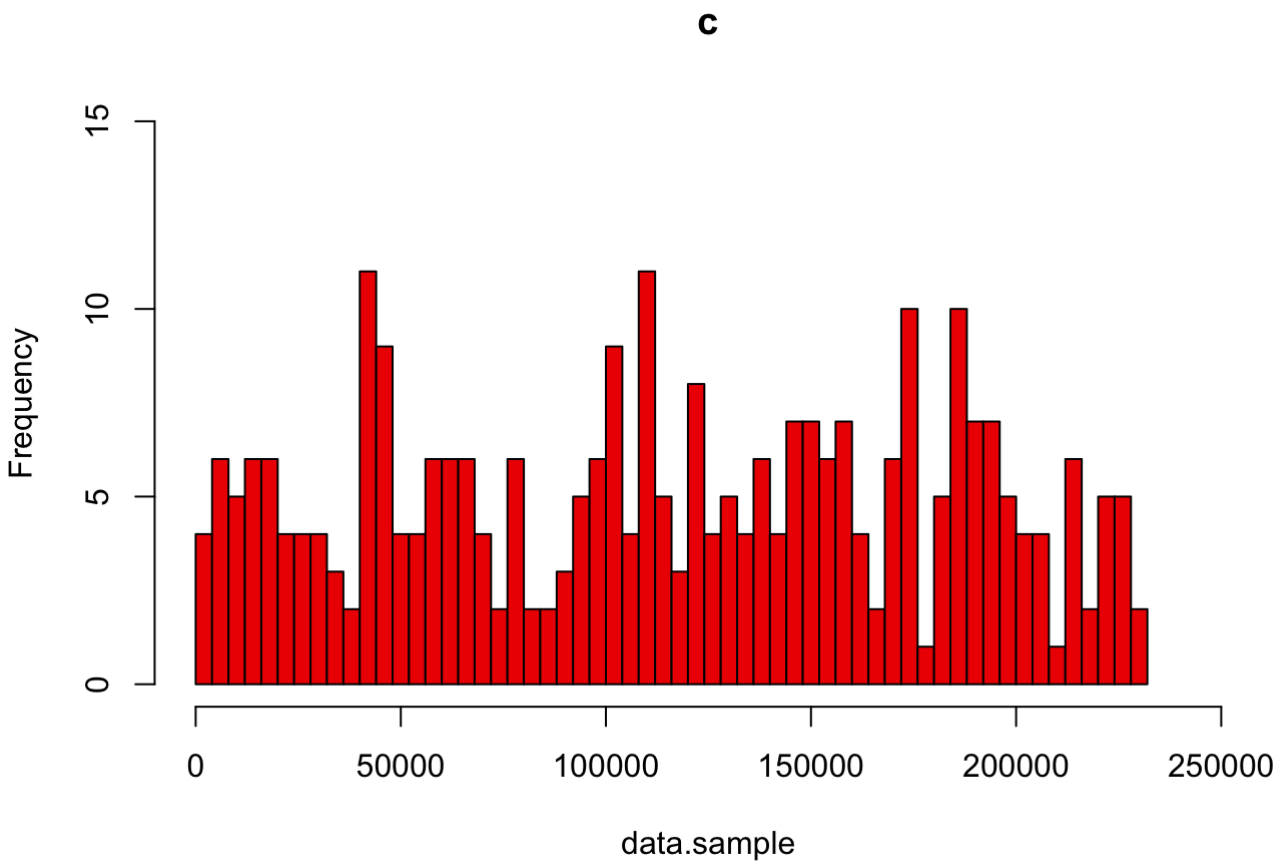

```
N <- max(data$location, na.rm=TRUE)
n <- 296
gene <- seq(177, N)

set.seed(200)
data.sample <- sample.int(N, size=n, replace=FALSE)
aHistogram = hist(data.sample, xlim=c(0,250000), ylim=c(0,15), breaks=seq(0,232000, by=4
000), col='red2', main="a")
```



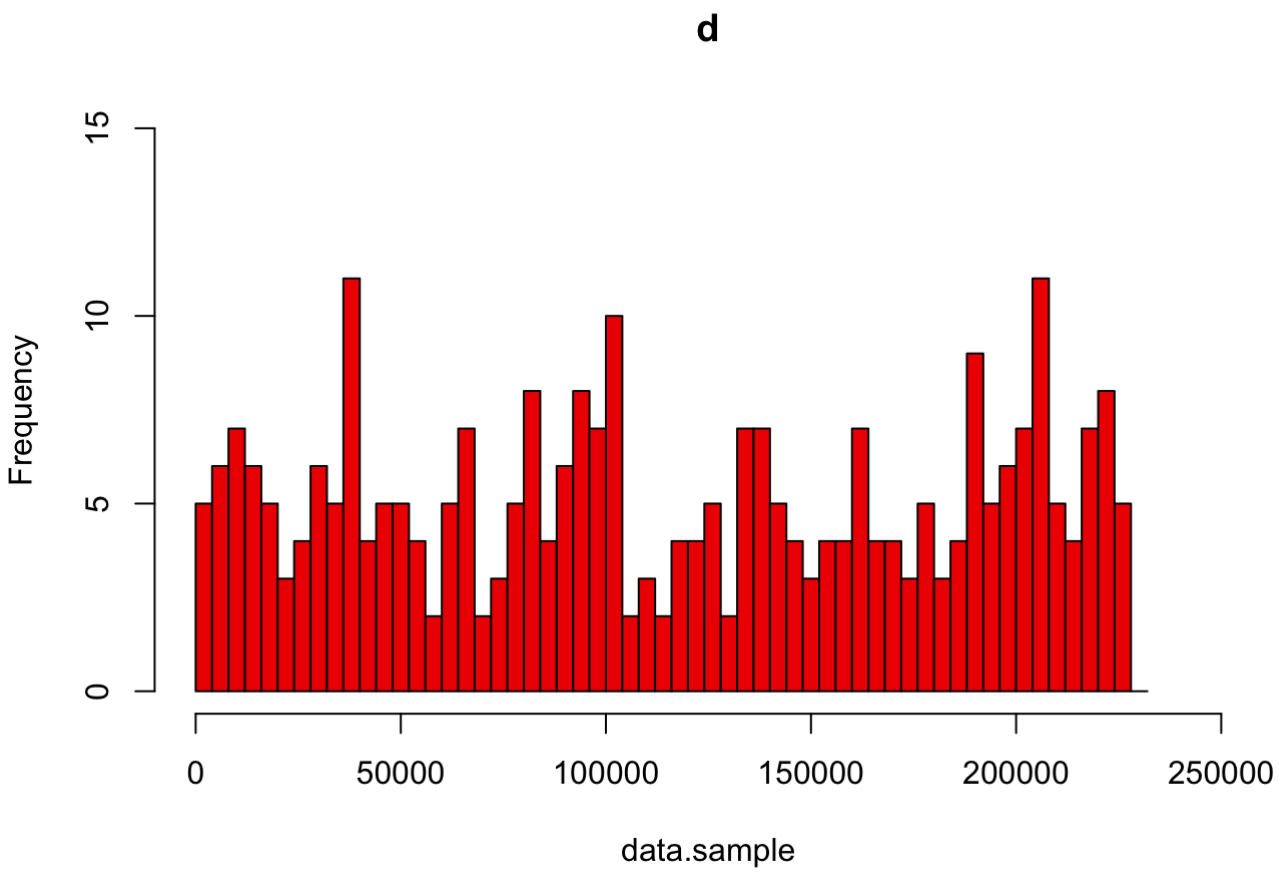

```
data.sample <- sample.int(N, size=n, replace=FALSE)
bHistogram = hist(data.sample, xlim=c(0,250000), ylim=c(0,15), breaks=seq(0,232000, by=4
000), col='red2', main="b")
```



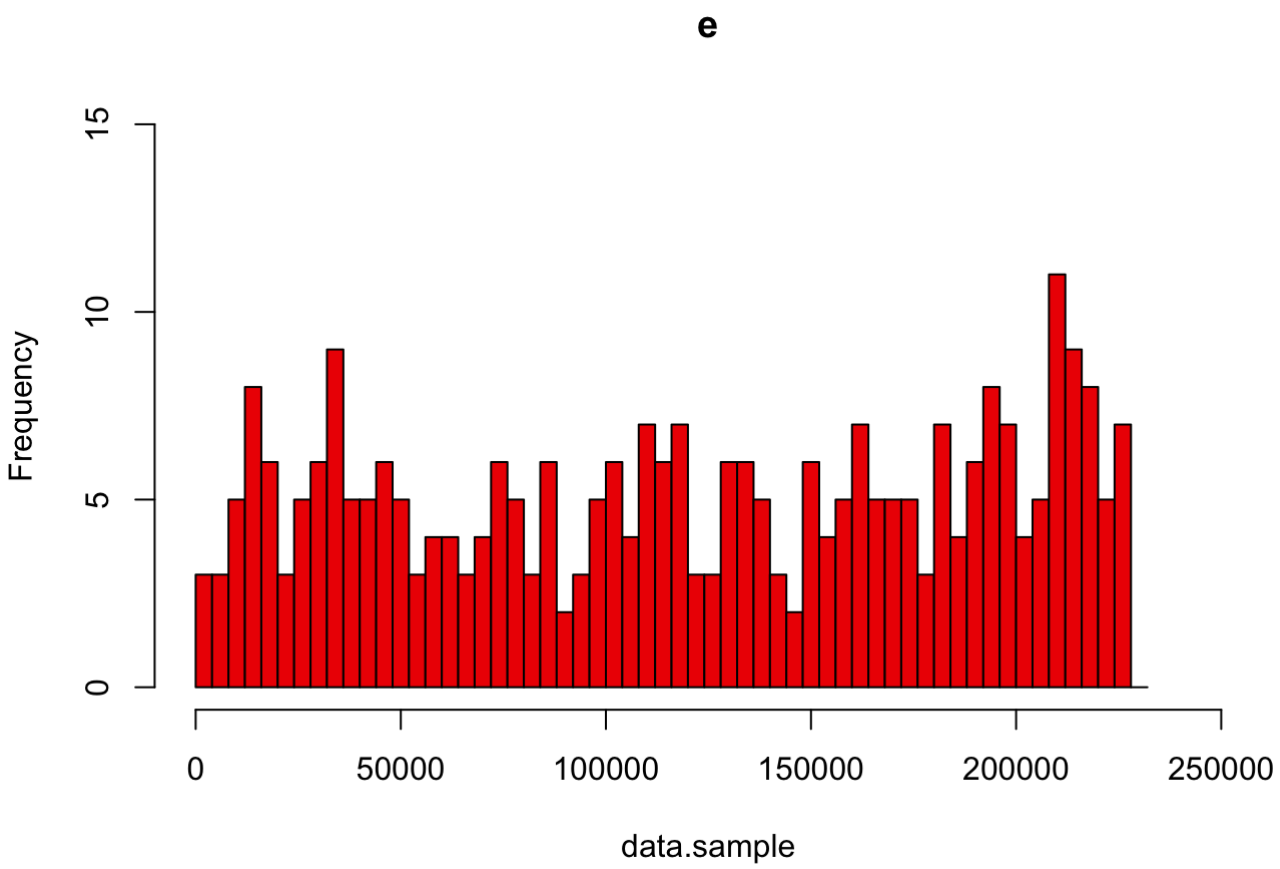

```
data.sample <- sample.int(N, size=n, replace=FALSE)
cHistogram = hist(data.sample, xlim=c(0,250000), ylim=c(0,15), breaks=seq(0,232000, by=4
000), col='red2', main="c")
```



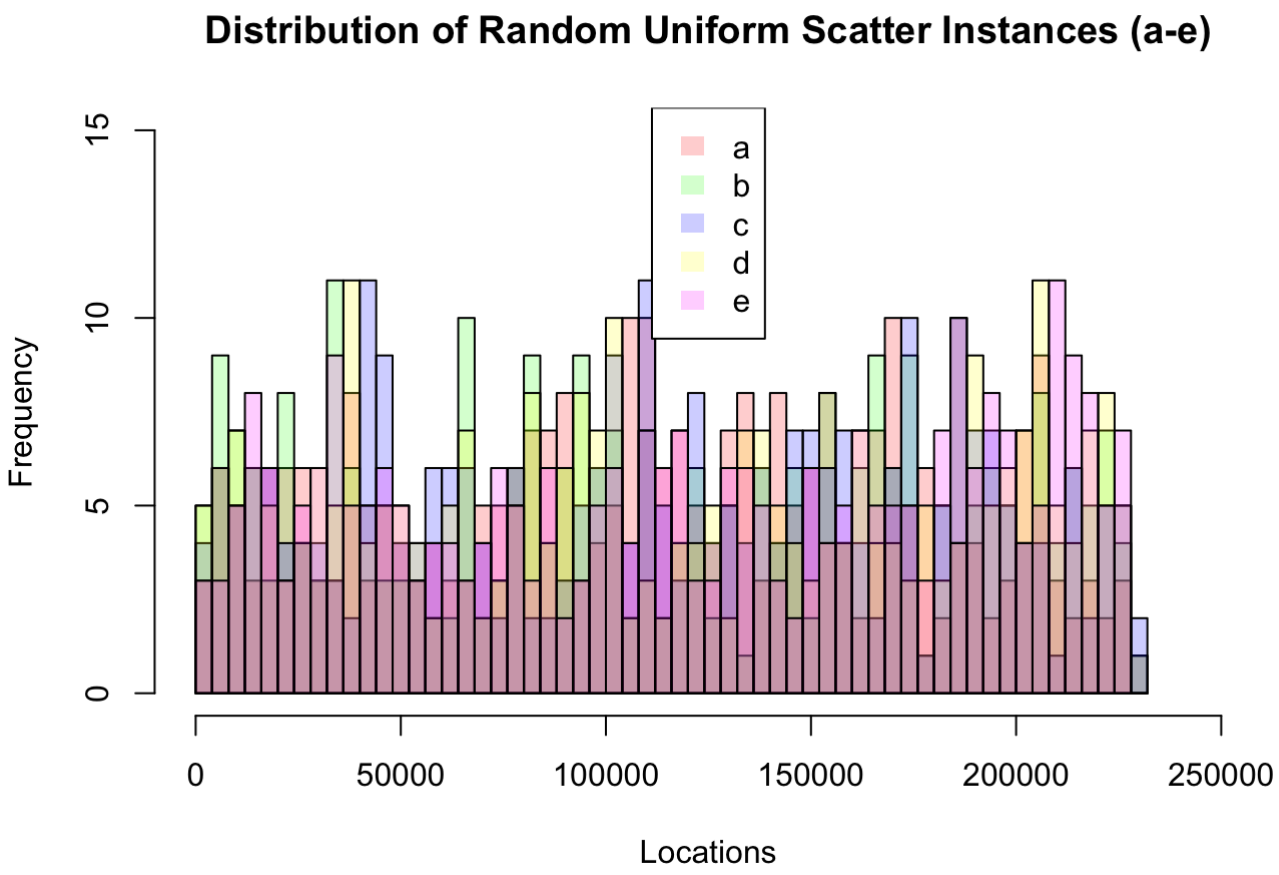

```
data.sample <- sample.int(N, size=n, replace=FALSE)
dHistogram = hist(data.sample, xlim=c(0,250000), ylim=c(0,15), breaks=seq(0,232000, by=4
000), col='red2', main="d")
```



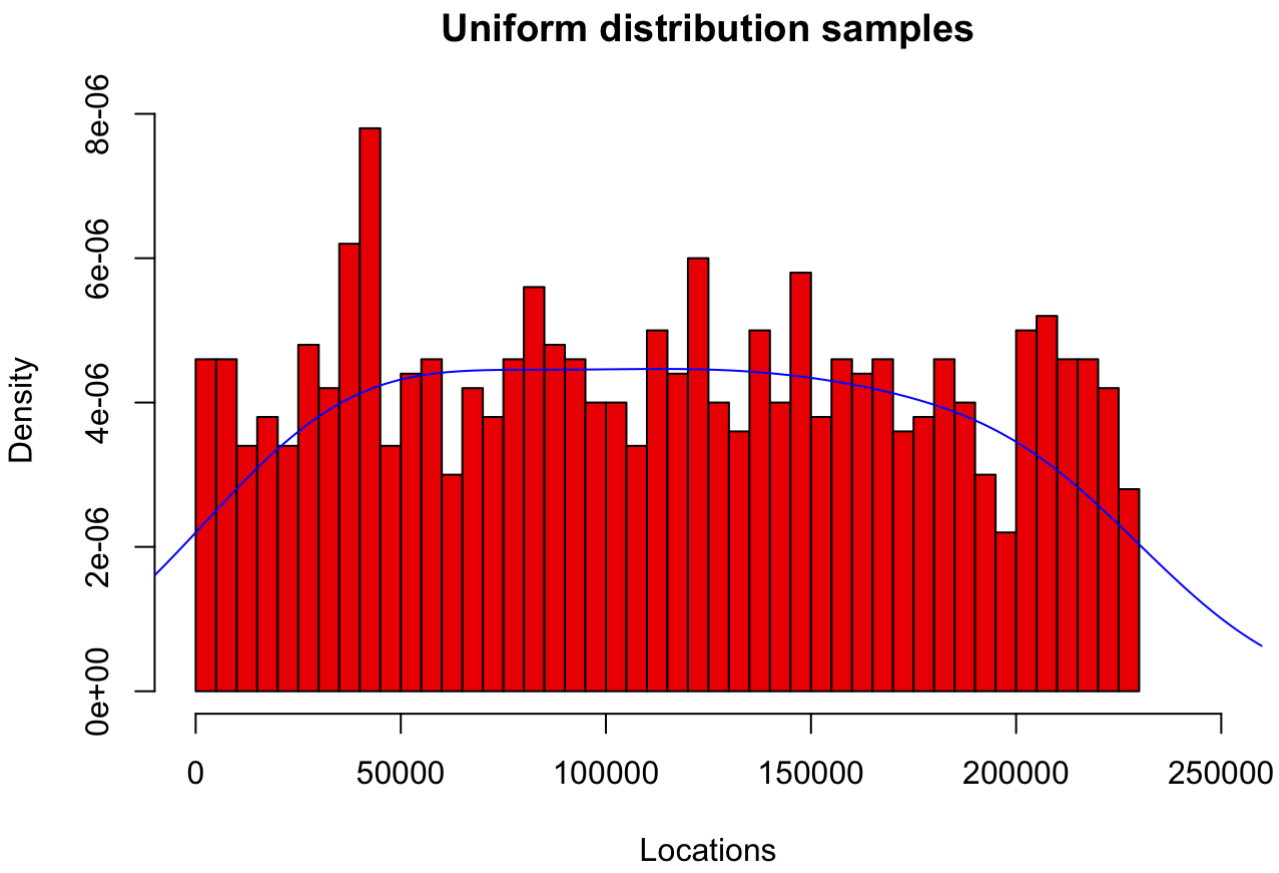

```
data.sample <- sample.int(N, size=n, replace=FALSE)
eHistogram = hist(data.sample, xlim=c(0,250000), ylim=c(0,15), breaks=seq(0,232000, by=4
000), col='red2', main="e")
```




```
aHistColor = rgb(1,0,0,0.2)
bHistColor = rgb(0,1,0,0.2)
cHistColor = rgb(0,0,1,0.2)
dHistColor = rgb(1,1,0,0.2)
eHistColor = rgb(1,0,1,0.2)

plot(aHistogram, col=aHistColor, ylim=c(0,15), xlim=c(0,250000), main="Distribution of R
andom Uniform Scatter Instances (a-e)", xlab="Locations")
plot(bHistogram, col=bHistColor, ylim=c(0,15), xlim=c(0,250000), add=T)
plot(cHistogram, col=cHistColor, ylim=c(0,15), xlim=c(0,250000), add=T)
plot(dHistogram, col=dHistColor, ylim=c(0,15), xlim=c(0,250000), add=T)
plot(eHistogram, col=eHistColor, ylim=c(0,15), xlim=c(0,250000), add=T)

legend('top', c('a', 'b', 'c', 'd', 'e'), fill=c(aHistColor, bHistColor, cHistColor, dHi
stColor, eHistColor), border=NA)
```




```
set.seed(2017)
n <- 1000
data_unif <- runif(n, min=177, max=228953)
hist(data_unif, xlim=c(0,250000), breaks=58, probability=TRUE, col='red2', main = "Unifo
rm distribution samples", xlab="Locations")
lines(density(data_unif, adjust=2), col=4)
```



2.

Spacings: Here the goal is to graphically examine the distribution of your sample spacings. There are 3 types of spacings to examine: spacings between consecutive palindromes, spacings between palindromes with one in between (i.e. sums of pairs of consecutive spacings), and spacings between palindromes with two in between (i.e. sums of triplets of consecutive spacings). Next, you can graphically compare these 3 types of spacings to those that come from random uniform scatter (using empirical cdf or histograms). Again, you should simulate at least 5 random uniform scatters. Lastly, using theoretical results discussed in lecture, identify the theoretical distributions of the spacings from random uniform scatter (you won't be expected to know the distribution for the sums of consecutive triplets but it shouldn't be too hard to intuit). Overlay these theoretical distributions as a cdf or density on your plots.

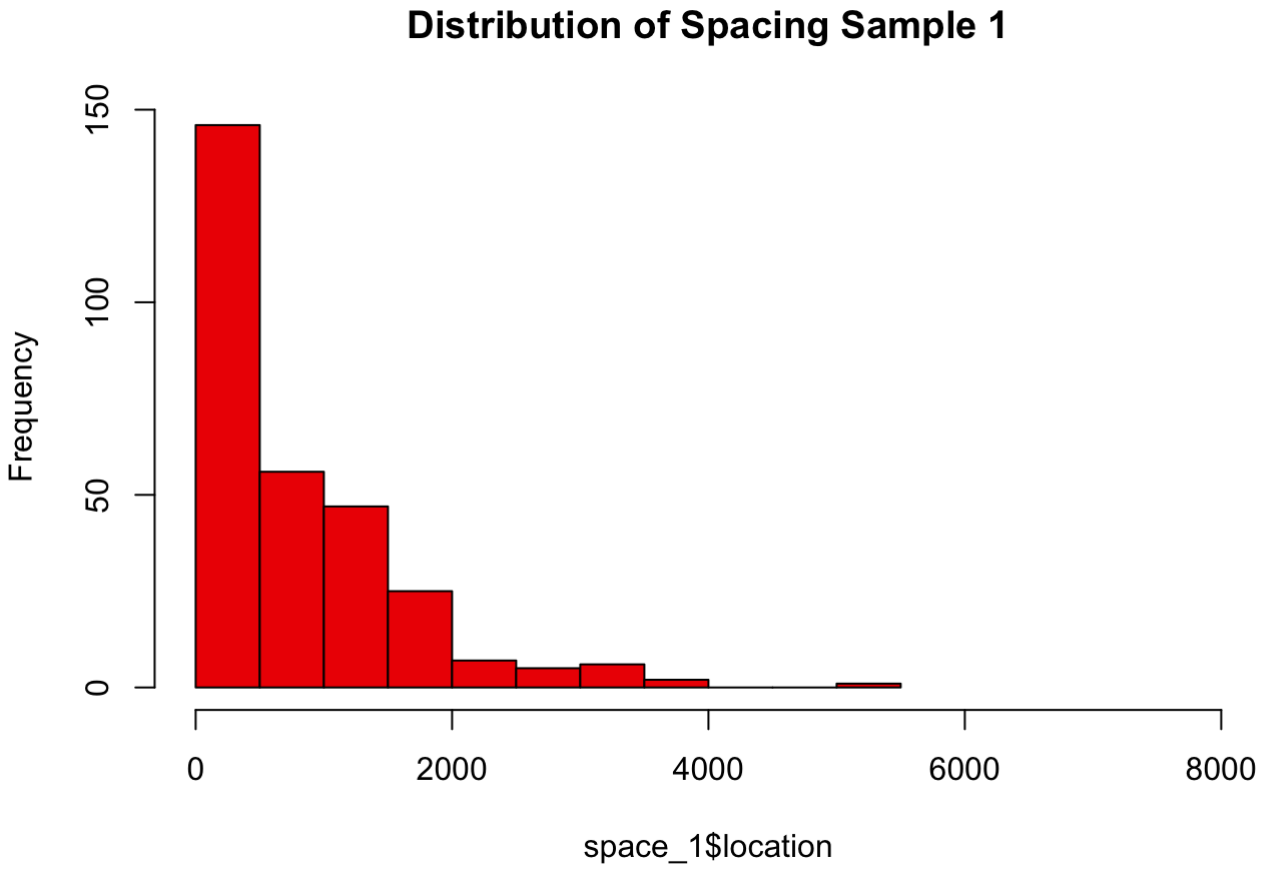
2a)

Examine 3 types of spacings by plotting histograms

```
space_1 = data.frame(diff(as.matrix(data)))
head(space_1)
```

```
##   location
## 1     1144
## 2      112
## 3       44
## 4     1771
## 5        7
## 6       31
```

```
hist_1 = hist(space_1$location, xlim=c(0,8000), breaks=16, col='red2', main="Distributio
n of Spacing Sample 1")
```




```
library(zoo)
```

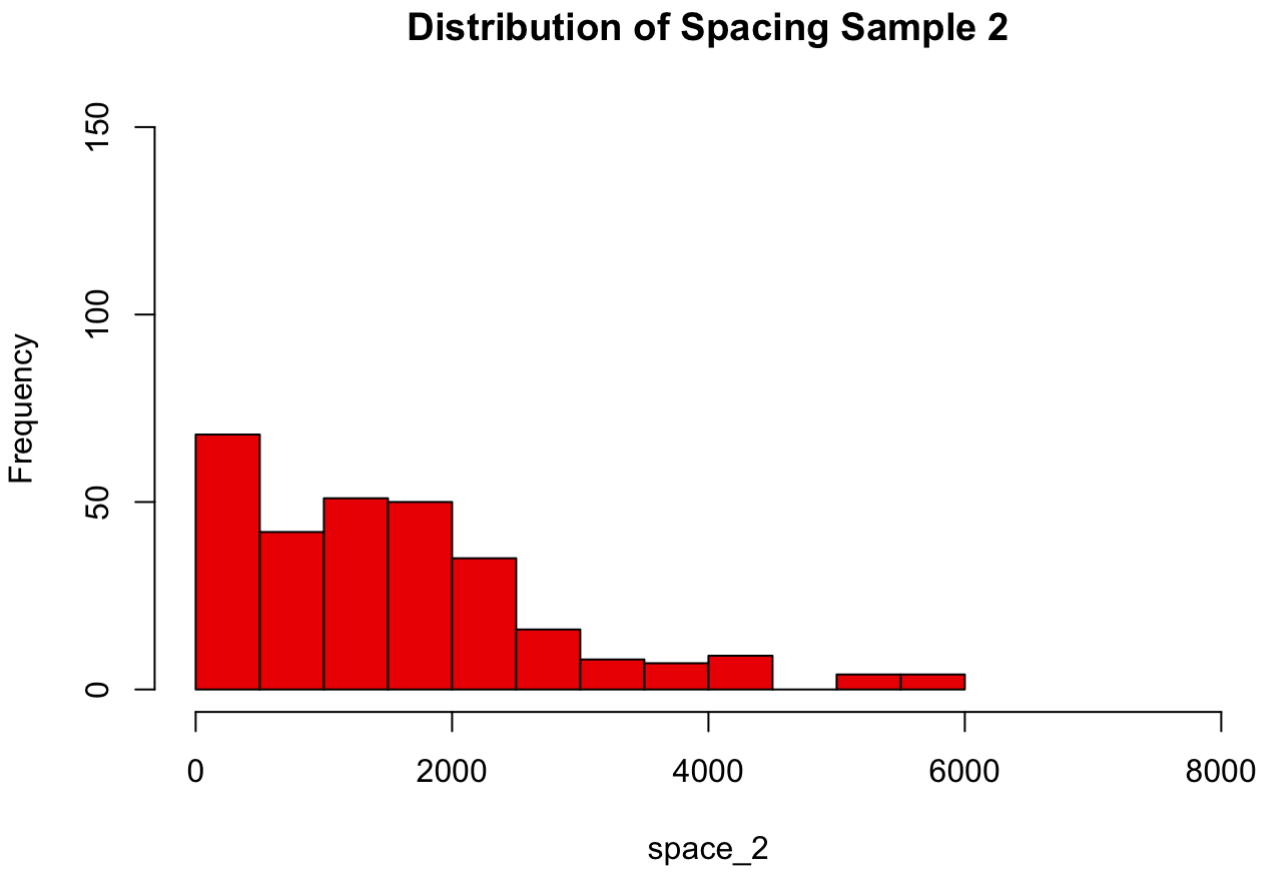
```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
space_2 = rollapply(space_1, 2, sum)
head(space_2)
```

```
##      location
## [1,]     1256
## [2,]      156
## [3,]     1815
## [4,]     1778
## [5,]       38
## [6,]     4008
```

```
hist_2 = hist(space_2, xlim=c(0,8000), ylim=c(0,150), breaks=16, col='red2', main="Distr
ibution of Spacing Sample 2")
```




```
space_2
```

```
##        location
##   [1,]     1256
##   [2,]      156
##   [3,]     1815
##   [4,]     1778
##   [5,]       38
##   [6,]     4008
##   [7,]     5737
##   [8,]     1821
##   [9,]      310
##  [10,]     1800
##  [11,]     2421
##  [12,]     1979
##  [13,]     2509
##  [14,]     1856
##  [15,]     1750
##  [16,]     1706
##  [17,]      739
##  [18,]      387
##  [19,]     1257
##  [20,]     2364
##  [21,]     1316
##  [22,]      239
##  [23,]      705
##  [24,]     1417
##  [25,]     1997
##  [26,]     1907
##  [27,]      883
##  [28,]      502
##  [29,]     3039
##  [30,]     5424
##  [31,]     4429
##  [32,]     2325
##  [33,]     1125
##  [34,]     1933
##  [35,]     2600
##  [36,]     1475
##  [37,]      300
##  [38,]      325
##  [39,]     2193
##  [40,]     1984
##  [41,]     2030
##  [42,]     3847
##  [43,]     2474
##  [44,]      668
##  [45,]     1276
##  [46,]     2253
##  [47,]     1320
##  [48,]     1713
##  [49,]     4209
##  [50,]     3091
##  [51,]      465
##  [52,]      420
```

```
##  [53,]     2800
##  [54,]     2762
##  [55,]     1073
##  [56,]     1168
##  [57,]     1196
##  [58,]     1049
##  [59,]      573
##  [60,]      359
##  [61,]      130
##  [62,]     1038
##  [63,]     2553
##  [64,]     2048
##  [65,]     3373
##  [66,]     3251
##  [67,]      484
##  [68,]     1067
##  [69,]     2394
##  [70,]     1562
##  [71,]       77
##  [72,]      546
##  [73,]      746
##  [74,]      953
##  [75,]     1786
##  [76,]     1287
##  [77,]      247
##  [78,]      226
##  [79,]     1803
##  [80,]     2206
##  [81,]     2128
##  [82,]     2579
##  [83,]     1524
##  [84,]     1420
##  [85,]     1296
##  [86,]     1833
##  [87,]     1506
##  [88,]      488
##  [89,]     1563
##  [90,]     1234
##  [91,]      190
##  [92,]      103
##  [93,]      231
##  [94,]      246
##  [95,]     1599
##  [96,]     3600
##  [97,]     5391
##  [98,]     5406
##  [99,]     2480
## [100,]      399
## [101,]      127
## [102,]      602
## [103,]      497
## [104,]     1586
## [105,]     2666
## [106,]     1869
```

```
## [107,]     1448
## [108,]     1177
## [109,]     1239
## [110,]      874
## [111,]      463
## [112,]      889
## [113,]      617
## [114,]      117
## [115,]      131
## [116,]       66
## [117,]       46
## [118,]       74
## [119,]      112
## [120,]      327
## [121,]      391
## [122,]      401
## [123,]      351
## [124,]      663
## [125,]     2374
## [126,]     3314
## [127,]     2518
## [128,]     1420
## [129,]     1216
## [130,]     1956
## [131,]     2430
## [132,]     1404
## [133,]      572
## [134,]     2095
## [135,]     1791
## [136,]     1171
## [137,]     2912
## [138,]     2589
## [139,]     1771
## [140,]     2101
## [141,]     4193
## [142,]     3917
## [143,]     2249
## [144,]     1653
## [145,]      191
## [146,]     1303
## [147,]     2737
## [148,]     2568
## [149,]     1202
## [150,]      312
## [151,]      654
## [152,]      455
## [153,]      959
## [154,]     4282
## [155,]     4176
## [156,]     2101
## [157,]     1478
## [158,]      231
## [159,]      563
## [160,]     1755
```

```
## [161,]     1470
## [162,]      736
## [163,]     2143
## [164,]     2197
## [165,]     1840
## [166,]     2487
## [167,]     2321
## [168,]     1830
## [169,]     1044
## [170,]      527
## [171,]      465
## [172,]      802
## [173,]      723
## [174,]      315
## [175,]      518
## [176,]     1385
## [177,]     2468
## [178,]     2121
## [179,]     1415
## [180,]     1215
## [181,]      997
## [182,]      836
## [183,]      558
## [184,]      303
## [185,]      189
## [186,]     3112
## [187,]     3874
## [188,]     1100
## [189,]      266
## [190,]      766
## [191,]      943
## [192,]     1523
## [193,]     2493
## [194,]     1750
## [195,]      731
## [196,]      416
## [197,]      286
## [198,]     2249
## [199,]     2742
## [200,]     1447
## [201,]     2544
## [202,]     5123
## [203,]     3699
## [204,]     1641
## [205,]     1387
## [206,]       33
## [207,]     1042
## [208,]     1280
## [209,]      327
## [210,]     1076
## [211,]     1811
## [212,]      820
## [213,]       48
## [214,]      481
```

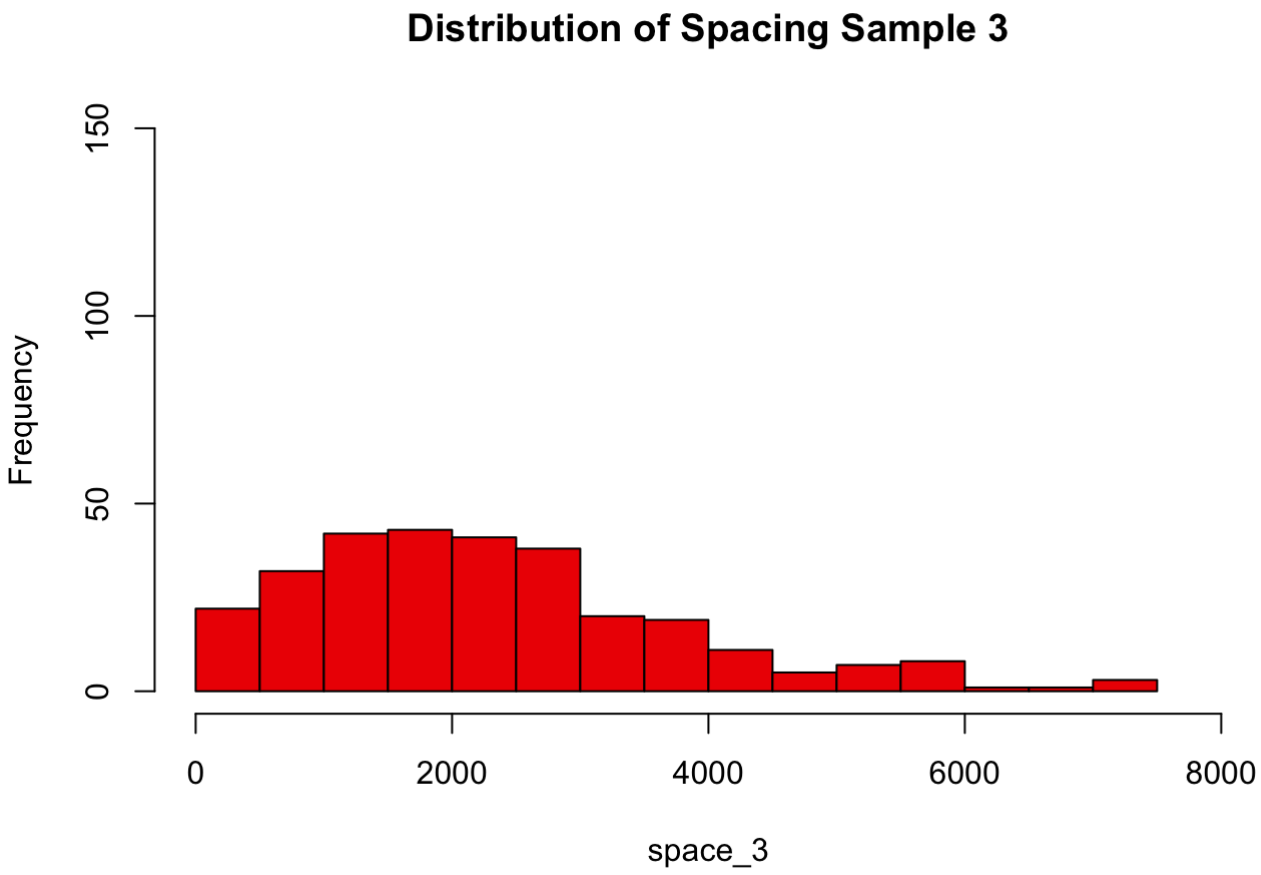
```
## [215,]    2330
## [216,]    2338
## [217,]     554
## [218,]    1635
## [219,]    2173
## [220,]     644
## [221,]     619
## [222,]    2874
## [223,]    2442
## [224,]     269
## [225,]     136
## [226,]     128
## [227,]    3542
## [228,]    3696
## [229,]     847
## [230,]    2169
## [231,]    1800
## [232,]     310
## [233,]    1821
## [234,]    5737
## [235,]    4008
## [236,]      38
## [237,]    1778
## [238,]    1815
## [239,]     156
## [240,]    1256
## [241,]    1673
## [242,]    1637
## [243,]    1175
## [244,]      78
## [245,]     313
## [246,]    1531
## [247,]    2149
## [248,]    1375
## [249,]     664
## [250,]    1130
## [251,]    1001
## [252,]      85
## [253,]      39
## [254,]     104
## [255,]     111
## [256,]     614
## [257,]    1730
## [258,]    1187
## [259,]     199
## [260,]    1173
## [261,]    1518
## [262,]    2828
## [263,]    2347
## [264,]    1175
## [265,]    3492
## [266,]    3305
## [267,]    1452
## [268,]    2024
```

```
## [269,]     1788
## [270,]      371
## [271,]      784
## [272,]     1978
## [273,]     1897
## [274,]     5926
## [275,]     5721
## [276,]      490
## [277,]      349
## [278,]      784
## [279,]     4010
## [280,]     4451
## [281,]     1400
## [282,]      632
## [283,]      624
## [284,]      660
## [285,]      428
## [286,]      725
## [287,]     1993
## [288,]     2268
## [289,]     1942
## [290,]     1426
## [291,]      313
## [292,]       78
## [293,]     1175
## [294,]     1637
```

```
library(zoo)
space_3 = rollapply(space_1, 3, sum)
head(space_3)
```

```
##      location
## [1,]     1300
## [2,]     1927
## [3,]     1822
## [4,]     1809
## [5,]     4015
## [6,]     5768
```

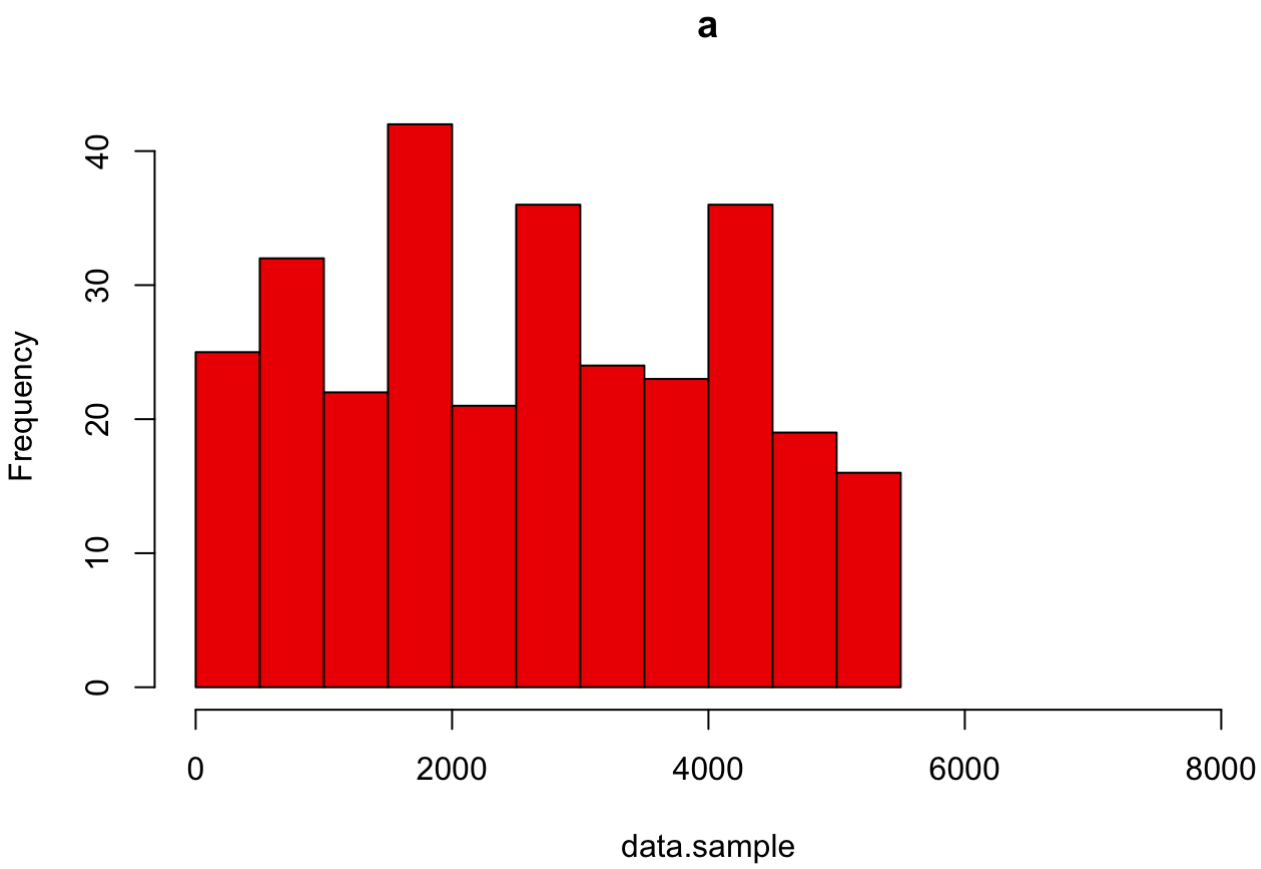
```
hist_3 = hist(space_3, xlim=c(0,8000), ylim=c(0,150), breaks=16, col='red2', main="Distr
ibution of Spacing Sample 3")
```



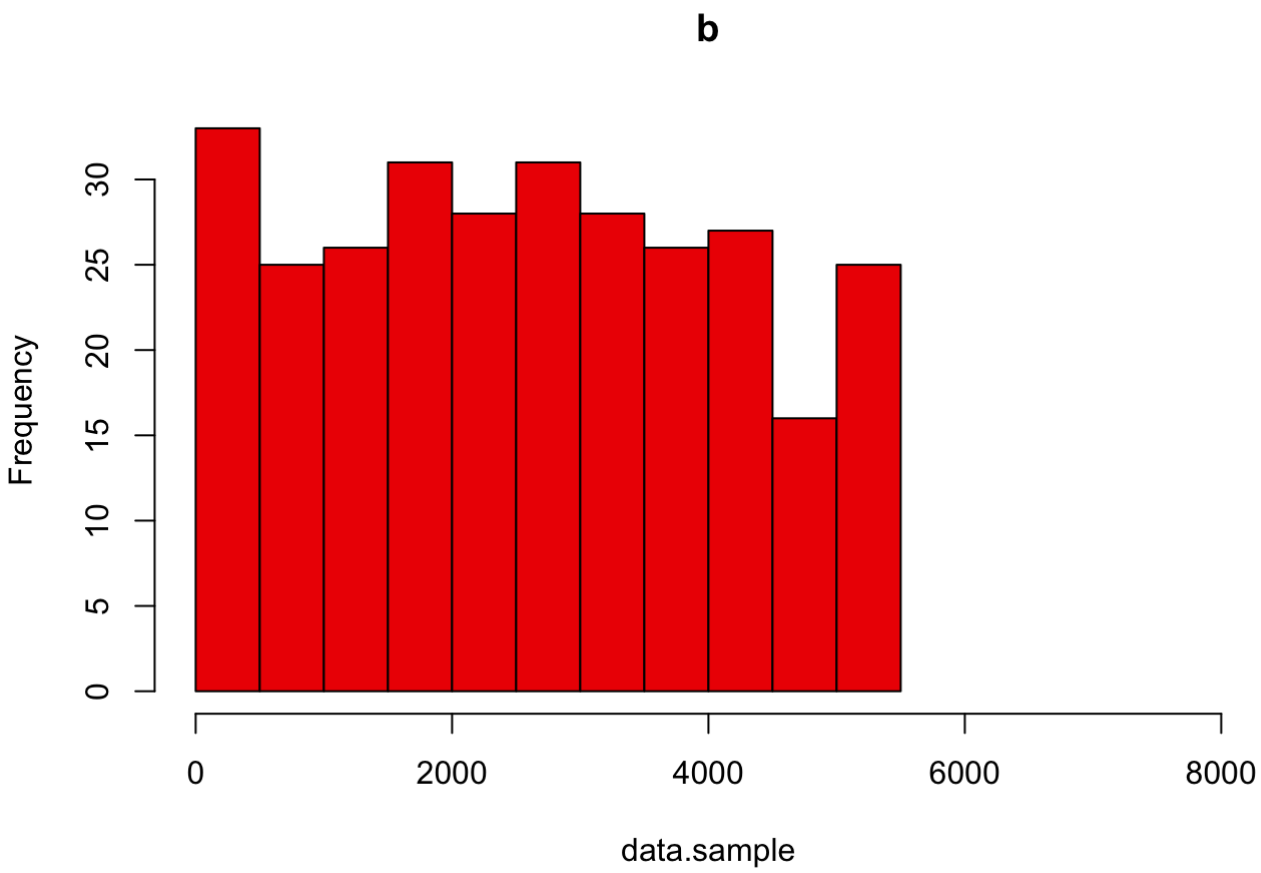
2b)

Plot 5x3 histograms of the random uniform scatters (5 random uniform scatters, 3 types of spacings for each)

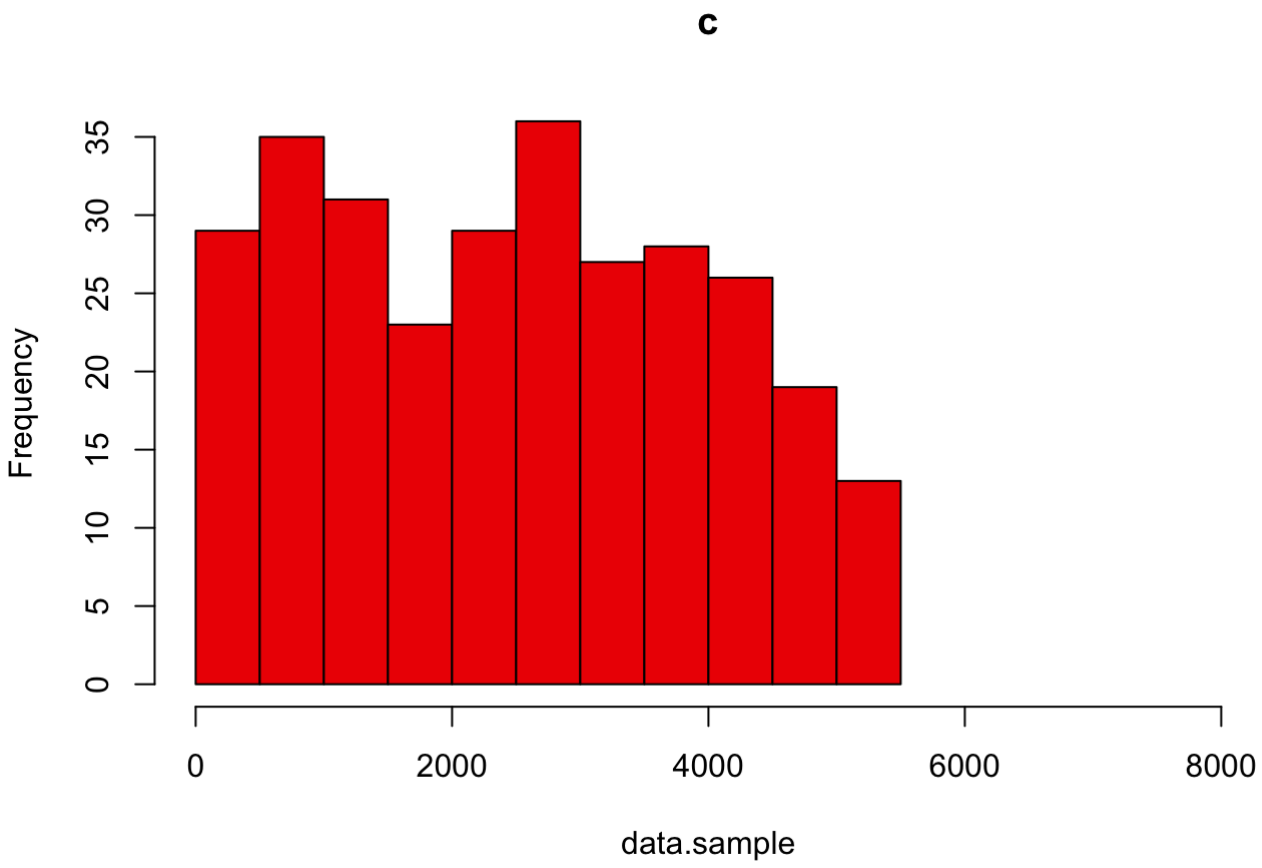
```
N <- max(space_1$location, na.rm=TRUE)
n <- 296
gene <- seq(min(space_1$location, na.rm=TRUE), N)

set.seed(200)
data.sample <- sample.int(N, size=n, replace=FALSE)
aHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="a")
```



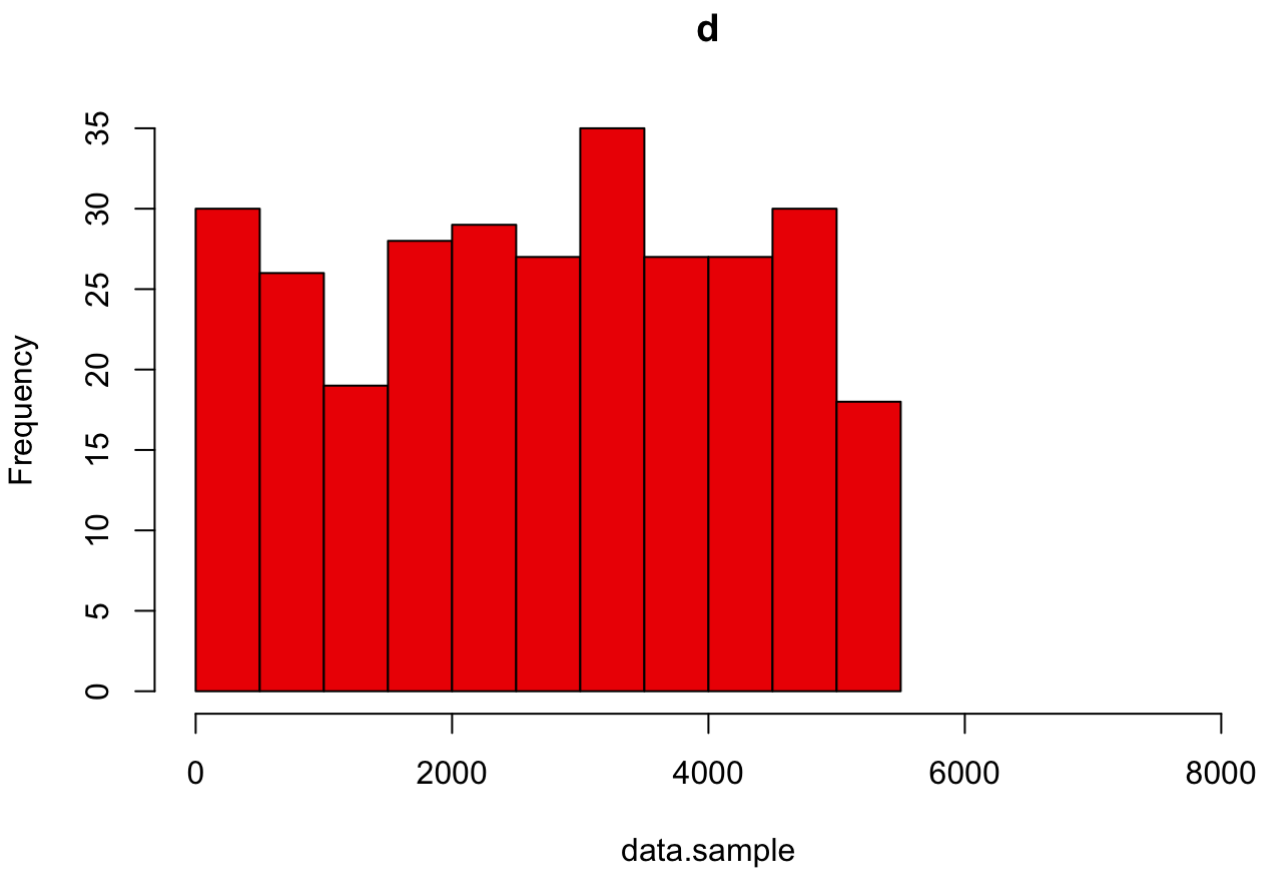

```
aHistogram = data.sample


data.sample <- sample.int(N, size=n, replace=FALSE)
bHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="b")
```



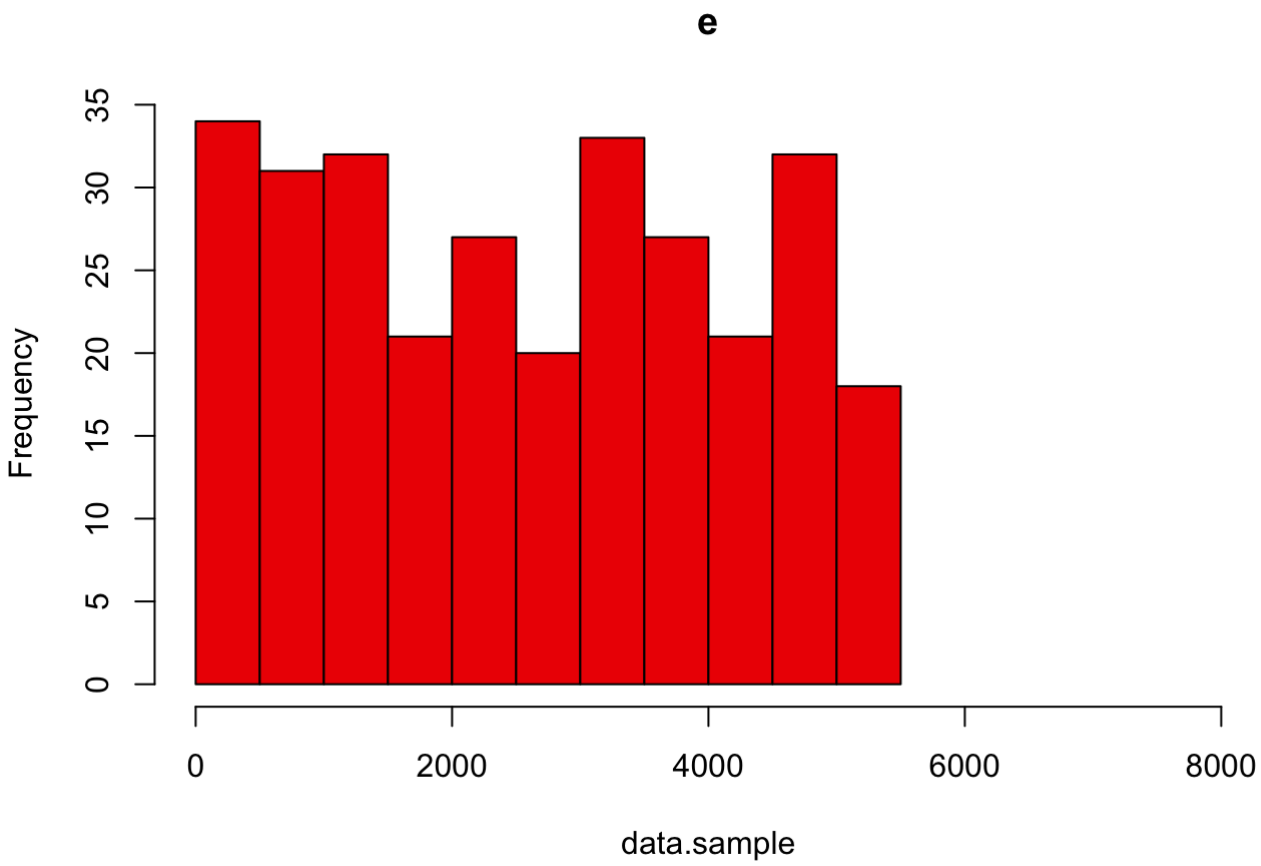

```
bHistogram = data.sample

data.sample <- sample.int(N, size=n, replace=FALSE)
cHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="c")
```



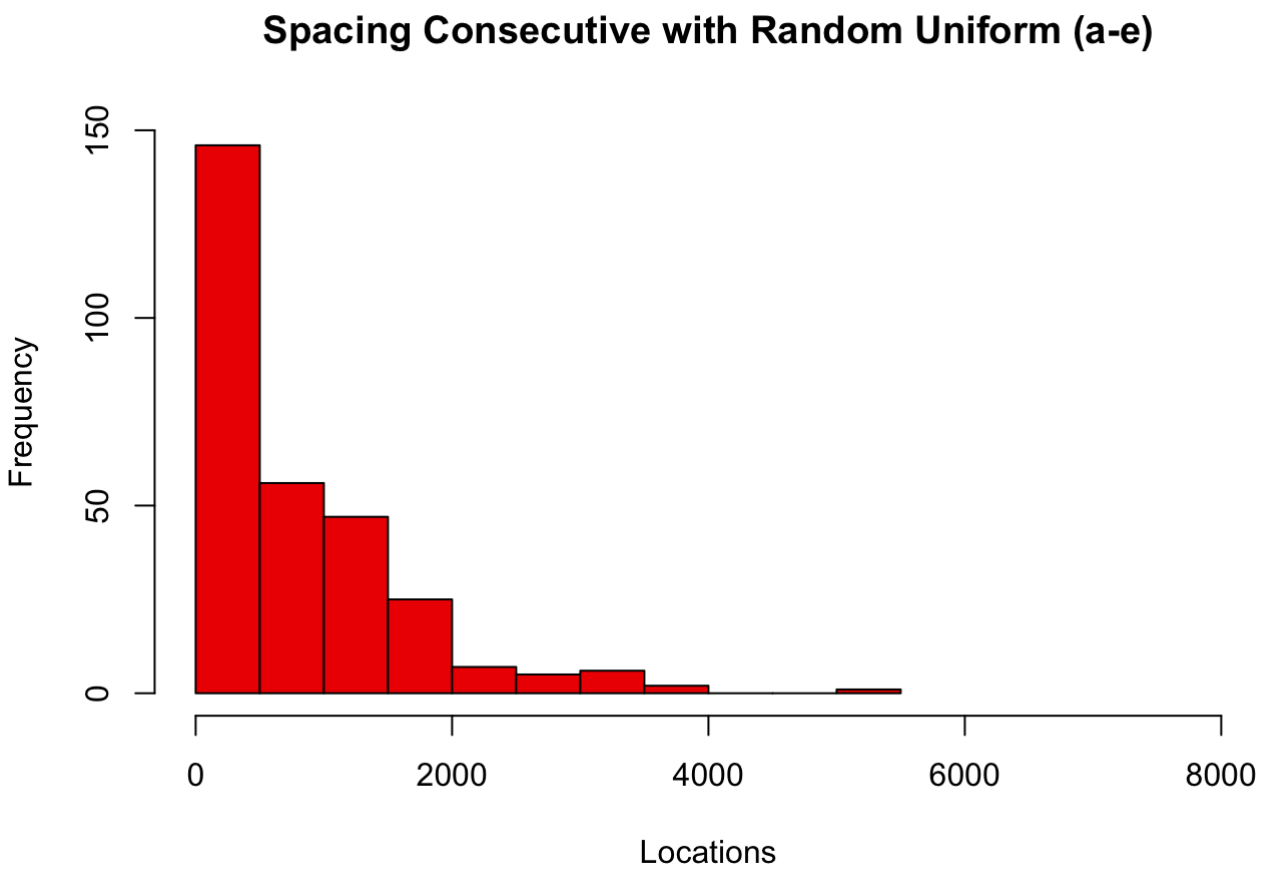

```
cHistogram = data.sample

data.sample <- sample.int(N, size=n, replace=FALSE)
dHistogram = hist(data.sample, xlim=c(0,8000), breaks=16,col='red2', main="d")
```




```
dHistogram = data.sample

data.sample <- sample.int(N, size=n, replace=FALSE)
eHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="e")
```




```
eHistogram = data.sample

aHistColor = rgb(1,0,0)
bHistColor = rgb(0,1,0,0.2)
cHistColor = rgb(0,0,1,0.2)
dHistColor = rgb(1,1,0,0.2)
eHistColor = rgb(1,0,1,0.2)

plot(hist_1, col='red2', ylim=c(0,150), xlim=c(0,8000), main="Spacing Consecutive with R
andom Uniform (a-e)", xlab="Locations")
```




```
plot(aHistogram, col=aHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
```

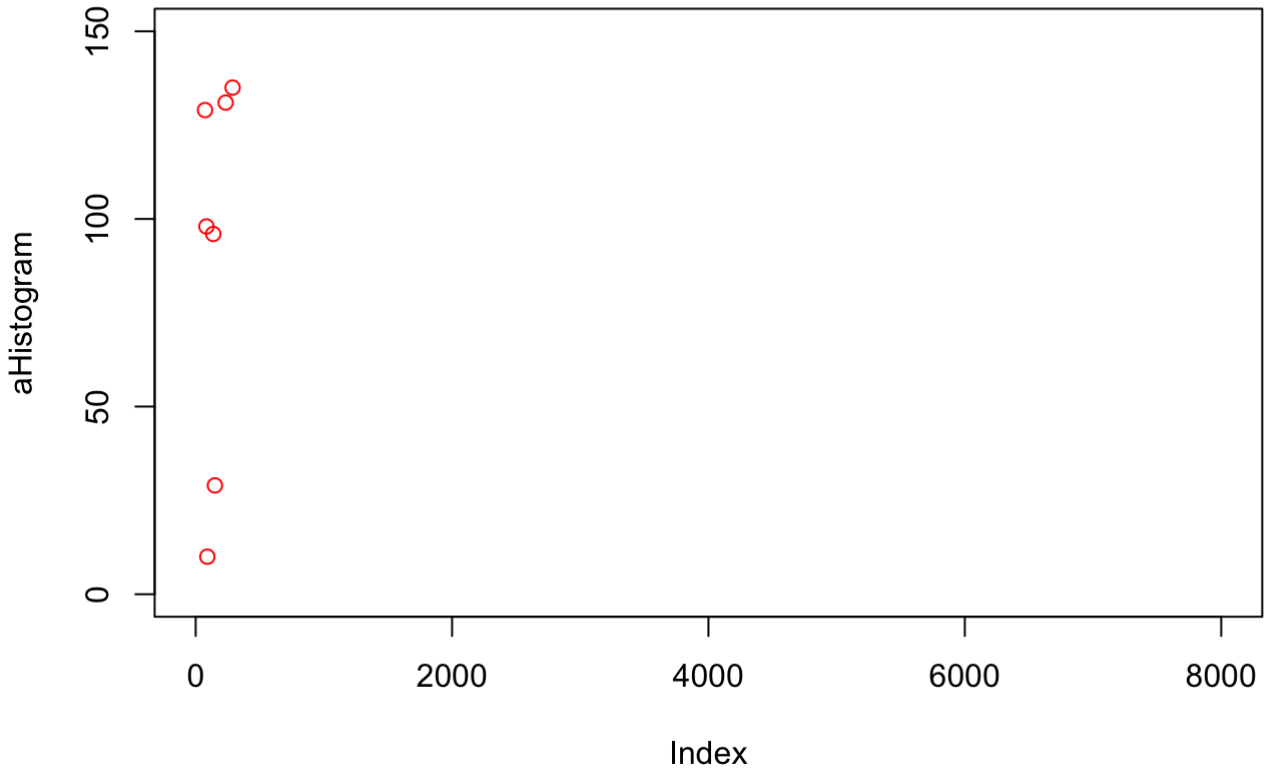
```
## Warning in plot.window(...): "add" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "add" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter
```

```
## Warning in box(...): "add" is not a graphical parameter
```

```
## Warning in title(...): "add" is not a graphical parameter
```




```
plot(bHistogram, col=bHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
```

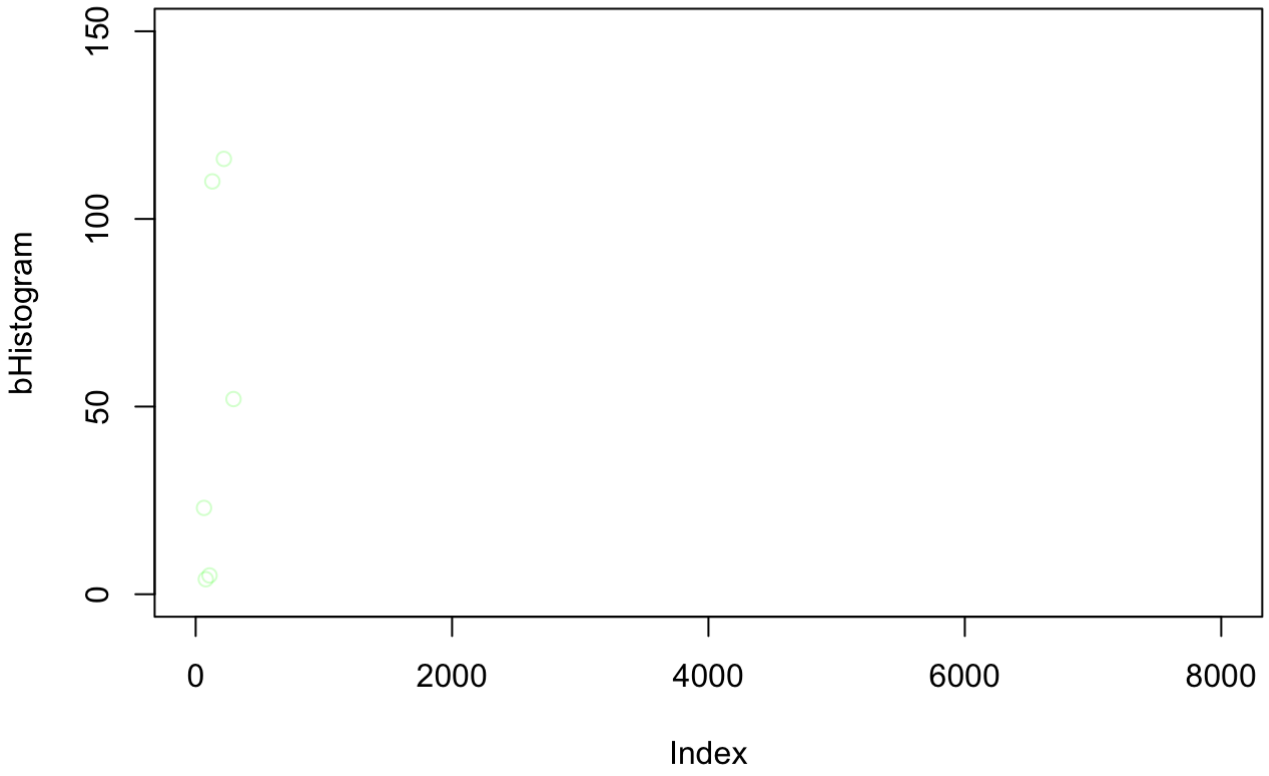
```
## Warning in plot.window(...): "add" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "add" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter
```

```
## Warning in box(...): "add" is not a graphical parameter
```

```
## Warning in title(...): "add" is not a graphical parameter
```




```
plot(cHistogram, col=cHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
```

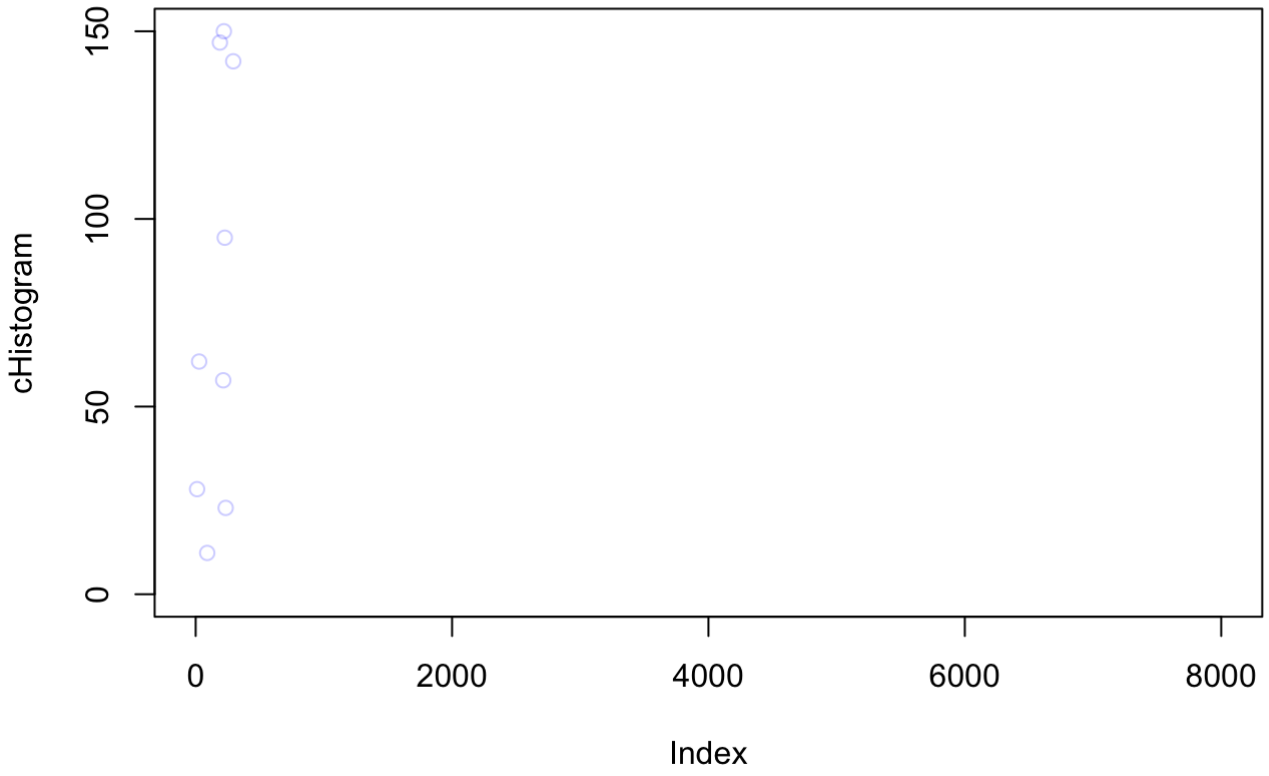
```
## Warning in plot.window(...): "add" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "add" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter
```

```
## Warning in box(...): "add" is not a graphical parameter
```

```
## Warning in title(...): "add" is not a graphical parameter
```




```
plot(dHistogram, col=dHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
```

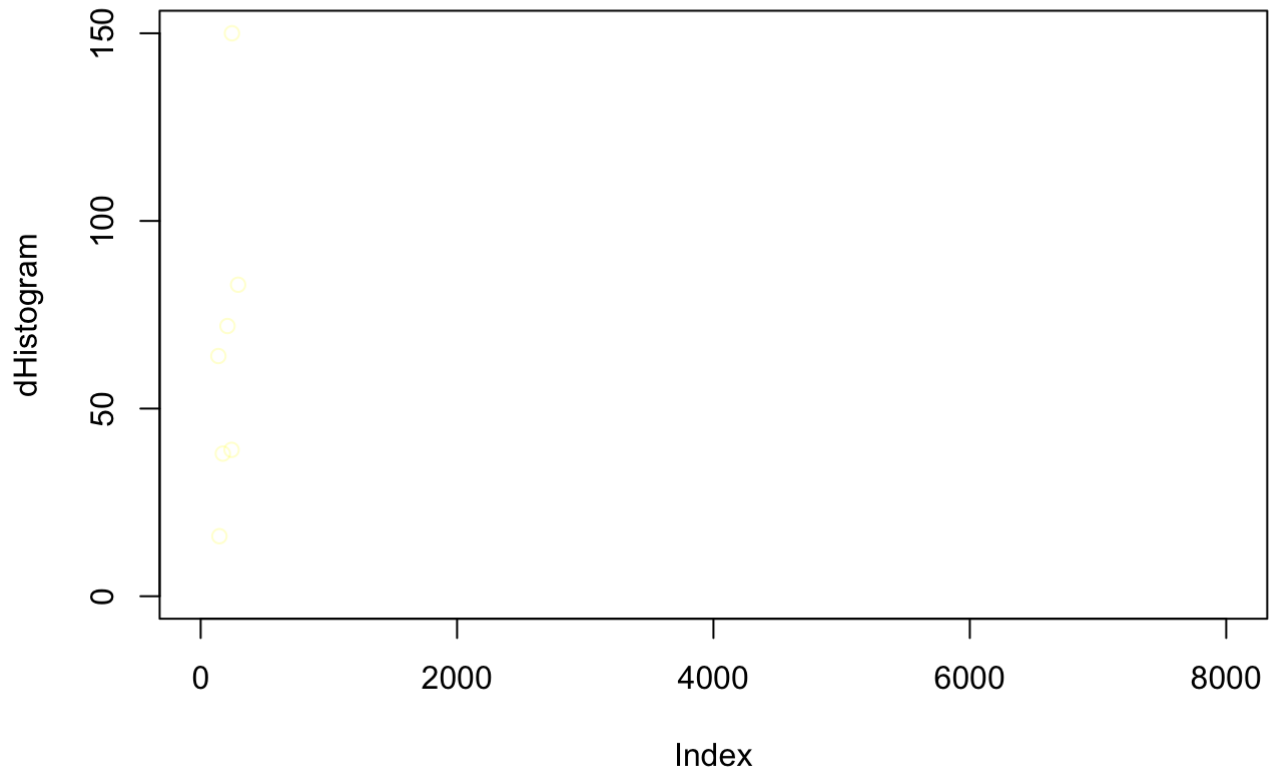
```
## Warning in plot.window(...): "add" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "add" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter
```

```
## Warning in box(...): "add" is not a graphical parameter
```

```
## Warning in title(...): "add" is not a graphical parameter
```




```
plot(eHistogram, col=eHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
```

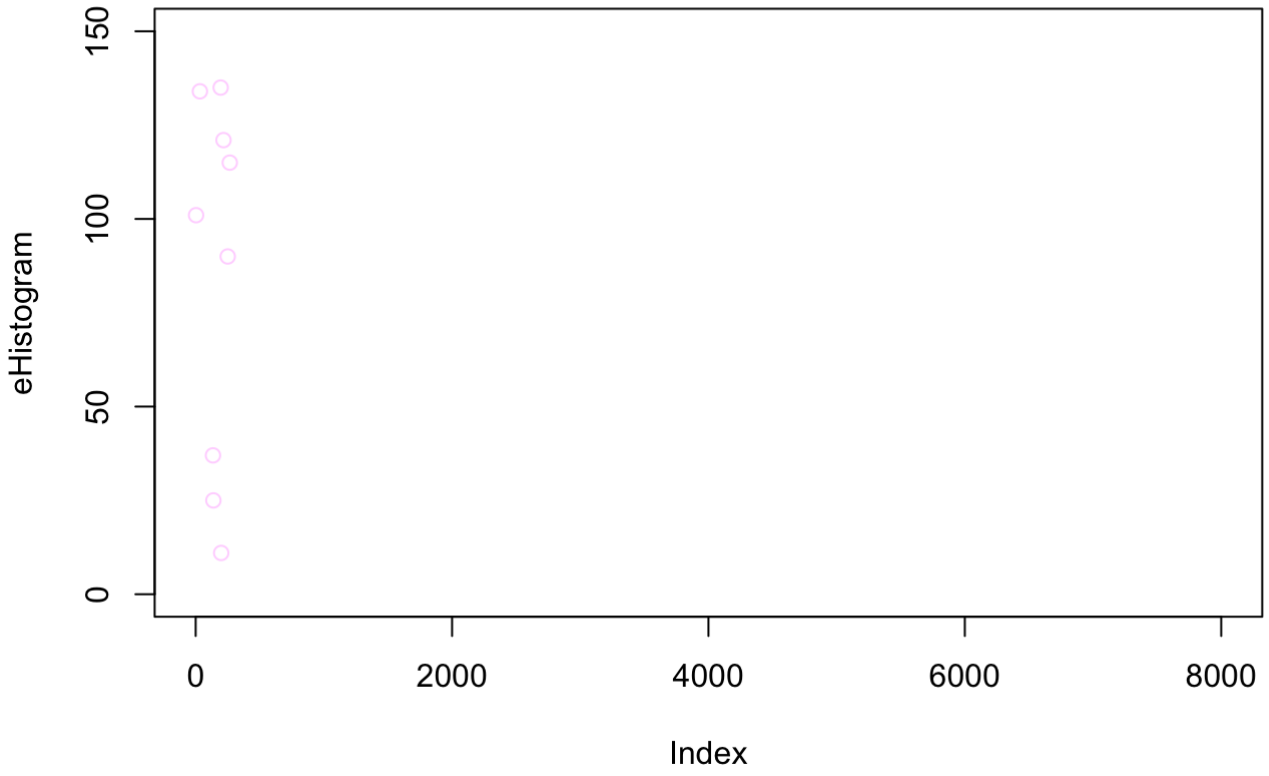
```
## Warning in plot.window(...): "add" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "add" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter
```

```
## Warning in box(...): "add" is not a graphical parameter
```

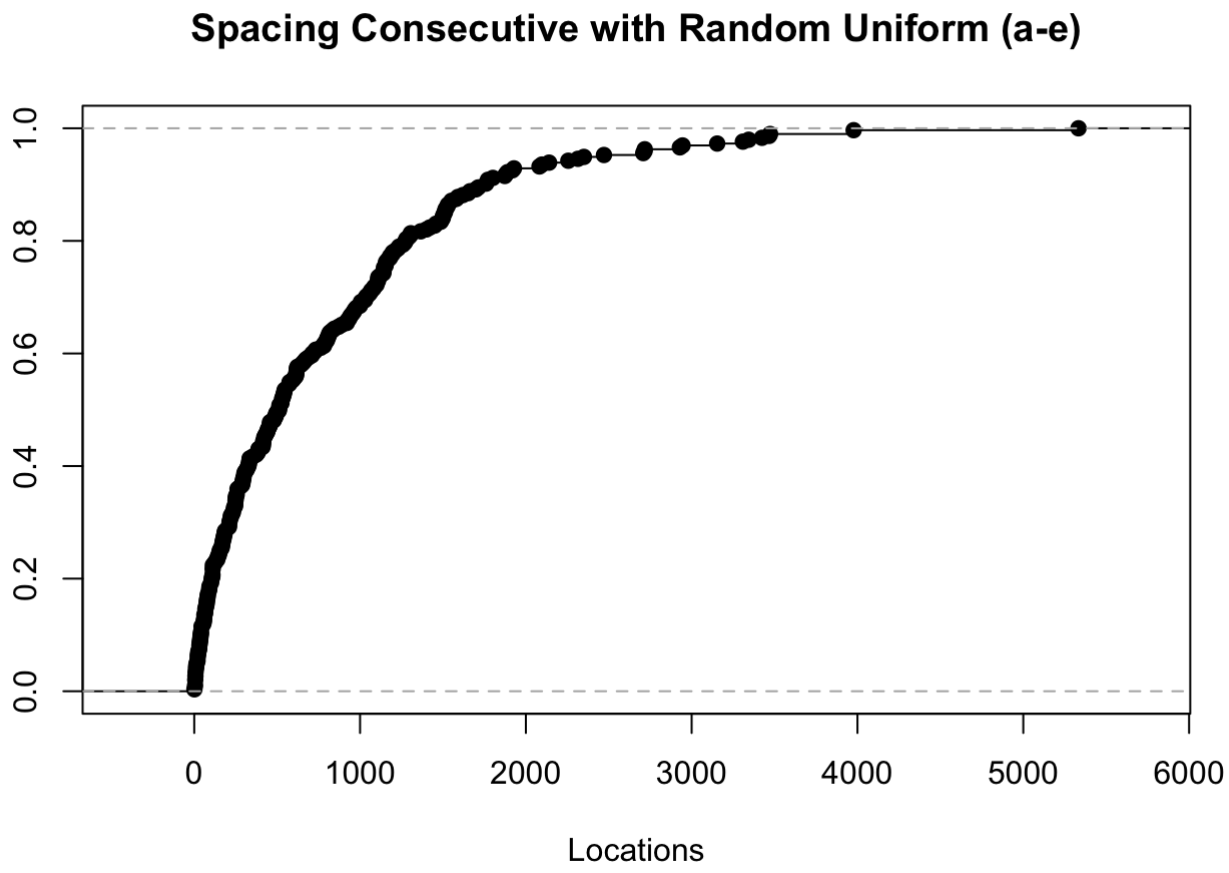
```
## Warning in title(...): "add" is not a graphical parameter
```



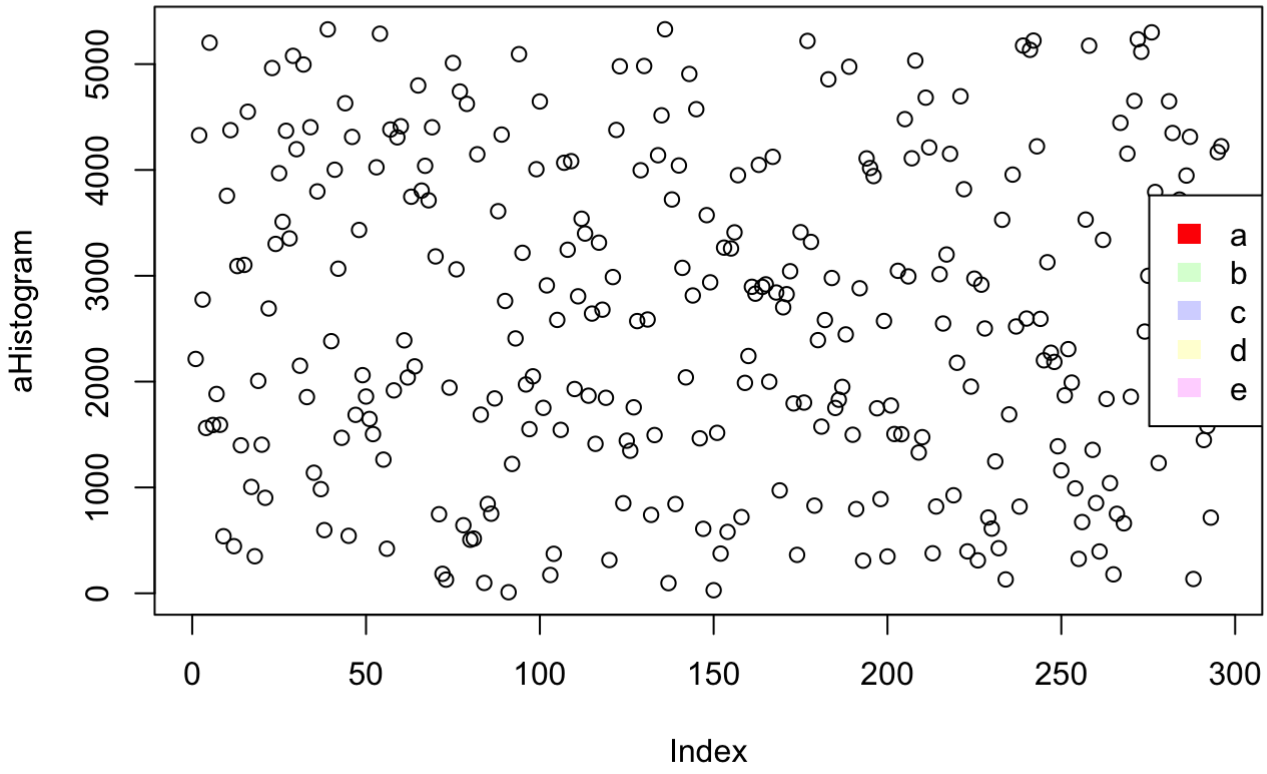

```
hist_1.ecdf = ecdf(space_1$location)
hist_1.ecdf
```

```
## Empirical CDF
## Call: ecdf(space_1$location)
##  x[1:258] =      1,      5,      6,  ...,   3977,   5333
```

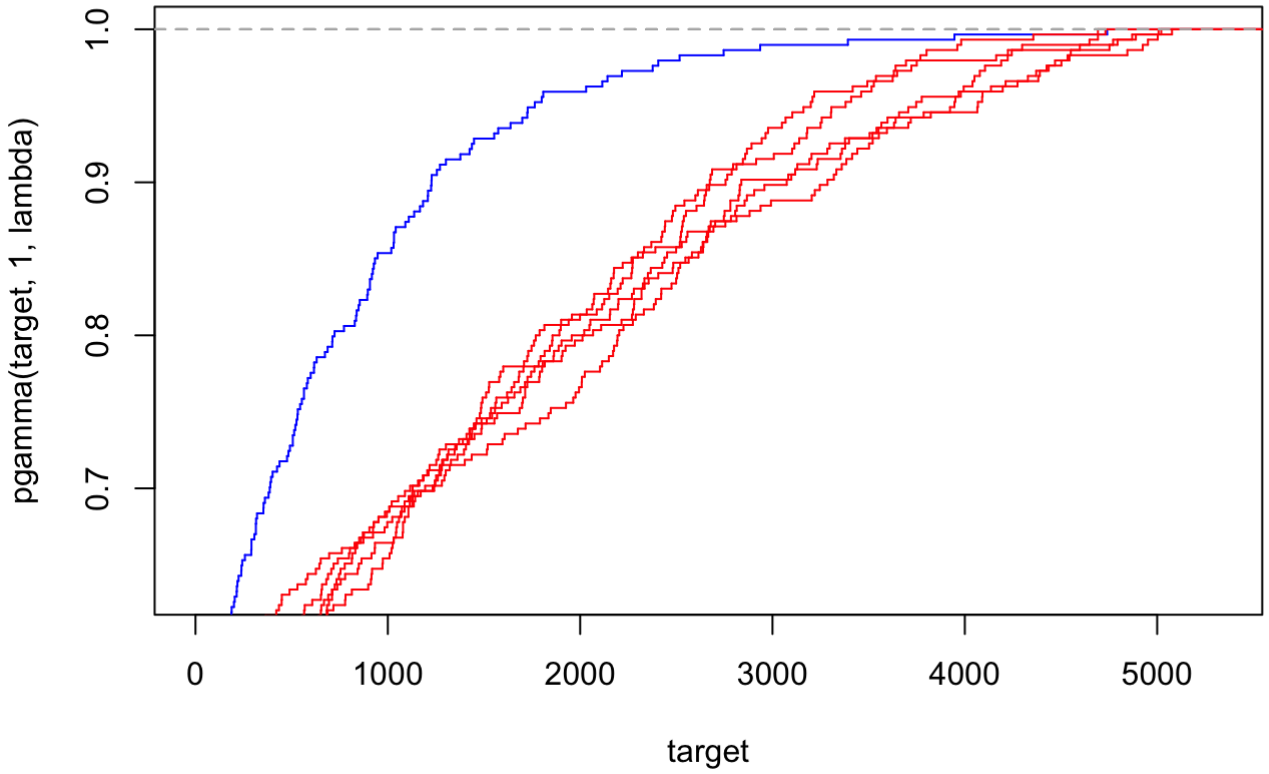
```
plot(hist_1.ecdf, xlab = 'Locations', ylab = '', main = 'Spacing Consecutive with Random
Uniform (a-e)')
```



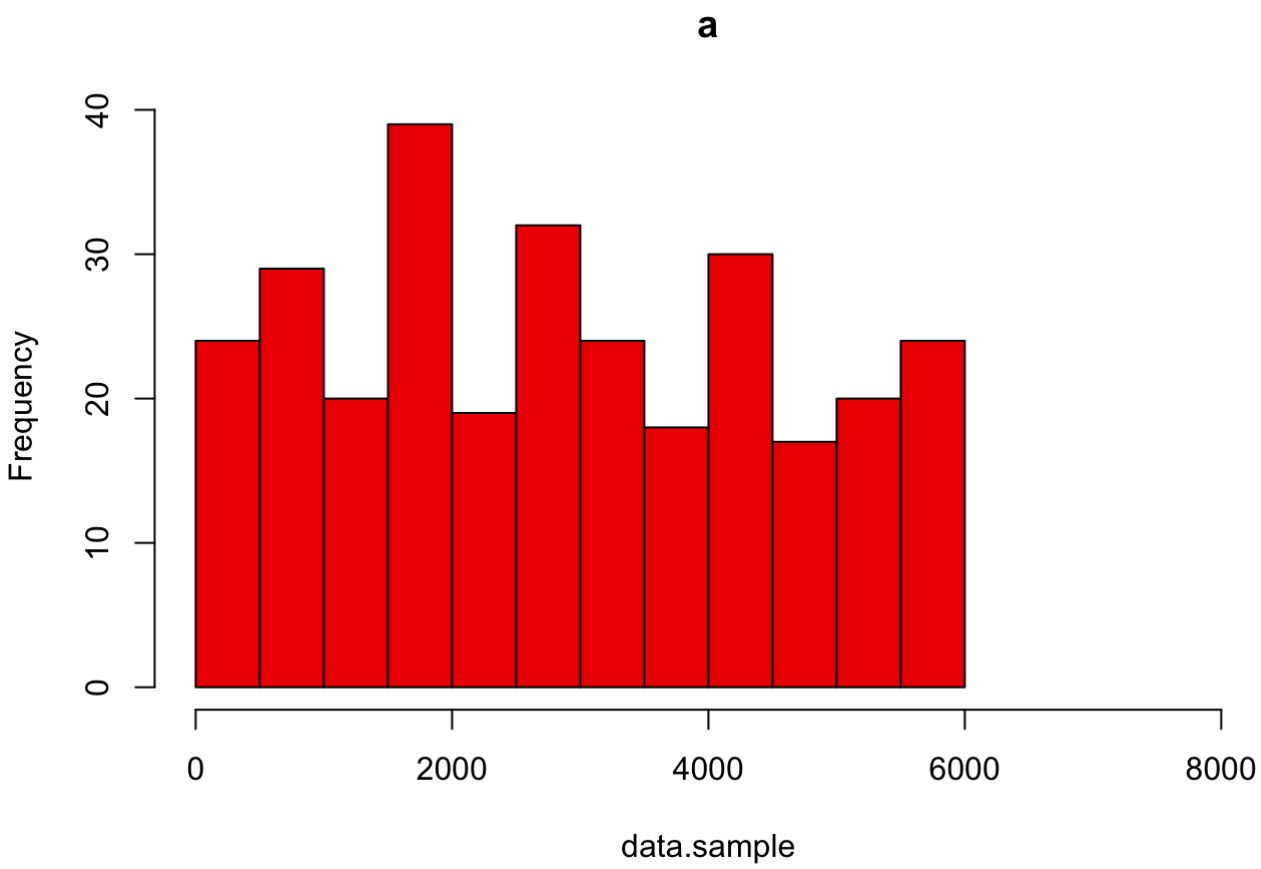

```
plot(aHistogram)


legend('right', c('a', 'b', 'c', 'd', 'e'), fill=c(aHistColor, bHistColor, cHistColor, d
HistColor, eHistColor), border=NA)
```



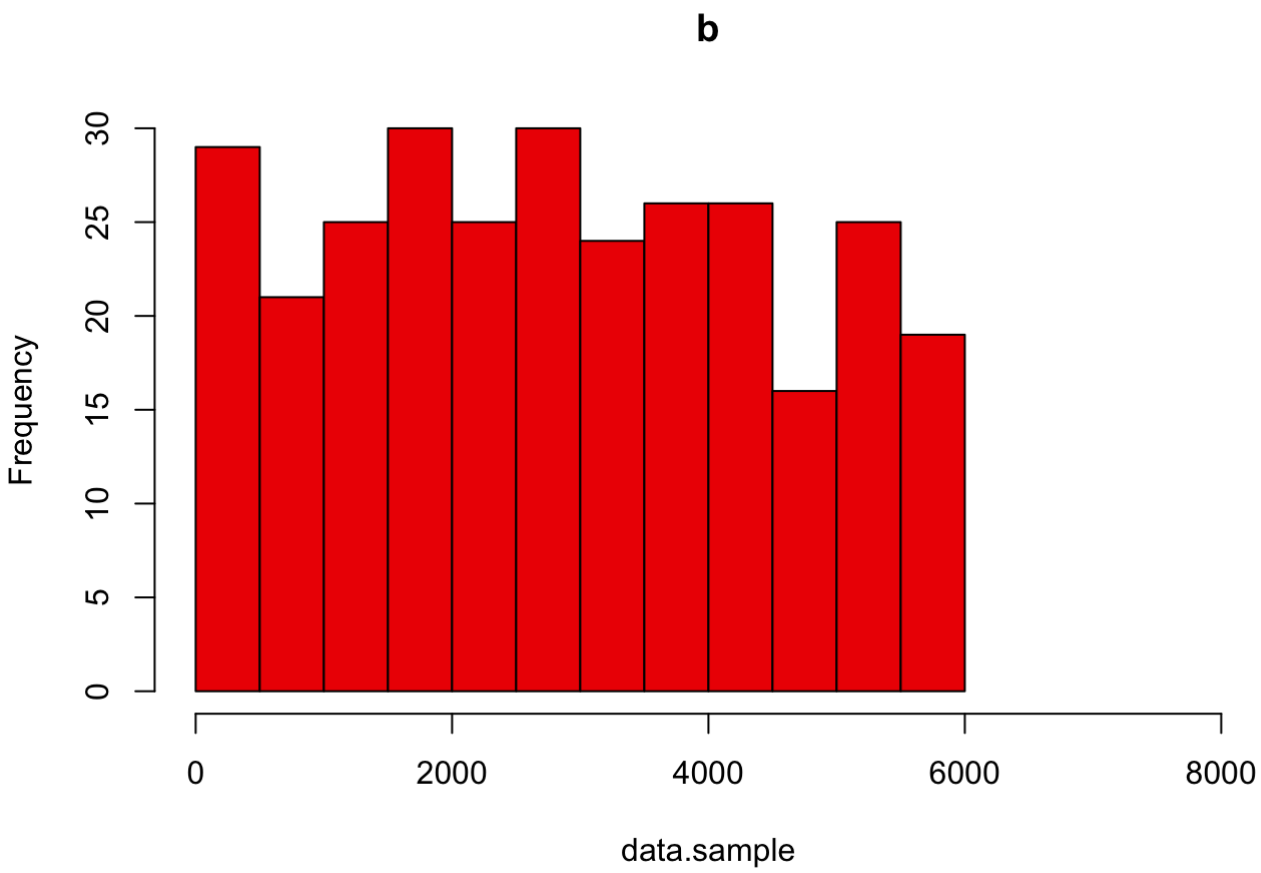

```
target = space_1$location
x = space_1$location
r1_x = aHistogram
r2_x = bHistogram
r3_x = cHistogram
r4_x = dHistogram
r5_x = eHistogram
lambda = 1
plot(target,pgamma(target,1,lambda),col='0')
plot(ecdf(diff(x)), do.points = F, verticals = T,add=T,col='blue')
plot(ecdf(diff(r1_x)), do.points = F, verticals = T,add=T,col='red')
plot(ecdf(diff(r2_x)), do.points = F, verticals = T,add=T,col='red')
plot(ecdf(diff(r3_x)), do.points = F, verticals = T,add=T,col='red')
plot(ecdf(diff(r4_x)), do.points = F, verticals = T,add=T,col='red')
plot(ecdf(diff(r5_x)), do.points = F, verticals = T,add=T,col='red')
```



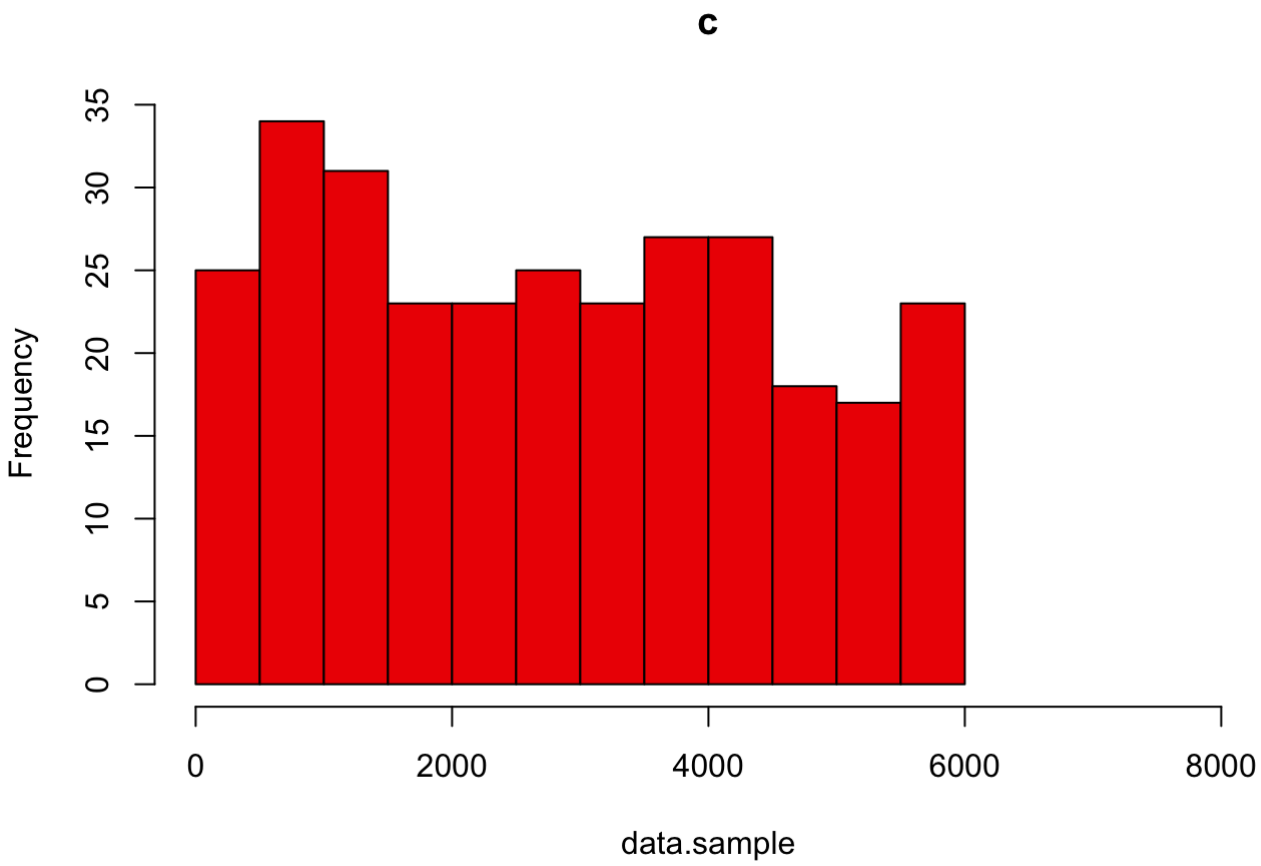

```
N <- max(space_2, na.rm=TRUE)
n <- 296
gene <- seq(min(space_2, na.rm=TRUE), N)

set.seed(200)
data.sample <- sample.int(N, size=n, replace=FALSE)
aHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="a")
```



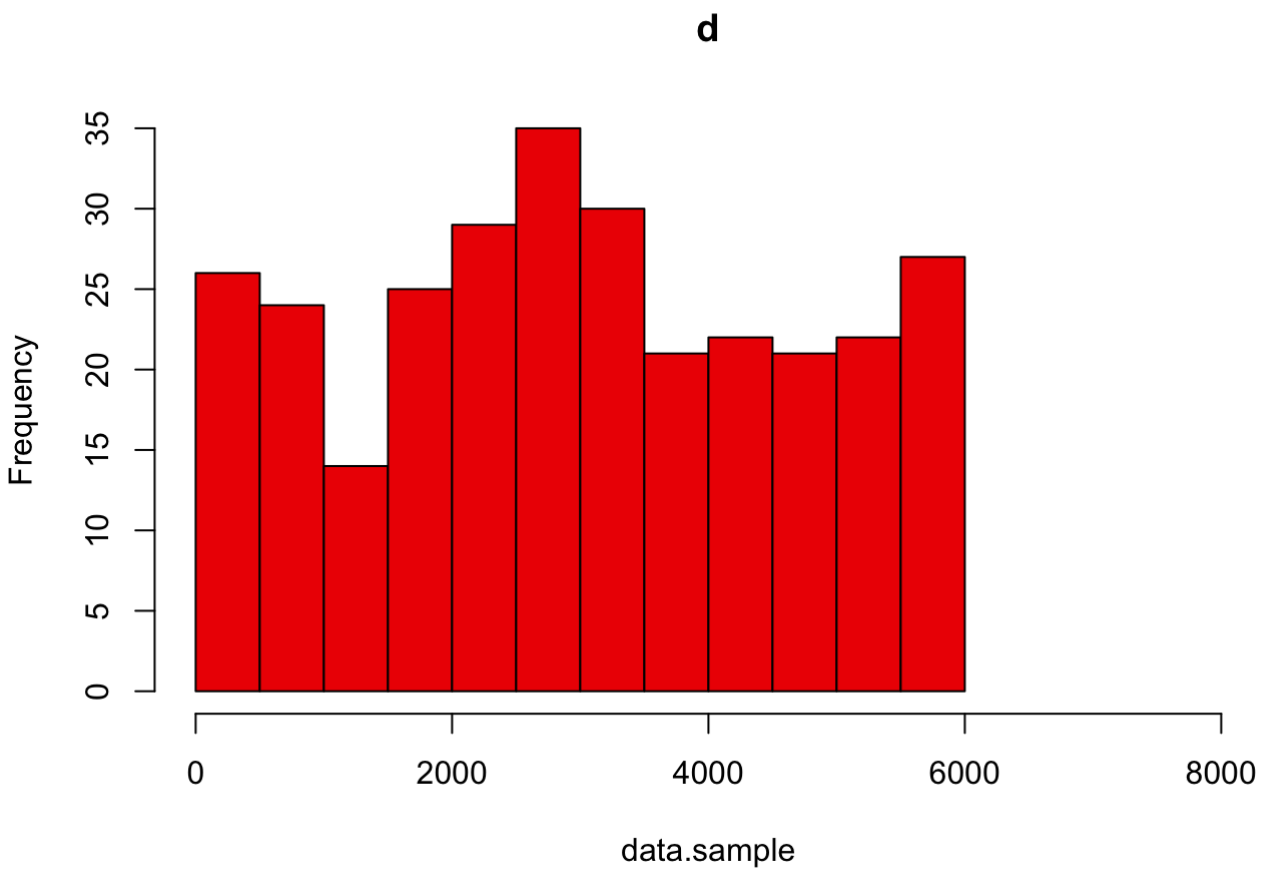

```
aHistogram = data.sample

data.sample <- sample.int(N, size=n, replace=FALSE)
bHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="b")
```



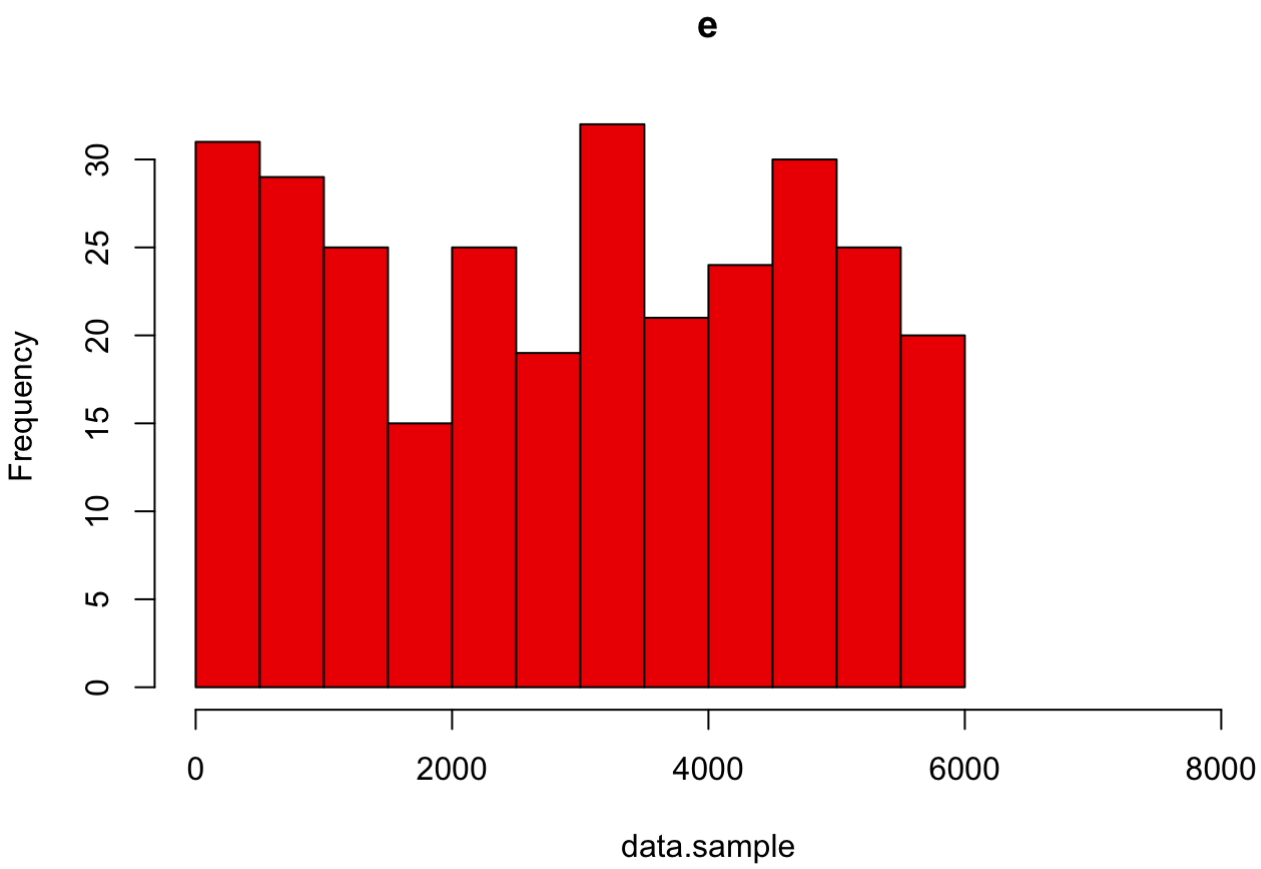

```
bHistogram = data.sample

data.sample <- sample.int(N, size=n, replace=FALSE)
cHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="c")
```



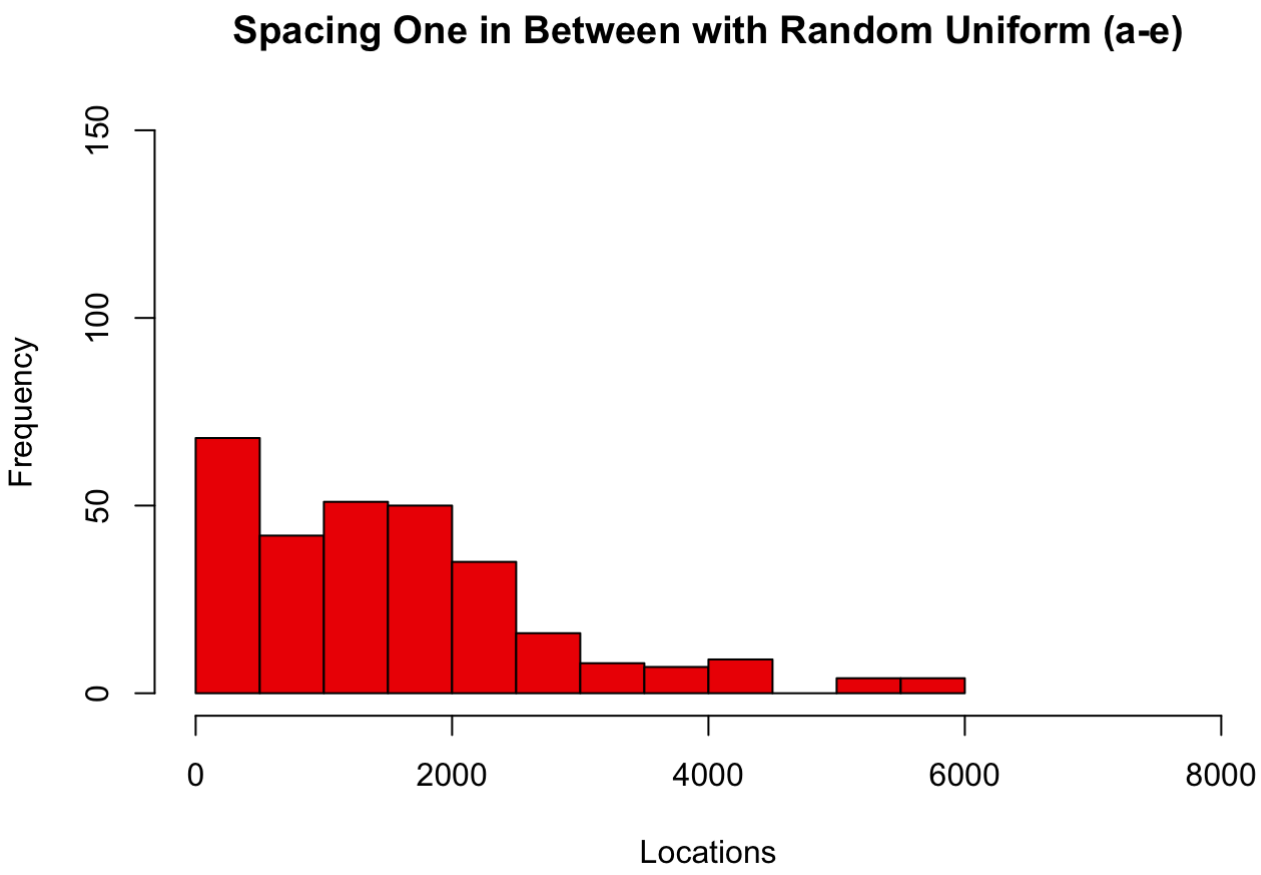

```
cHistogram = data.sample

data.sample <- sample.int(N, size=n, replace=FALSE)
dHistogram = hist(data.sample, xlim=c(0,8000), breaks=16,col='red2', main="d")
```




```
dHistogram = data.sample

data.sample <- sample.int(N, size=n, replace=FALSE)
eHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="e")
```




```
eHistogram = data.sample

aHistColor = rgb(1,0,0,0.2)
bHistColor = rgb(0,1,0,0.2)
cHistColor = rgb(0,0,1,0.2)
dHistColor = rgb(1,1,0,0.2)
eHistColor = rgb(1,0,1,0.2)

plot(hist_2, col='red2', ylim=c(0,150), xlim=c(0,8000), main="Spacing One in Between wit
h Random Uniform (a-e)", xlab="Locations")
```




```
plot(aHistogram, col=aHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
```

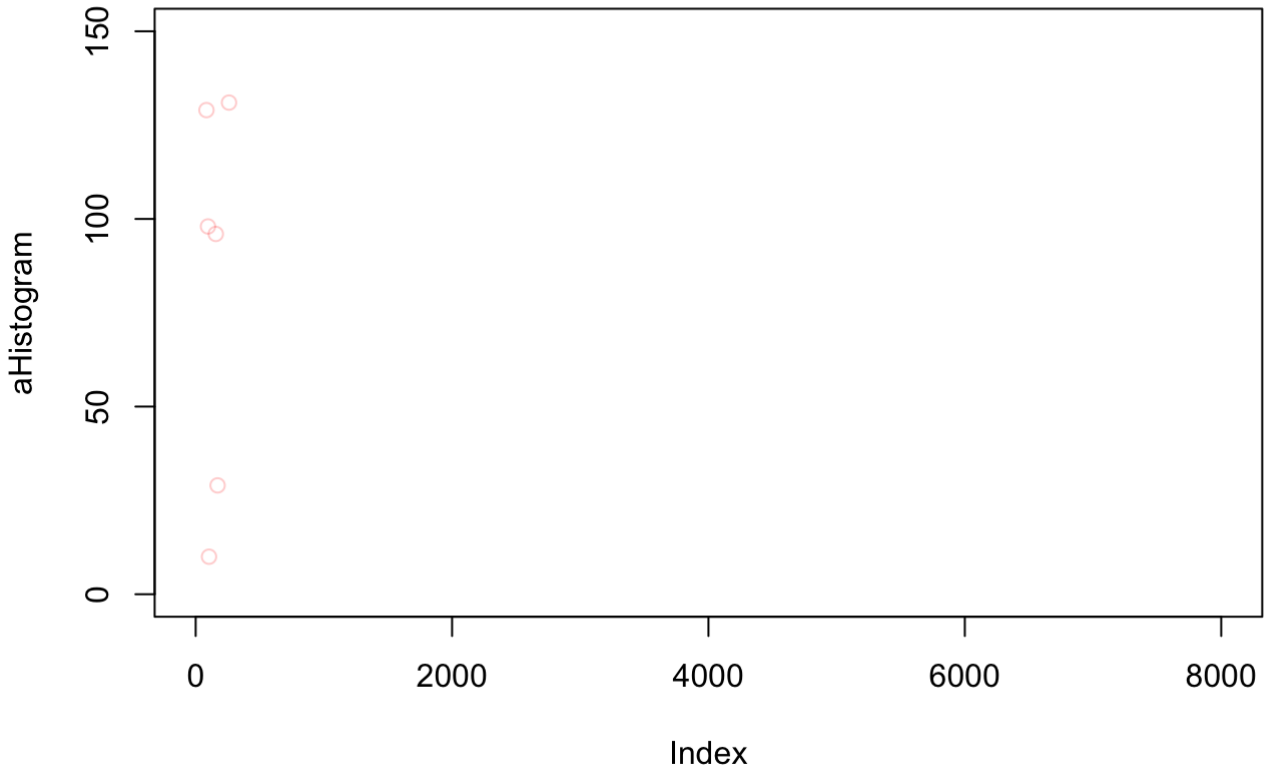
```
## Warning in plot.window(...): "add" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "add" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter
```

```
## Warning in box(...): "add" is not a graphical parameter
```

```
## Warning in title(...): "add" is not a graphical parameter
```




```
plot(bHistogram, col=bHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
```

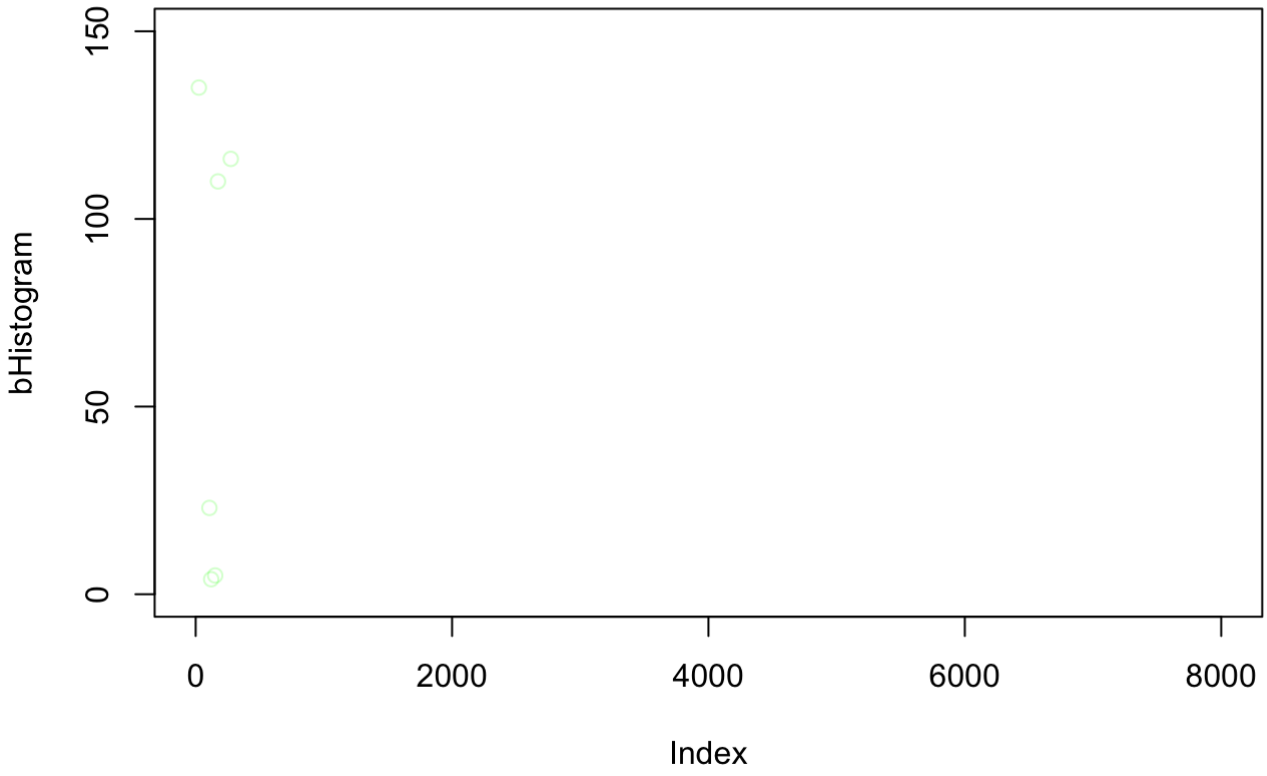
```
## Warning in plot.window(...): "add" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "add" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter
```

```
## Warning in box(...): "add" is not a graphical parameter
```

```
## Warning in title(...): "add" is not a graphical parameter
```




```
plot(cHistogram, col=cHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
```

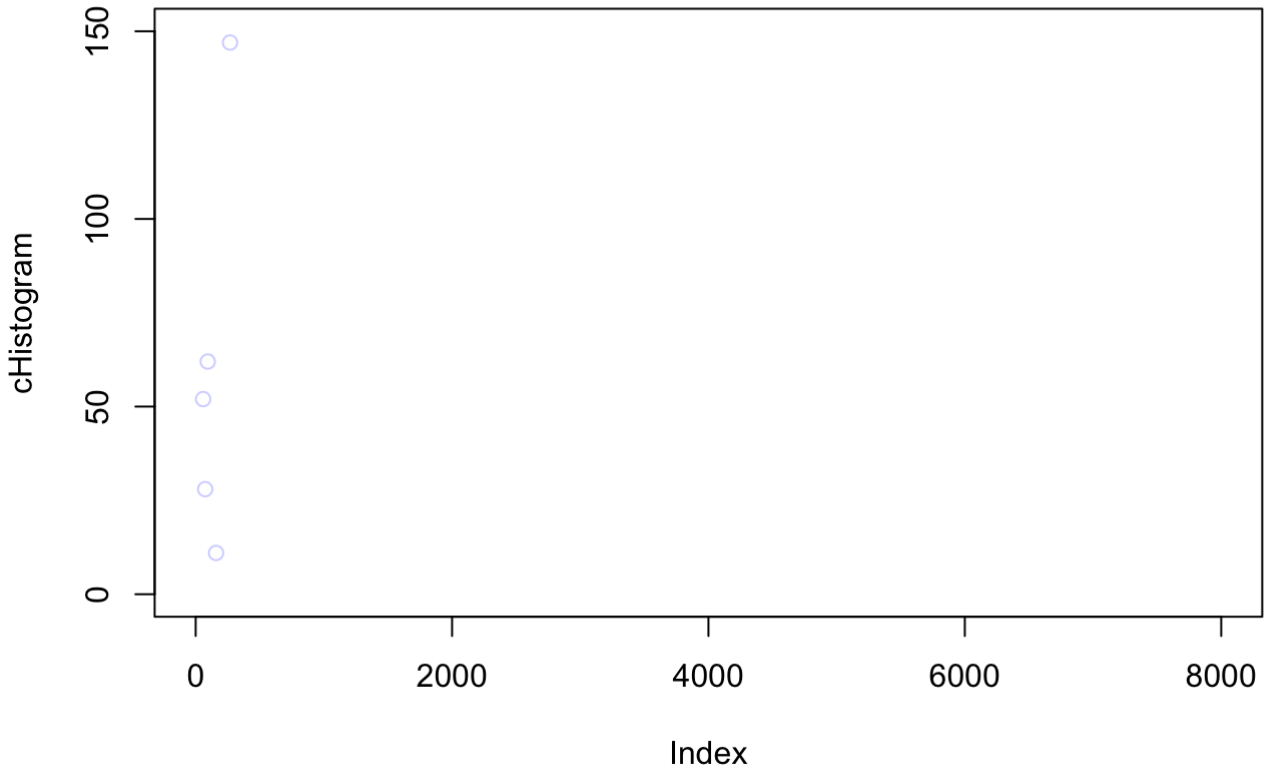
```
## Warning in plot.window(...): "add" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "add" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter
```

```
## Warning in box(...): "add" is not a graphical parameter
```

```
## Warning in title(...): "add" is not a graphical parameter
```




```
plot(dHistogram, col=dHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
```

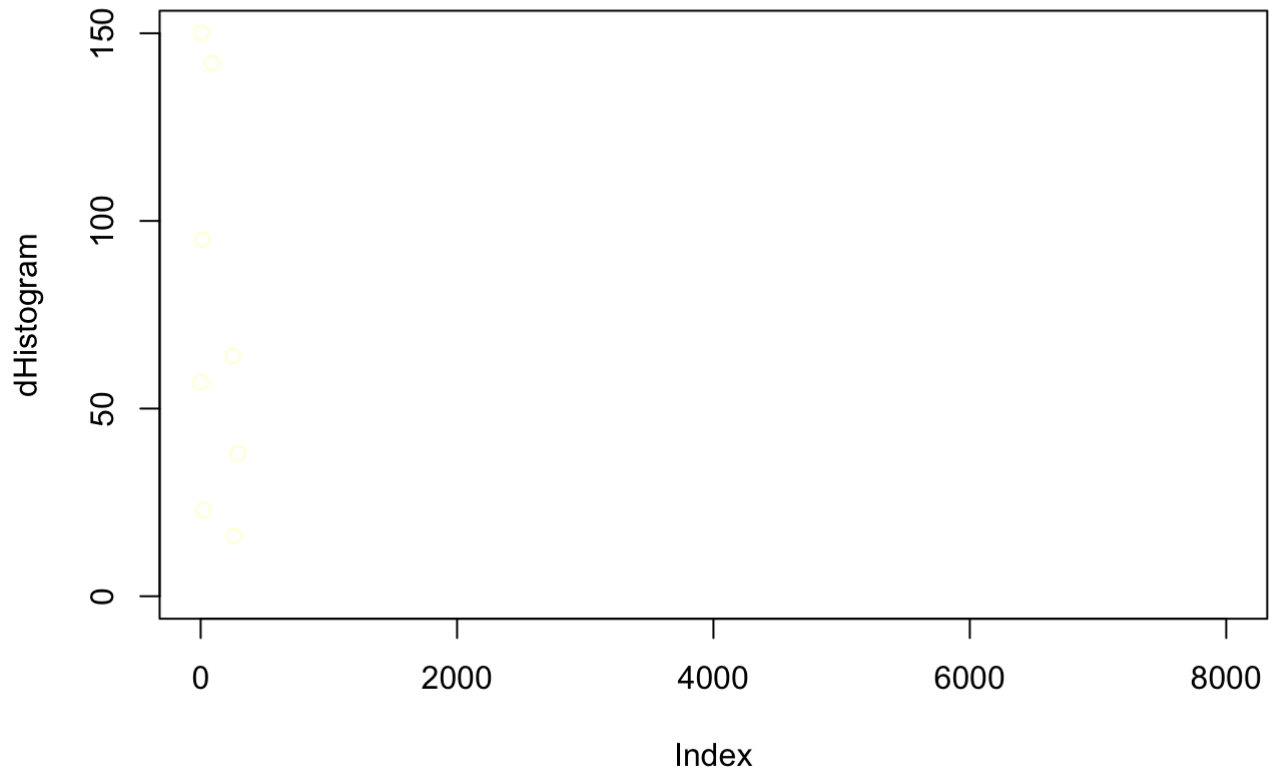
```
## Warning in plot.window(...): "add" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "add" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter
```

```
## Warning in box(...): "add" is not a graphical parameter
```

```
## Warning in title(...): "add" is not a graphical parameter
```




```
plot(eHistogram, col=eHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
```

```
## Warning in plot.window(...): "add" is not a graphical parameter
```

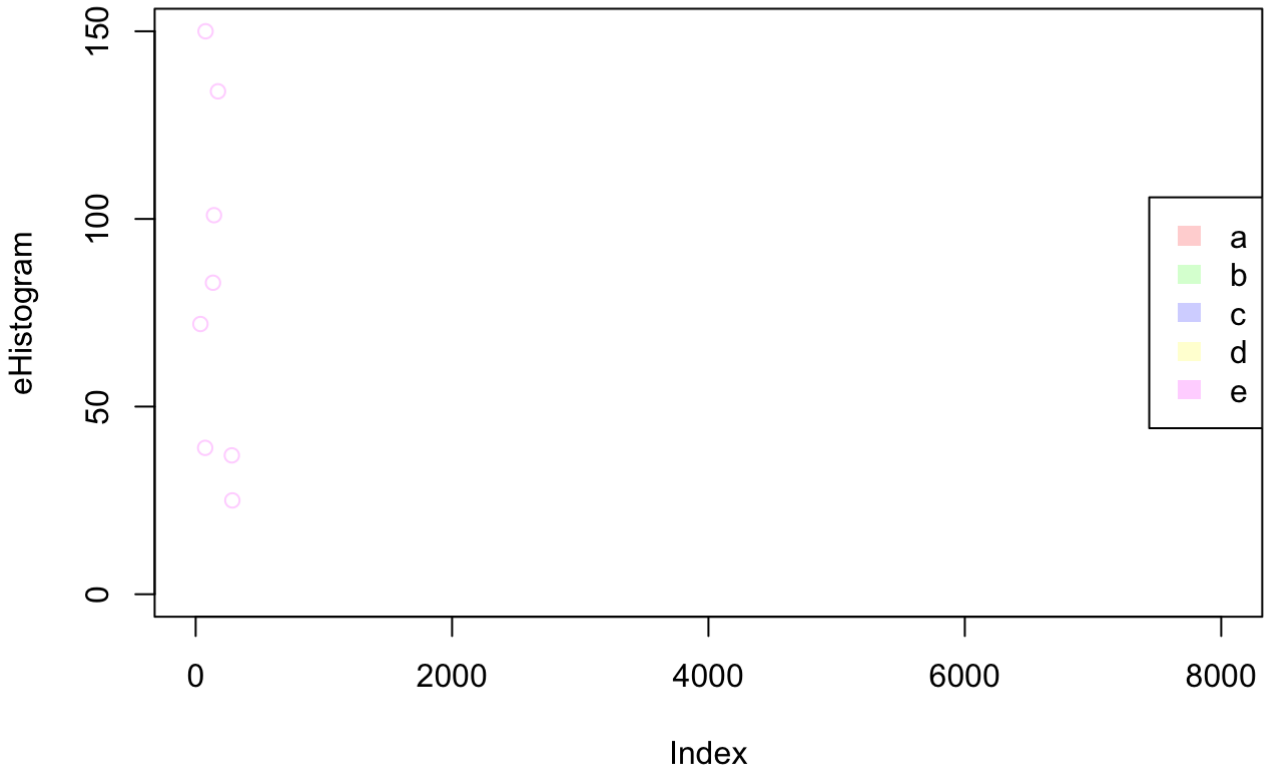
```
## Warning in plot.xy(xy, type, ...): "add" is not a graphical parameter
```

```
## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "add" is not a
## graphical parameter
```

```
## Warning in box(...): "add" is not a graphical parameter
```

```
## Warning in title(...): "add" is not a graphical parameter
```

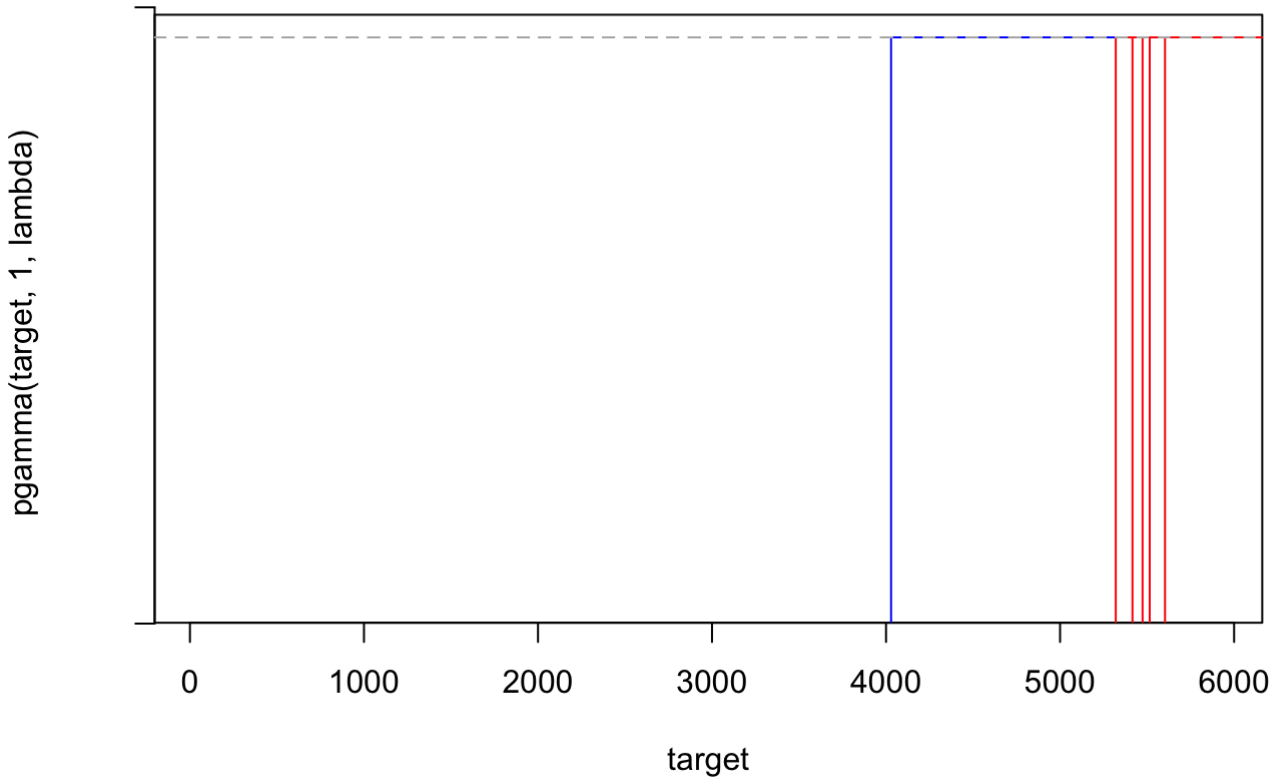
```
legend('right', c('a', 'b', 'c', 'd', 'e'), fill=c(aHistColor, bHistColor, cHistColor, d
HistColor, eHistColor), border=NA)
```



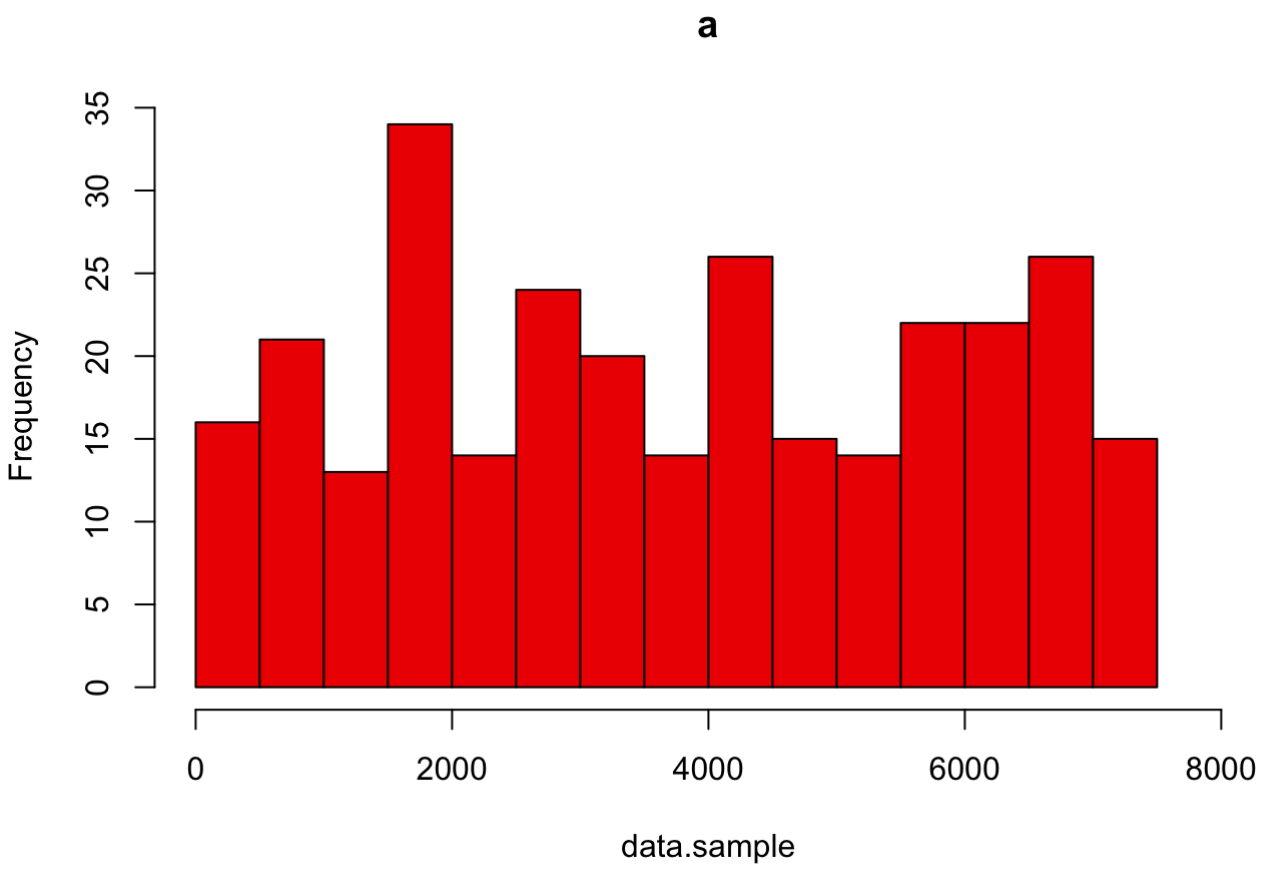

```
target = space_2
x = space_2
r1_x = aHistogram
r2_x = bHistogram
r3_x = cHistogram
r4_x = dHistogram
r5_x = eHistogram
lambda = 1
plot(target,pgamma(target,1,lambda),col='0')
```

```
## Warning in plot.window(...): relative range of values ( 0 * EPS) is small (axis
## 2)
```

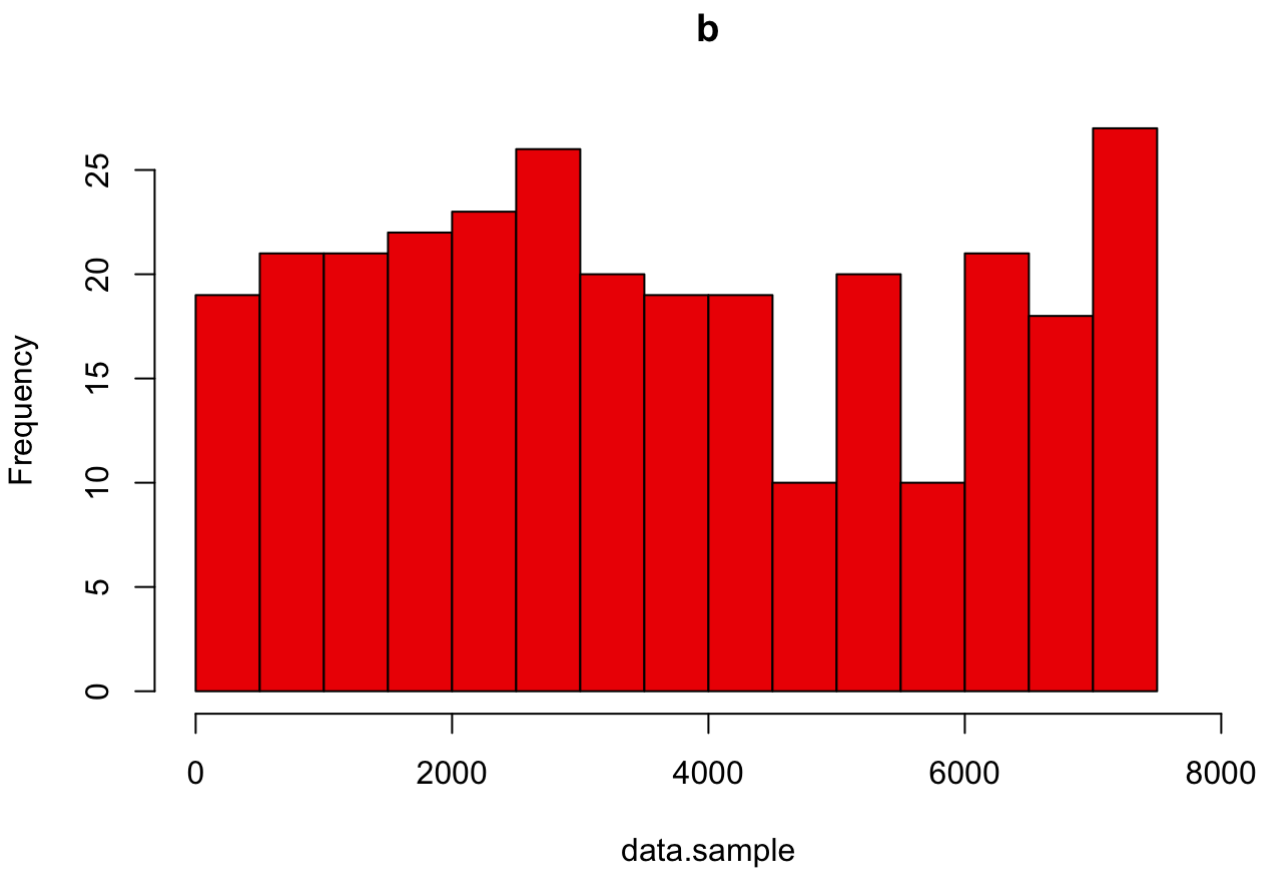
```
plot(ecdf(diff(x)), do.points = F, verticals = T,add=T,col='blue')
plot(ecdf(diff(r1_x)), do.points = F, verticals = T,add=T,col='red')
plot(ecdf(diff(r2_x)), do.points = F, verticals = T,add=T,col='red')
plot(ecdf(diff(r3_x)), do.points = F, verticals = T,add=T,col='red')
plot(ecdf(diff(r4_x)), do.points = F, verticals = T,add=T,col='red')
plot(ecdf(diff(r5_x)), do.points = F, verticals = T,add=T,col='red')
```



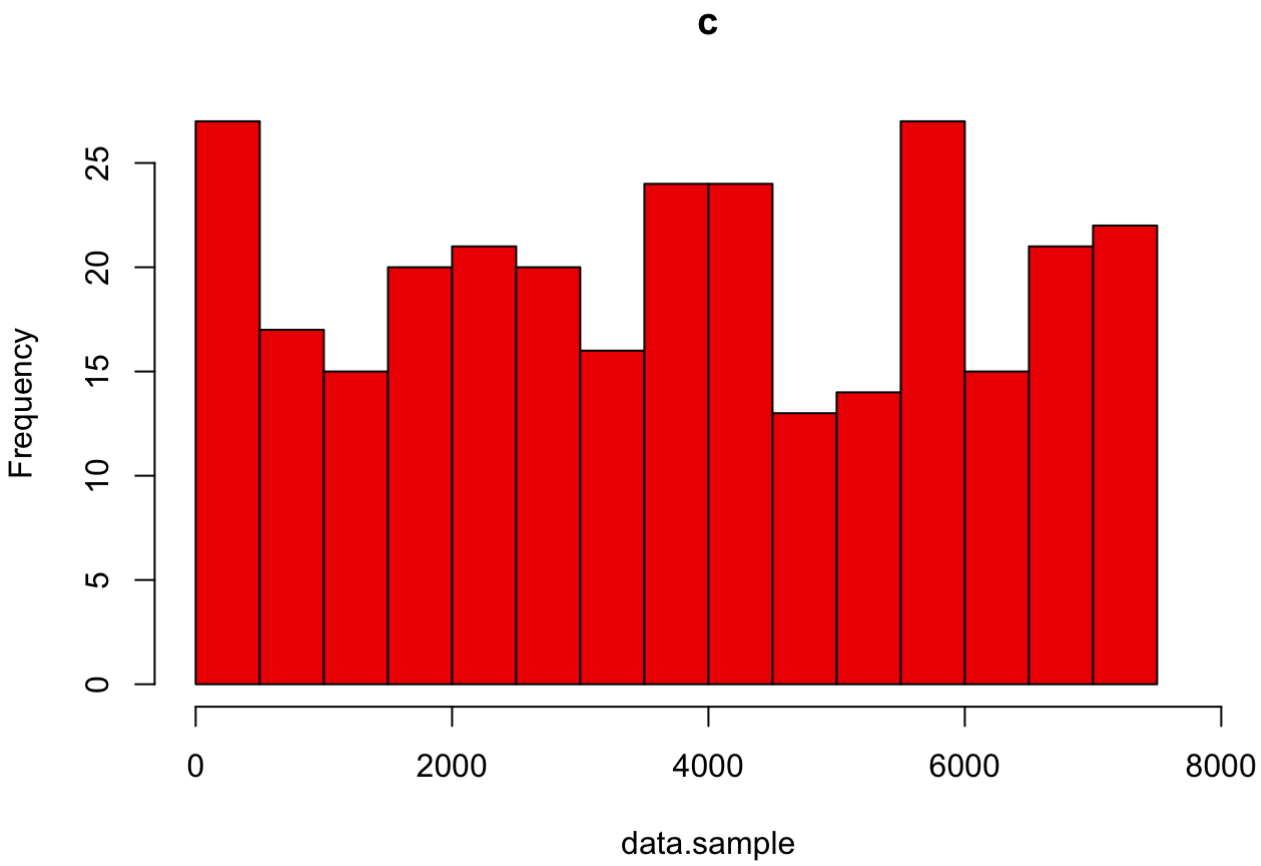

```
N <- max(space_3, na.rm=TRUE)
n <- 296
gene <- seq(min(space_3, na.rm=TRUE), N)

set.seed(200)
data.sample <- sample.int(N, size=n, replace=FALSE)
aHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="a")
```



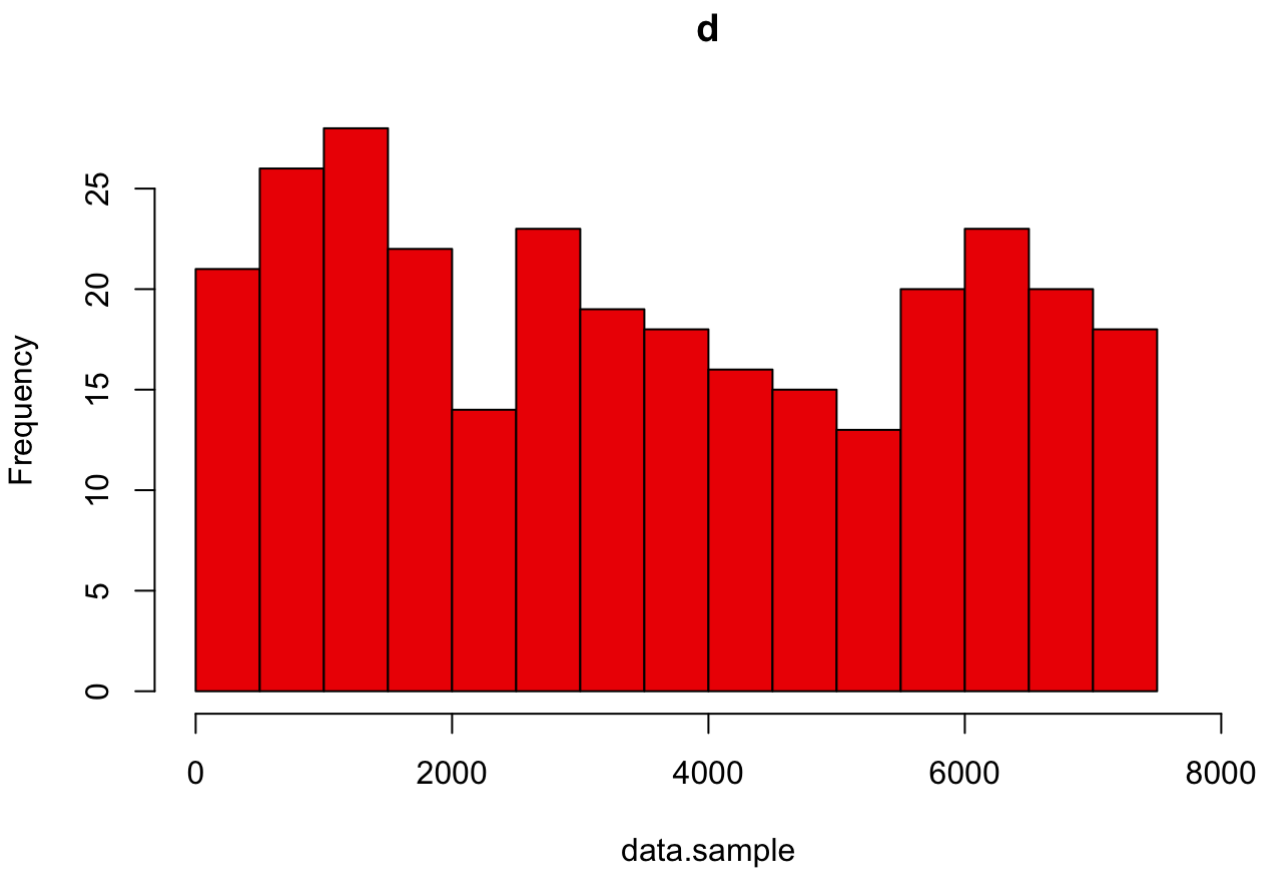

```
data.sample <- sample.int(N, size=n, replace=FALSE)
bHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="b")
```



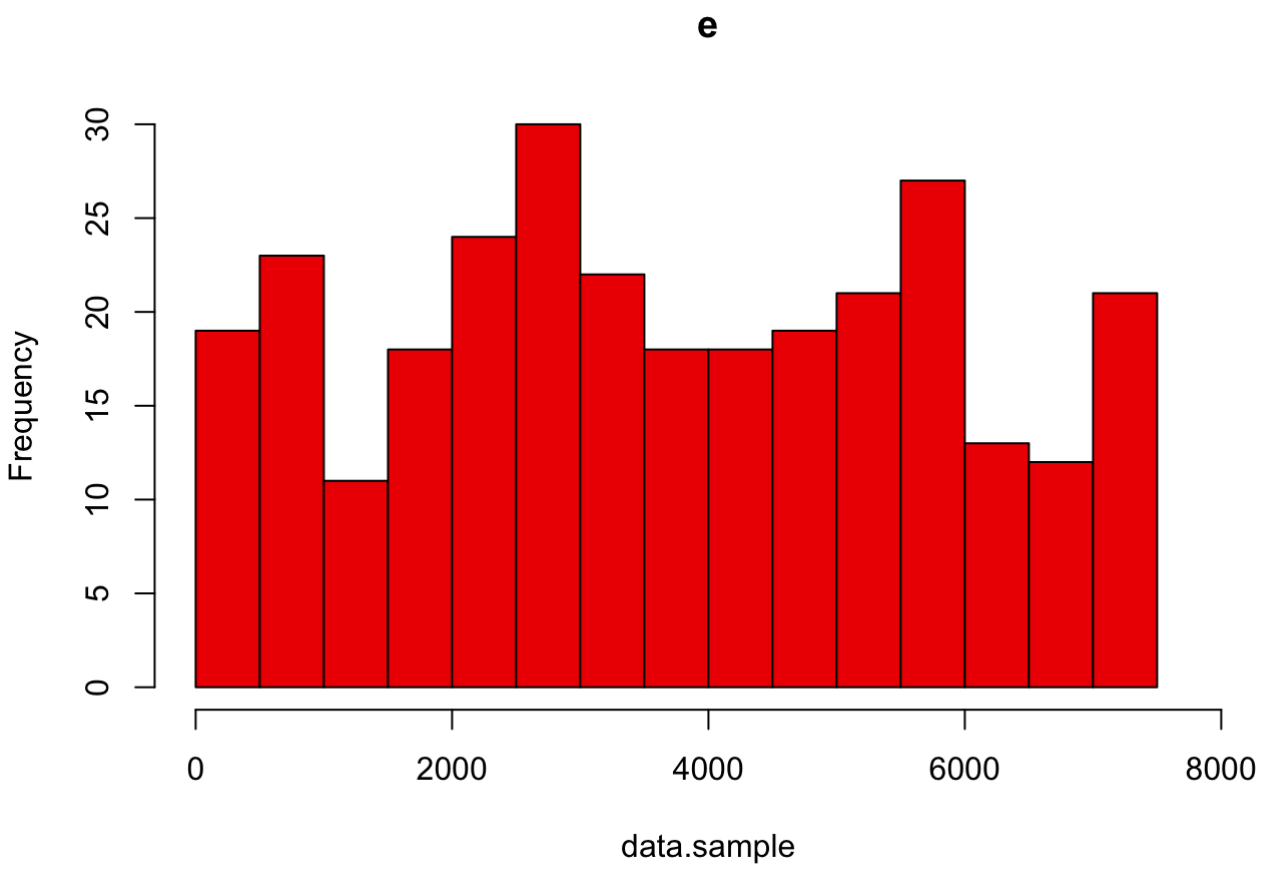

```
data.sample <- sample.int(N, size=n, replace=FALSE)
cHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="c")
```



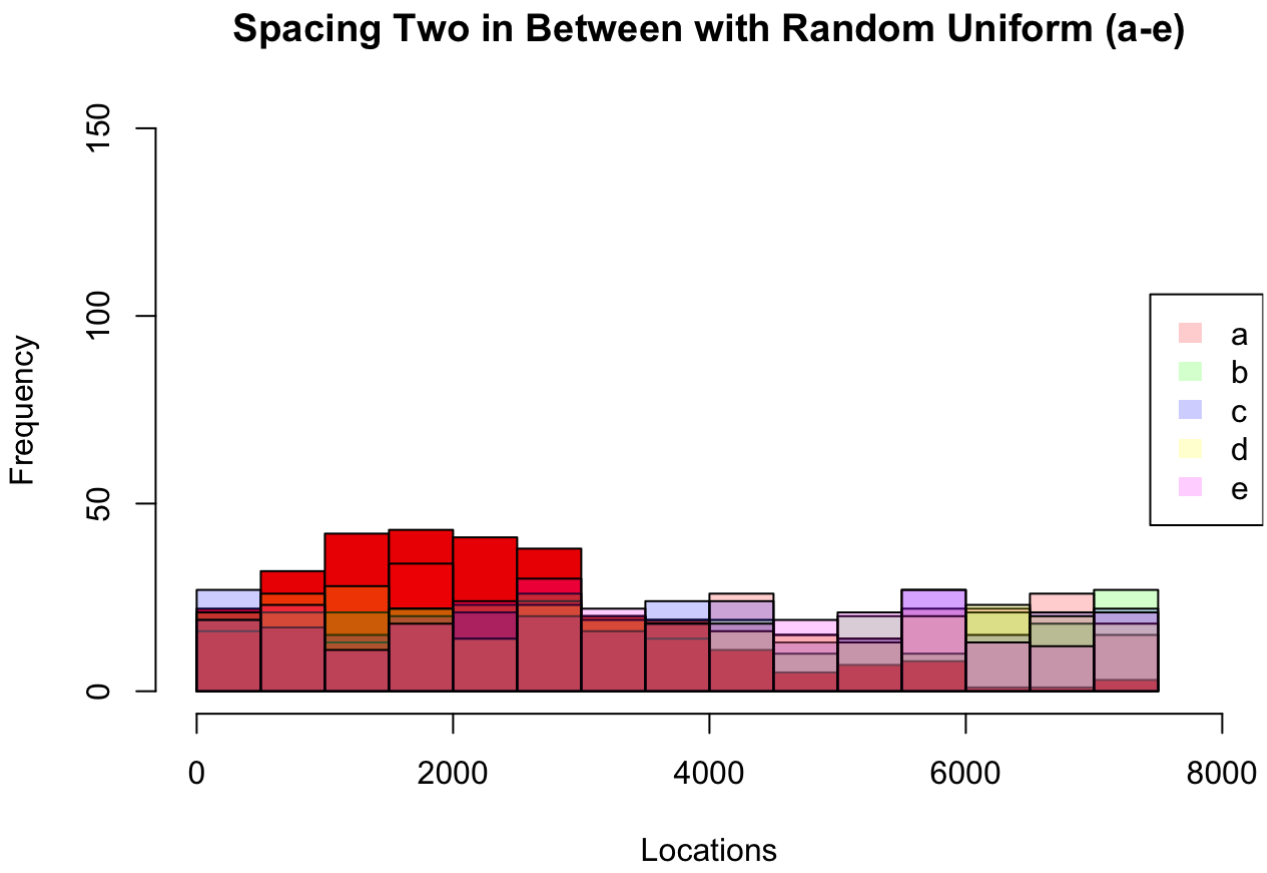

```
data.sample <- sample.int(N, size=n, replace=FALSE)
dHistogram = hist(data.sample, xlim=c(0,8000), breaks=16,col='red2', main="d")
```




```
data.sample <- sample.int(N, size=n, replace=FALSE)
eHistogram = hist(data.sample, xlim=c(0,8000), breaks=16, col='red2', main="e")
```




```
aHistColor = rgb(1,0,0,0.2)
bHistColor = rgb(0,1,0,0.2)
cHistColor = rgb(0,0,1,0.2)
dHistColor = rgb(1,1,0,0.2)
eHistColor = rgb(1,0,1,0.2)

plot(hist_3, col='red2', ylim=c(0,150), xlim=c(0,8000), main="Spacing Two in Between wit
h Random Uniform (a-e)", xlab="Locations")
plot(aHistogram, col=aHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
plot(bHistogram, col=bHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
plot(cHistogram, col=cHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
plot(dHistogram, col=dHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)
plot(eHistogram, col=eHistColor, ylim=c(0,150), xlim=c(0,8000), add=T)


legend('right', c('a', 'b', 'c', 'd', 'e'), fill=c(aHistColor, bHistColor, cHistColor, d
HistColor, eHistColor), border=NA)
```



2c)

Compare 2a and 2b

2d)

Identify the theoretical distributions of the spacings from the random uniform scatter (we should identify 3 theoretical distributions, each for a different spacing)

2e)

Overlay the identified theoretical distributions as cdf on the plots in 2). (5x3 cdf?)

If you are plotting histograms, a density is more appropriate. If you are plotting cdfs, a cdf is more appropriate. Yes you are intended to overlay your densities/cdfs in their appropriate plots but again, try to be as concise as possible when visualizing.

3.

Counts: Here the goal is to use graphical and formal statistical methods to examine the counts of palindromes in regions of the DNA. Be sure to do this for a few different (but reasonable) interval lengths. The graphical displays should compare the distribution of counts to random uniform scatter (this will be analogous to the locations and spacings sections). Using results from lecture, you can organize a formal statistical test to further examine your distributions of counts.

```
regionsplit <- function(n.region, gene, site){
  count.int <- table(cut(site, breaks = seq(1, length(gene), length.out=n.region+1), inc
lude.lowest=TRUE))
  count.vector <- as.vector(count.int)
  count.tab <- table(factor(count.vector,levels=0:max(count.vector)))
  return (count.tab)
}
```

```
n.region <- 50
N <- max(data, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data$location)
O2.tab=regionsplit(n.region, gene, data_sample)
O3.tab=regionsplit(n.region, gene, data_unif)
O1.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
##  1  1  1  6  6  7  6 13  3  3  1  1  0  0  0  0  1
```

```
O2.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
##  3  1  6  4  4  5  6 10  1  3  3  2  0  0  0  1  0  0  0  1
```

```
O3.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  0  0  0  0  0  0  0  0  0  0  1  0  0  1  2  5  1  2  7  4  5  5  6  3  3  1
## 26 27 28 29 30 31 32 33 34
##  0  0  3  0  0  0  0  0  1
```

```
trunc=9
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O2=as.vector(O2.tab)
O3=as.vector(O3.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
O2.trunc=c(O2[1:trunc],sum(O2[-(1:trunc)]))
O3.trunc=c(O3[1:trunc],sum(O3[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 5.92
```

```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab2=data.frame(levels=lvls,Observed=O2.trunc,Expected=E)
tab3=data.frame(levels=lvls,Observed=O3.trunc,Expected=E)
tab1
```

```
##    levels Observed  Expected
## 1       0        1 0.1342600
## 2       1        1 0.7948193
## 3       2        1 2.3526650
## 4       3        6 4.6425922
## 5       4        6 6.8710365
## 6       5        7 8.1353072
## 7       6        6 8.0268365
## 8       7       13 6.7884103
## 9       8        3 5.0234236
## 10    >=9        6 7.2306494
```

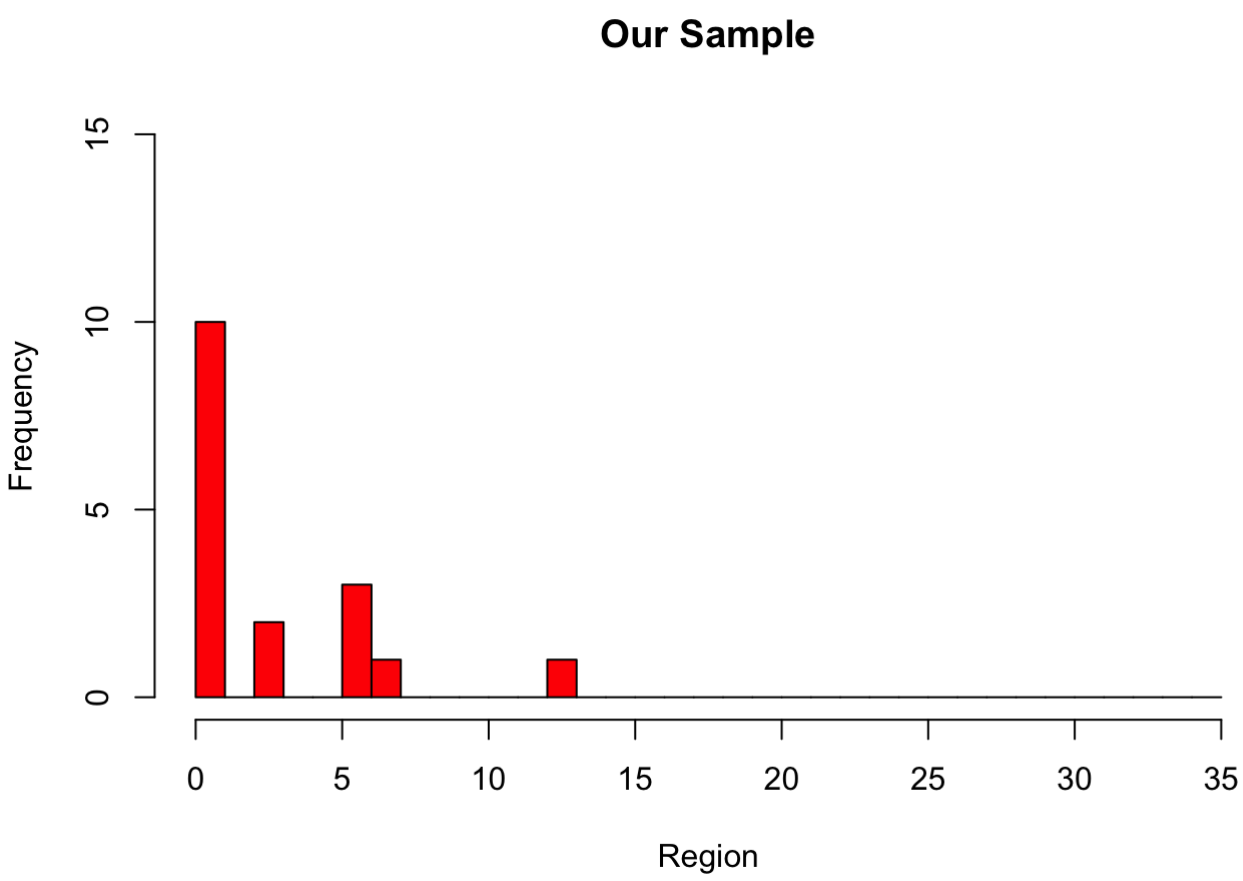
```
tab2
```

```
##    levels Observed  Expected
## 1       0        3 0.1342600
## 2       1        1 0.7948193
## 3       2        6 2.3526650
## 4       3        4 4.6425922
## 5       4        4 6.8710365
## 6       5        5 8.1353072
## 7       6        6 8.0268365
## 8       7       10 6.7884103
## 9       8        1 5.0234236
## 10    >=9       10 7.2306494
```

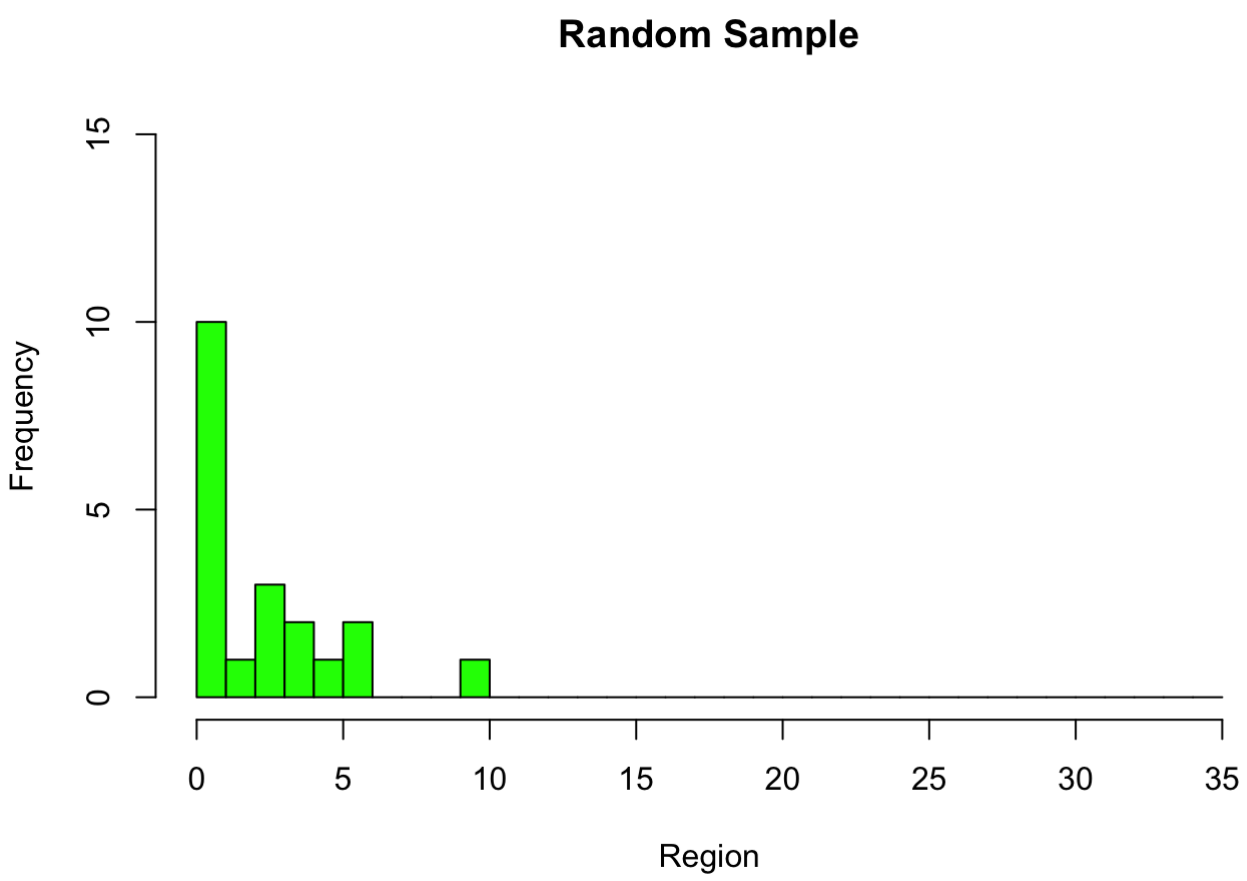
```
tab3
```

```
##    levels Observed  Expected
## 1       0        0 0.1342600
## 2       1        0 0.7948193
## 3       2        0 2.3526650
## 4       3        0 4.6425922
## 5       4        0 6.8710365
## 6       5        0 8.1353072
## 7       6        0 8.0268365
## 8       7        0 6.7884103
## 9       8        0 5.0234236
## 10    >=9       50 7.2306494
```

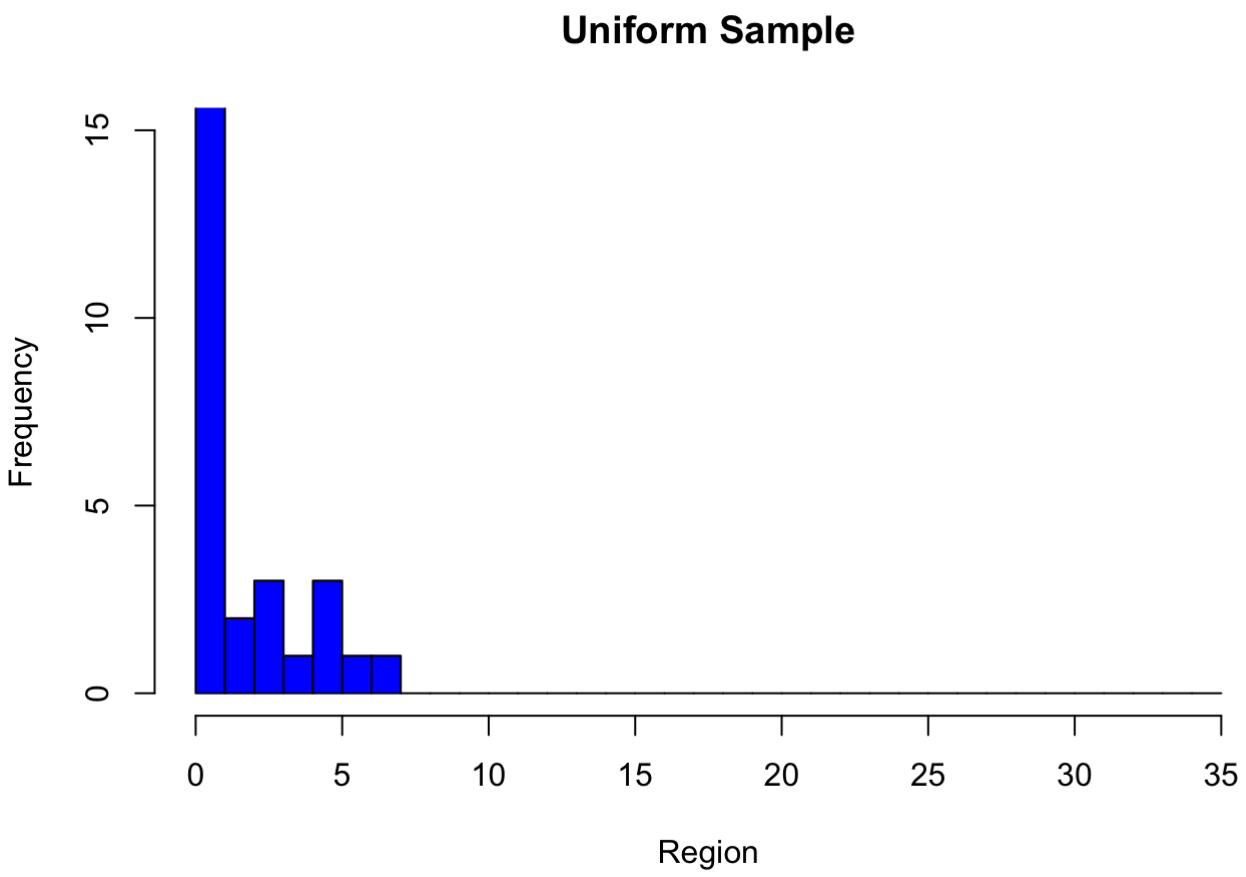
```
hist(O1,ylim=c(0,15), xlim=c(0,35),breaks=seq(0,35, by=1),col="red",xlab="Region",ylab=
"Frequency",main="Our Sample")
```




```
hist(O2,ylim=c(0,15), xlim=c(0,35),breaks=seq(0,35, by=1),col="green",xlab="Region",ylab
="Frequency",main="Random Sample")
```




```
hist(O3,ylim=c(0,15), xlim=c(0,35),breaks=seq(0,35, by=1),col="blue",xlab="Region",ylab=
"Frequency",main="Uniform Sample")
```




```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 14.299, df = NA, p-value = 0.1174
```

```
chisq.test(O2.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O2.trunc
## X-squared = 75.687, df = NA, p-value = 0.0009995
```

```
chisq.test(O3.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O3.trunc
## X-squared = 295.75, df = NA, p-value = 0.0004998
```

3 again

```
n.region <- 20
N <- max(data_unif, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data_unif)
O1.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51
##  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  3  0  1  1  1  3  1  3  0  0
## 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77
##  1  0  0  1  0  1  0  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 78 79 80
##  0  0  1
```

```
trunc=9
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 14.8
```

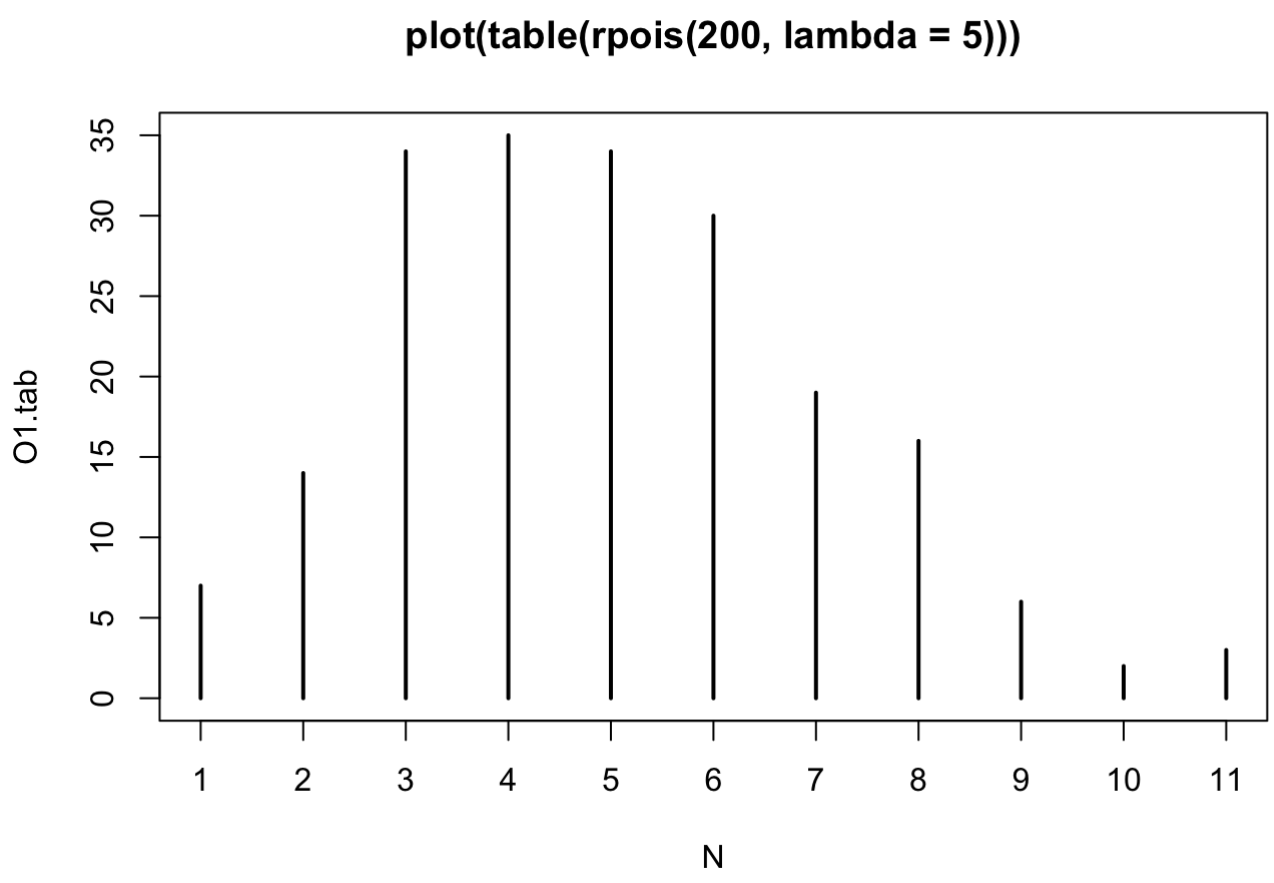
```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab1
```

```
##    levels Observed     Expected
## 1       0        0 7.472599e-06
## 2       1        0 1.105945e-04
## 3       2        0 8.183990e-04
## 4       3        0 4.037435e-03
## 5       4        0 1.493851e-02
## 6       5        0 4.421799e-02
## 7       6        0 1.090710e-01
## 8       7        0 2.306073e-01
## 9       8        0 4.266236e-01
## 10    >=9       20 1.916957e+01
```

```
(O1.tab <- table(N = stats::rpois(200, lambda = 5)))
```

```
## N
##  1  2  3  4  5  6  7  8  9 10 11
##  7 14 34 35 34 30 19 16  6  2  3
```

```
plot(O1.tab, main = "plot(table(rpois(200, lambda = 5)))")
```




```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 0.86641, df = NA, p-value = 1
```

```
n.region <- 30
N <- max(data_unif, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data_unif)
O1.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  1
## 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51
##  0  2  1  2  2  4  2  2  2  3  1  1  2  0  0  1  2  0  0  0  0  0  0  0  0  1
```

```
trunc=9
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 9.866667
```

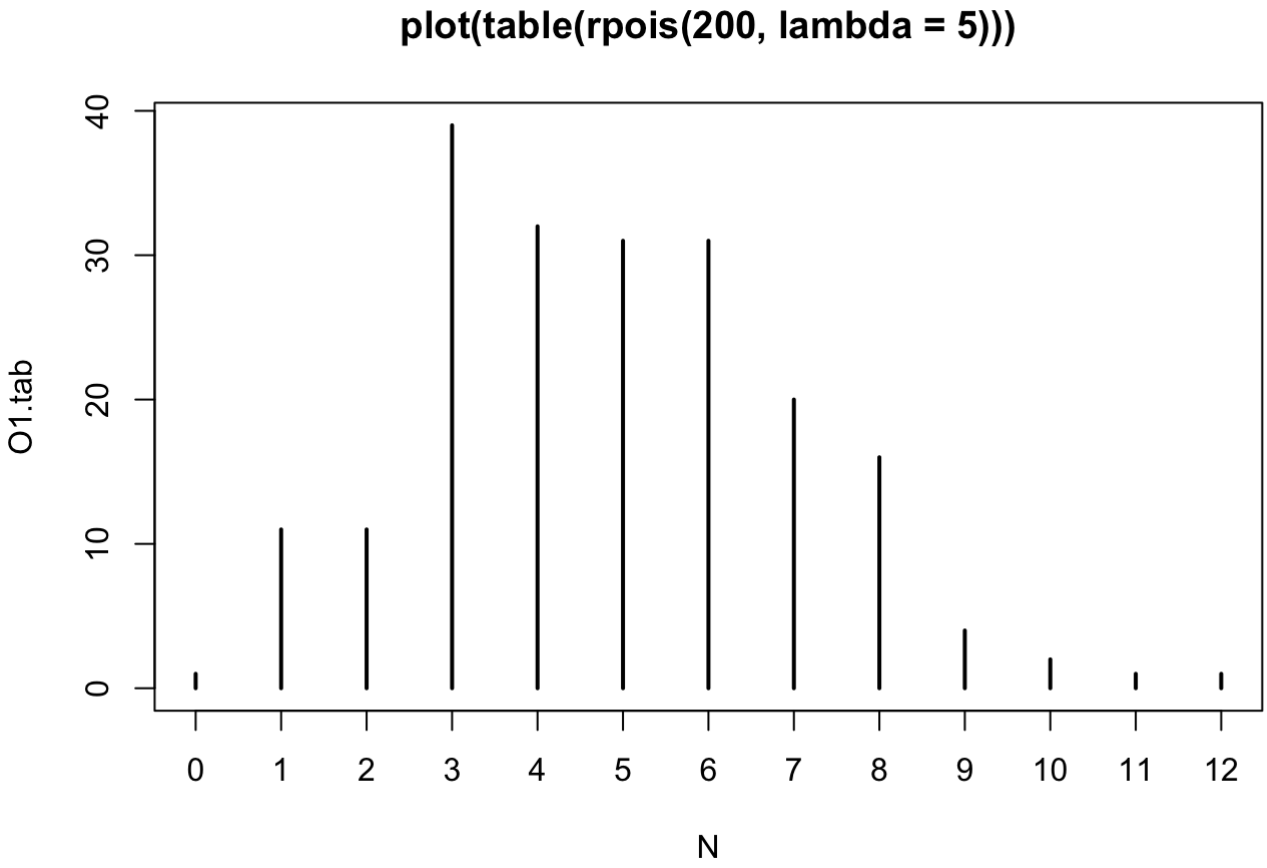
```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab1
```

```
##    levels Observed     Expected
## 1       0        0  0.001556261
## 2       1        0  0.015355106
## 3       2        0  0.075751857
## 4       3        0  0.249139441
## 5       4        0  0.614543954
## 6       5        0  1.212700069
## 7       6        0  1.994217891
## 8       7        0  2.810897599
## 9       8        0  3.466773705
## 10    >=9       30 19.559064117
```

```
(O1.tab <- table(N = stats::rpois(200, lambda = 5)))
```

```
## N
##  0  1  2  3  4  5  6  7  8  9 10 11 12
##  1 11 11 39 32 31 31 20 16  4  2  1  1
```

```
plot(O1.tab, main = "plot(table(rpois(200, lambda = 5)))")
```




```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 16.014, df = NA, p-value = 0.09195
```

```
n.region <- 40
N <- max(data_unif, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data_unif)
O1.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  3  0  7  2  5  1  1  1
## 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
##  3  3  4  3  0  1  1  2  0  0  0  0  0  1  0  1
```

```
trunc=9
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 7.4
```

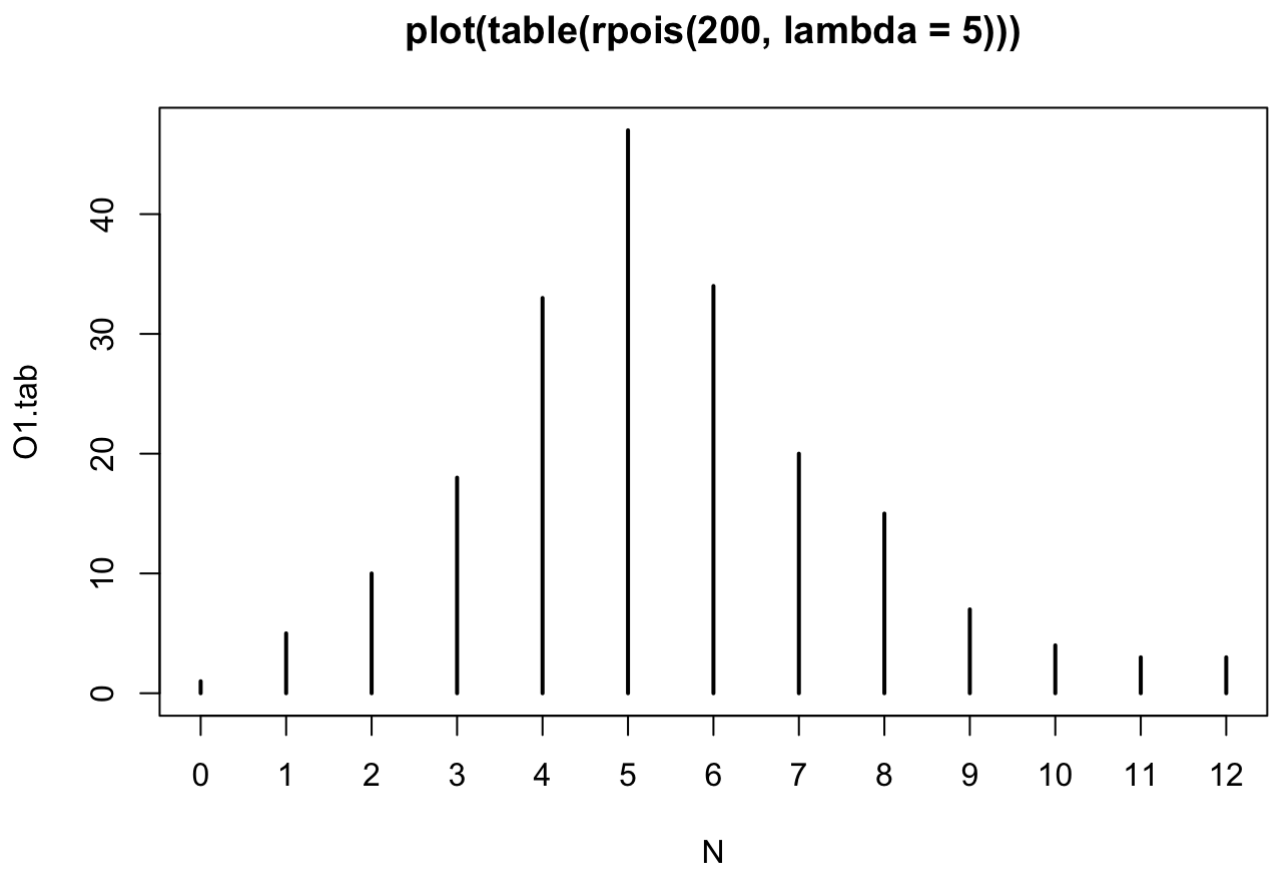
```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab1
```

```
##    levels Observed    Expected
## 1       0        0  0.02445011
## 2       1        0  0.18093082
## 3       2        0  0.66944402
## 4       3        0  1.65129526
## 5       4        0  3.05489623
## 6       5        0  4.52124642
## 7       6        0  5.57620392
## 8       7        0  5.89484414
## 9       8        0  5.45273083
## 10    >=9       40 12.97395825
```

```
(O1.tab <- table(N = stats::rpois(200, lambda = 5)))
```

```
## N
##  0  1  2  3  4  5  6  7  8  9 10 11 12
##  1  5 10 18 33 47 34 20 15  7  4  3  3
```

```
plot(O1.tab, main = "plot(table(rpois(200, lambda = 5)))")
```



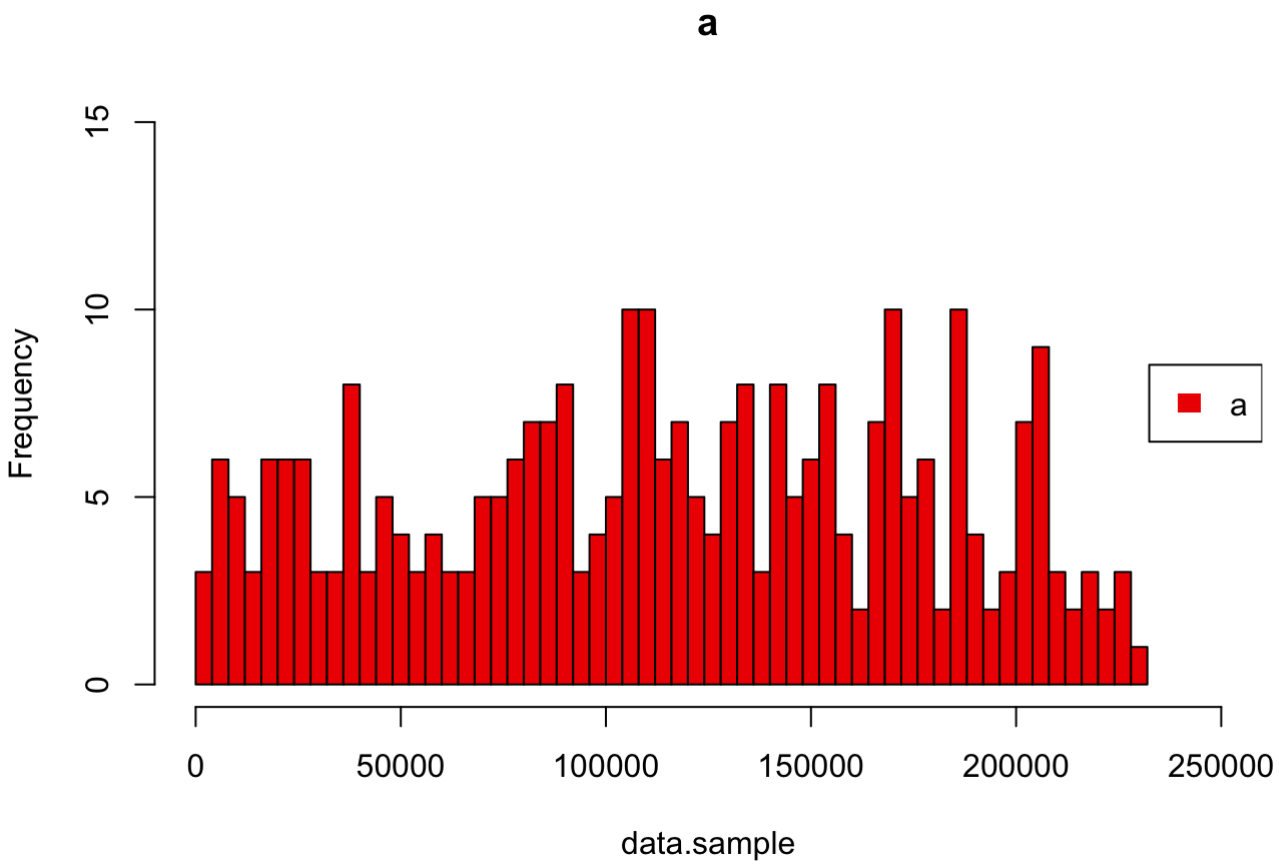

```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 83.324, df = NA, p-value = 0.0004998
```

4.

Biggest Cluster: Here the goal is to use randomization or theory to examine the largest cluster of palindromes in a sub-interval. Again, you're expected to try a few different interval sizes. With respect to the randomization, focus on the probability of obtaining, in any subinterval, a count as large or larger than the count you observe in your sample. There is also a theoretical approach to obtaining such a probability. You are free to implement either method.

```
N <- max(data$location, na.rm=TRUE)
n <- 296
gene <- seq(177, N)

set.seed(200)
data.sample <- sample.int(N, size=n, replace=FALSE)
aHistogram = hist(data.sample, ylim=c(0,15), xlim=c(0,250000), breaks=seq(0,232000, by=4
000), col='red2', main="a")

legend('right', c('a'), fill=c('red2'), border=NA)
```




```
N <- max(data$location, na.rm=TRUE)
n <- 296
gene <- seq(177, N)

set.seed(1)
data.sample <- sample.int(N, size=n, replace=FALSE)
```

```
n.region <- 20
N <- max(data.sample, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data.sample)
O1.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22
##  0  0  0  0  0  0  0  0  0  1  0  3  2  4  1  2  1  0  2  2  0  1  1
```

```
trunc=10
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 14.8
```

```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab1
```

```
##    levels Observed     Expected
## 1       0        0 7.472599e-06
## 2       1        0 1.105945e-04
## 3       2        0 8.183990e-04
## 4       3        0 4.037435e-03
## 5       4        0 1.493851e-02
## 6       5        0 4.421799e-02
## 7       6        0 1.090710e-01
## 8       7        0 2.306073e-01
## 9       8        0 4.266236e-01
## 10      9        1 7.015588e-01
## 11   >=10       19 1.846801e+01
```

```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 0.97271, df = NA, p-value = 1
```

4 revamp

```
n.region <- 10
N <- max(data_unif, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data_unif)
O1.tab
```

```
##
##   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99
##   0   0   0   0   1   0   0   0   0   0   1   1   0   0   1   0   1   0   0   0
## 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
##   0   1   1   0   0   0   1   0   1   0   0   0   0   0   0   0   0   0   0   0
## 120 121 122 123 124 125 126
##   0   0   0   0   0   0   1
```

```
trunc=9
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 29.6
```

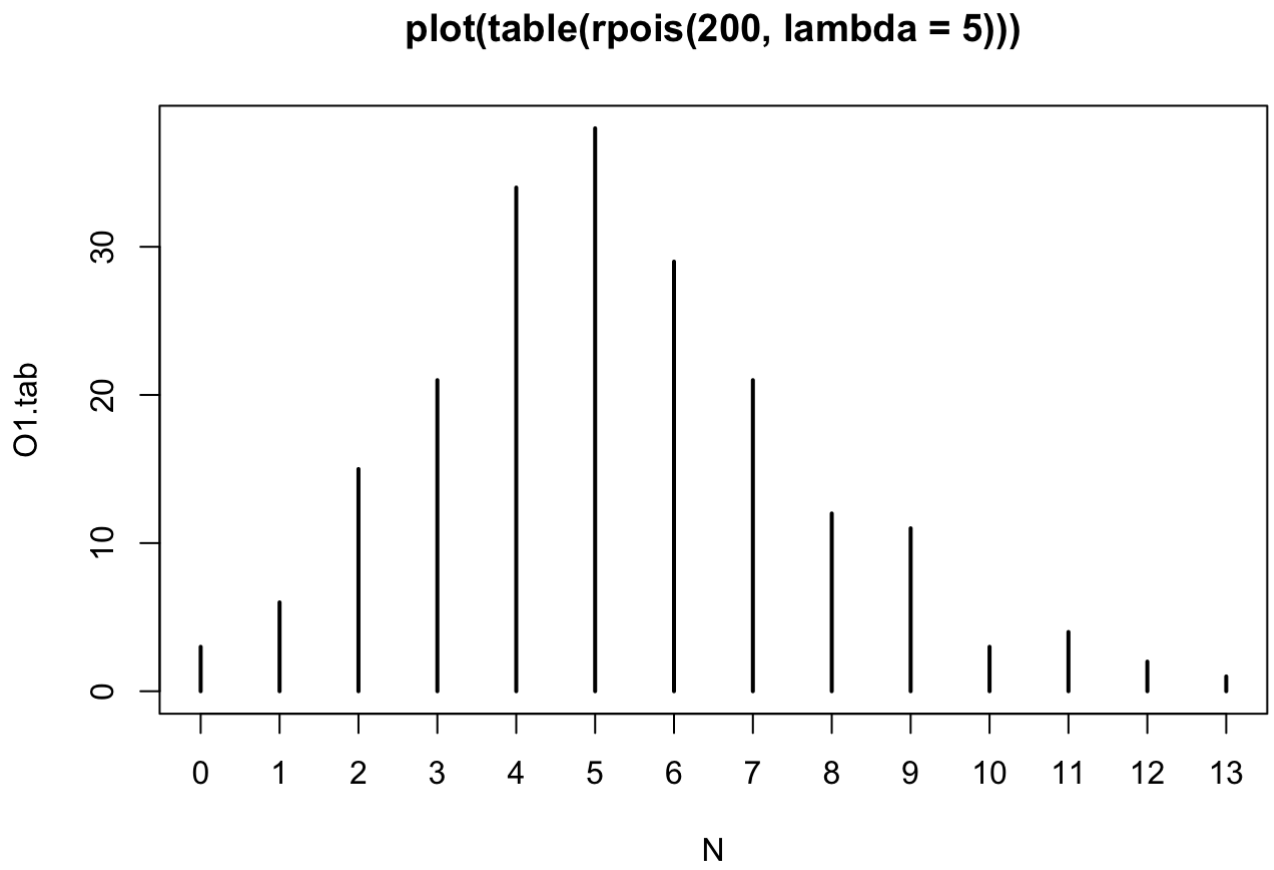
```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab1
```

```
##    levels Observed     Expected
## 1       0        0 1.395993e-12
## 2       1        0 4.132140e-11
## 3       2        0 6.115567e-10
## 4       3        0 6.034027e-09
## 5       4        0 4.465180e-08
## 6       5        0 2.643386e-07
## 7       6        0 1.304071e-06
## 8       7        0 5.514356e-06
## 9       8        0 2.040312e-05
## 10    >=9       10 9.999972e+00
```

```
(O1.tab <- table(N = stats::rpois(200, lambda = 5)))
```

```
## N
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13
##  3  6 15 21 34 38 29 21 12 11  3  4  2  1
```

```
plot(O1.tab, main = "plot(table(rpois(200, lambda = 5)))")
```




```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 2.7537e-05, df = NA, p-value = 1
```

```
n.region <- 20
N <- max(data_unif, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data_unif)
O1.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51
##  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  3  0  1  1  1  3  1  3  0  0
## 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77
##  1  0  0  1  0  1  0  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 78 79 80
##  0  0  1
```

```
trunc=9
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 14.8
```

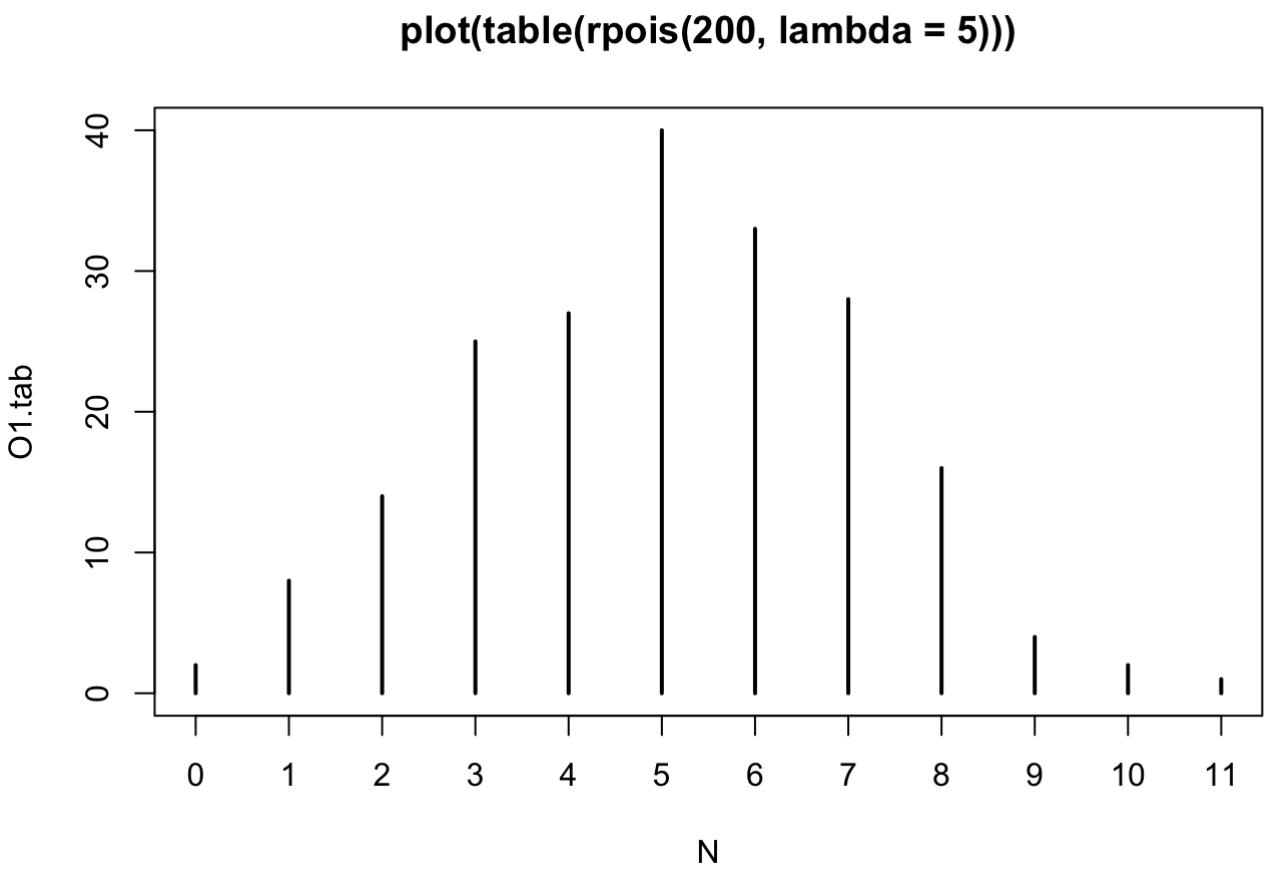
```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab1
```

```
##    levels Observed     Expected
## 1       0        0 7.472599e-06
## 2       1        0 1.105945e-04
## 3       2        0 8.183990e-04
## 4       3        0 4.037435e-03
## 5       4        0 1.493851e-02
## 6       5        0 4.421799e-02
## 7       6        0 1.090710e-01
## 8       7        0 2.306073e-01
## 9       8        0 4.266236e-01
## 10    >=9       20 1.916957e+01
```

```
(O1.tab <- table(N = stats::rpois(200, lambda = 5)))
```

```
## N
##  0  1  2  3  4  5  6  7  8  9 10 11
##  2  8 14 25 27 40 33 28 16  4  2  1
```

```
plot(O1.tab, main = "plot(table(rpois(200, lambda = 5)))")
```




```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 0.86641, df = NA, p-value = 1
```

```
n.region <- 30
N <- max(data_unif, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data_unif)
O1.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  1
## 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51
##  0  2  1  2  2  4  2  2  2  3  1  1  2  0  0  1  2  0  0  0  0  0  0  0  0  1
```

```
trunc=9
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 9.866667
```

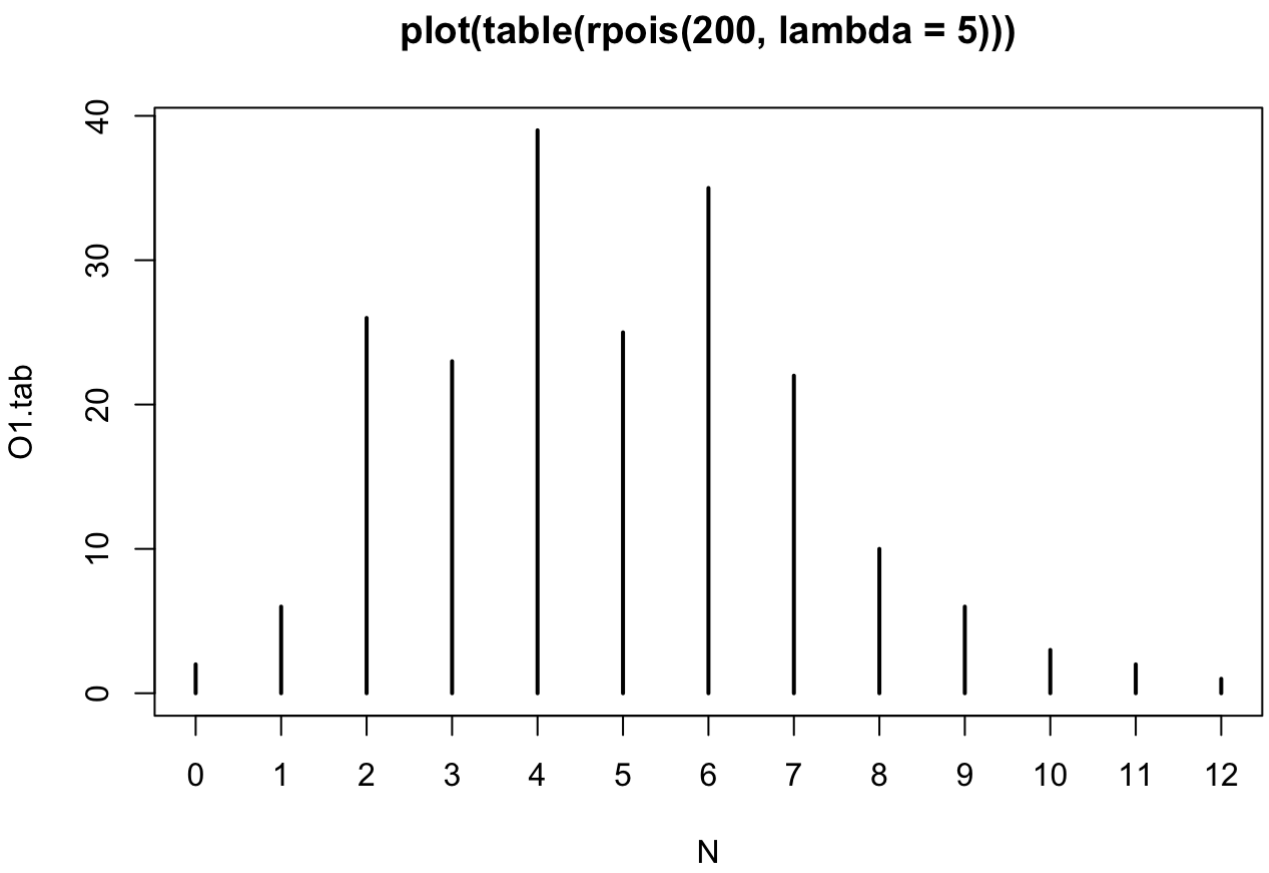
```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab1
```

```
##    levels Observed     Expected
## 1       0        0  0.001556261
## 2       1        0  0.015355106
## 3       2        0  0.075751857
## 4       3        0  0.249139441
## 5       4        0  0.614543954
## 6       5        0  1.212700069
## 7       6        0  1.994217891
## 8       7        0  2.810897599
## 9       8        0  3.466773705
## 10    >=9       30 19.559064117
```

```
(O1.tab <- table(N = stats::rpois(200, lambda = 5)))
```

```
## N
##  0  1  2  3  4  5  6  7  8  9 10 11 12
##  2  6 26 23 39 25 35 22 10  6  3  2  1
```

```
plot(O1.tab, main = "plot(table(rpois(200, lambda = 5)))")
```




```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 16.014, df = NA, p-value = 0.08796
```

```
n.region <- 40
N <- max(data_unif, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data_unif)
O1.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  3  0  7  2  5  1  1  1
## 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
##  3  3  4  3  0  1  1  2  0  0  0  0  0  1  0  1
```

```
trunc=9
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 7.4
```

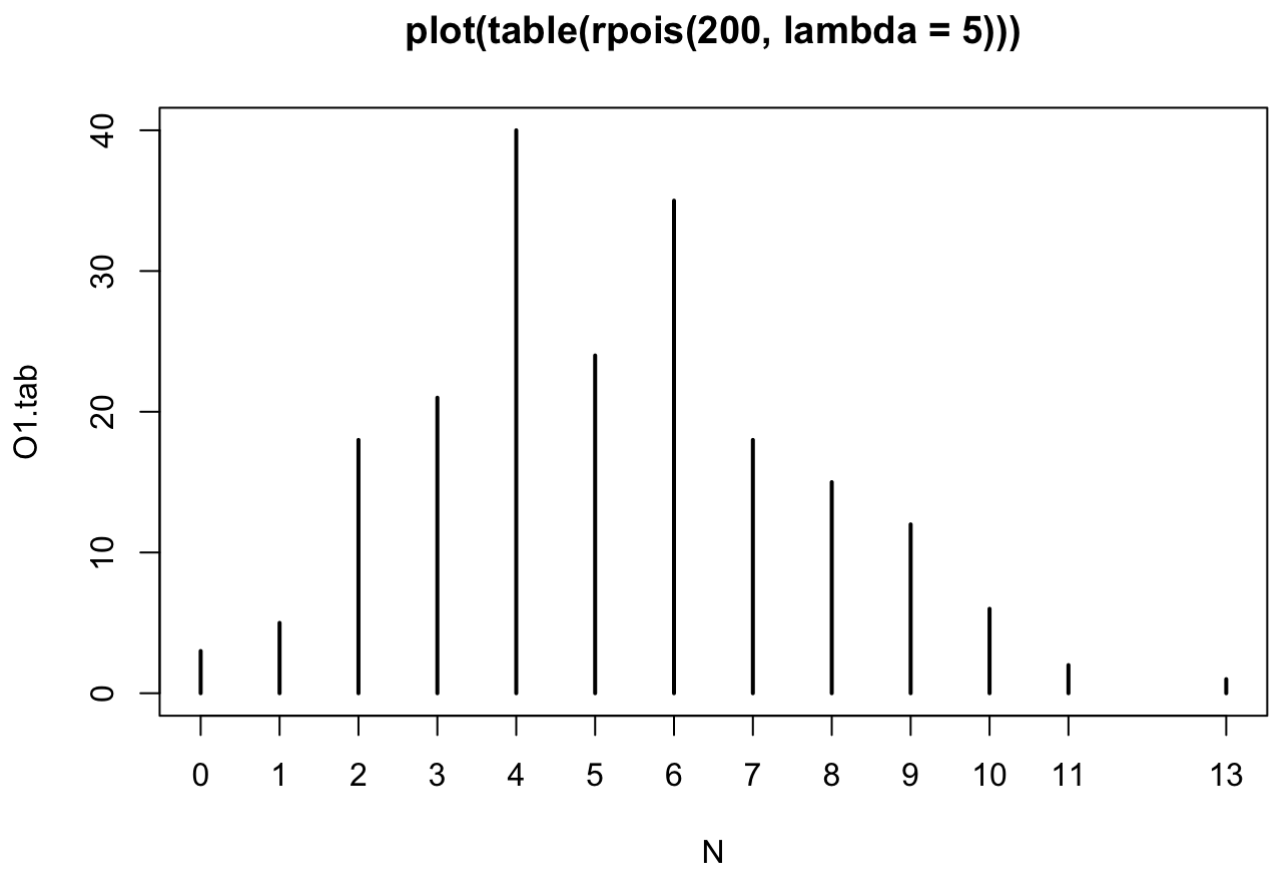
```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab1
```

```
##    levels Observed    Expected
## 1       0        0  0.02445011
## 2       1        0  0.18093082
## 3       2        0  0.66944402
## 4       3        0  1.65129526
## 5       4        0  3.05489623
## 6       5        0  4.52124642
## 7       6        0  5.57620392
## 8       7        0  5.89484414
## 9       8        0  5.45273083
## 10    >=9       40 12.97395825
```

```
(O1.tab <- table(N = stats::rpois(200, lambda = 5)))
```

```
## N
##  0  1  2  3  4  5  6  7  8  9 10 11 13
##  3  5 18 21 40 24 35 18 15 12  6  2  1
```

```
plot(O1.tab, main = "plot(table(rpois(200, lambda = 5)))")
```




```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 83.324, df = NA, p-value = 0.0004998
```

```
n.region <- 50
N <- max(data_unif, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data_unif)
O1.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  0  0  0  0  0  0  0  0  0  0  1  0  0  2  1  4  2  2  7  4  4  8  4  3  3  1
## 26 27 28 29 30 31 32 33 34
##  0  0  3  0  0  0  0  0  1
```

```
trunc=9
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 5.92
```

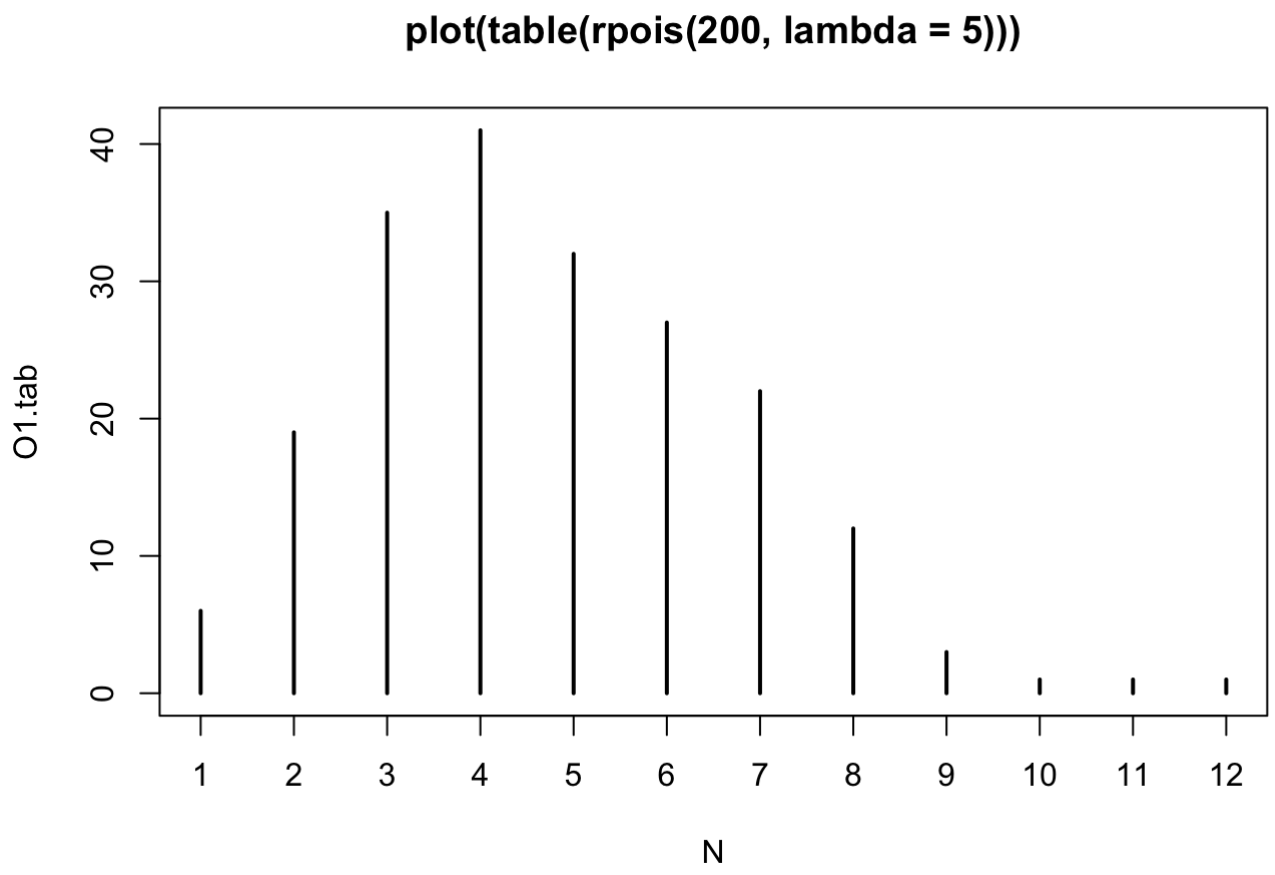
```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab1
```

```
##    levels Observed  Expected
## 1       0        0 0.1342600
## 2       1        0 0.7948193
## 3       2        0 2.3526650
## 4       3        0 4.6425922
## 5       4        0 6.8710365
## 6       5        0 8.1353072
## 7       6        0 8.0268365
## 8       7        0 6.7884103
## 9       8        0 5.0234236
## 10    >=9       50 7.2306494
```

```
(O1.tab <- table(N = stats::rpois(200, lambda = 5)))
```

```
## N
##  1  2  3  4  5  6  7  8  9 10 11 12
##  6 19 35 41 32 27 22 12  3  1  1  1
```

```
plot(O1.tab, main = "plot(table(rpois(200, lambda = 5)))")
```




```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 295.75, df = NA, p-value = 0.0004998
```

```
n.region <- 60
N <- max(data_unif, na.rm=TRUE)
gene <- seq(177, N)
O1.tab=regionsplit(n.region, gene, data_unif)
O1.tab
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
##  0  0  0  0  0  0  0  0  0  2  0  2  6  3  7  7  5  4  6  7  2  2  2  1  2  0
## 26 27 28 29
##  0  0  0  2
```

```
trunc=9
lvls=factor(c(0:(trunc-1),paste(">=",trunc,sep="")),levels=c(0:(trunc-1),paste(">=",trun
c,sep="")))
O1=as.vector(O1.tab)
O1.trunc=c(O1[1:trunc],sum(O1[-(1:trunc)]))
lambda=n/n.region
lambda
```

```
## [1] 4.933333
```

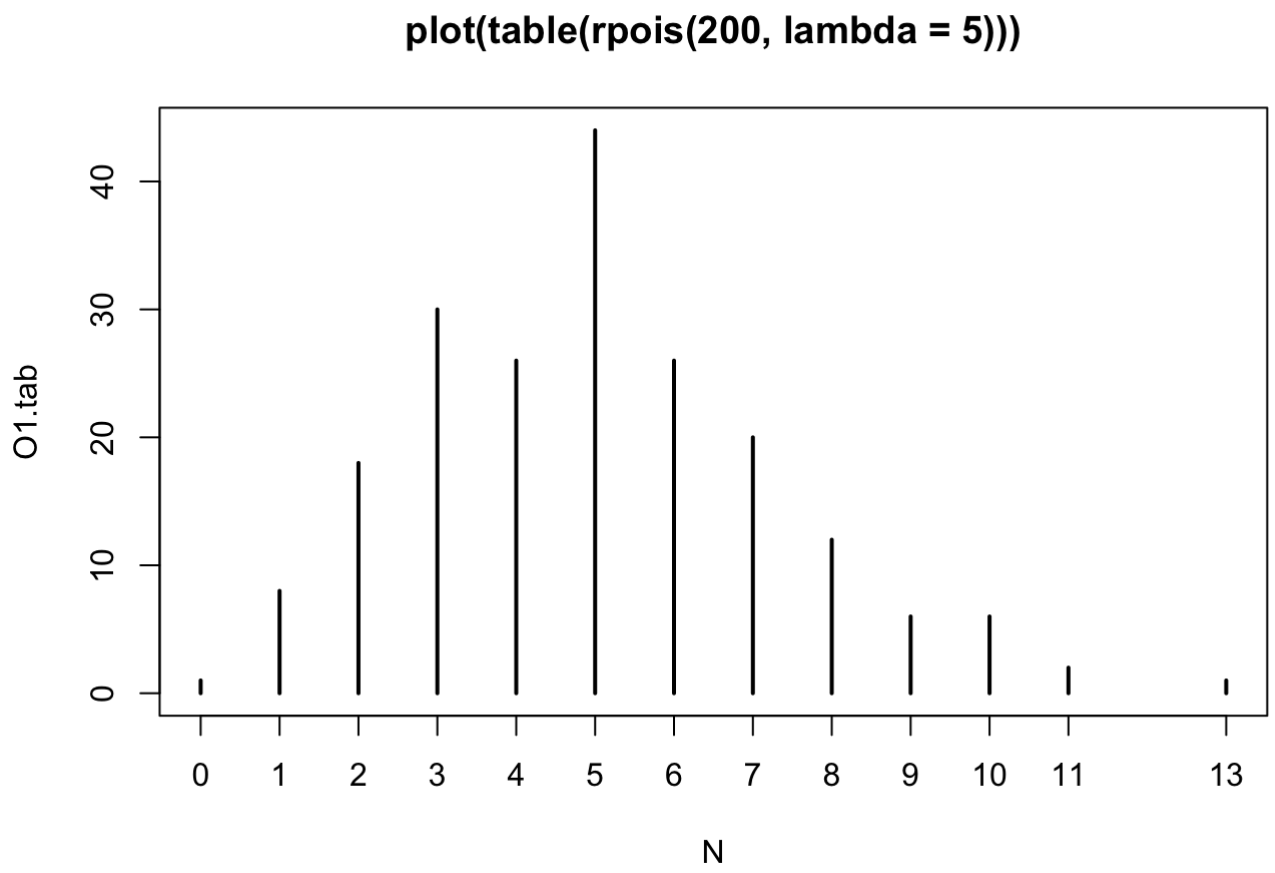
```
p=c(dpois(0:(trunc-1),lambda),1-sum(dpois(0:(trunc-1),lambda)))
E=p*n.region
tab1=data.frame(levels=lvls,Observed=O1.trunc,Expected=E)
tab1
```

```
##    levels Observed   Expected
## 1       0        0  0.4321473
## 2       1        0  2.1319267
## 3       2        0  5.2587525
## 4       3        0  8.6477263
## 5       4        0 10.6655292
## 6       5        0 10.5233221
## 7       6        0  8.6525093
## 8       7        0  6.0979589
## 9       8        0  3.7604080
## 10    >=9       60  3.8297197
```

```
(O1.tab <- table(N = stats::rpois(200, lambda = 5)))
```

```
## N
##  0  1  2  3  4  5  6  7  8  9 10 11 13
##  1  8 18 30 26 44 26 20 12  6  6  2  1
```

```
plot(O1.tab, main = "plot(table(rpois(200, lambda = 5)))")
```



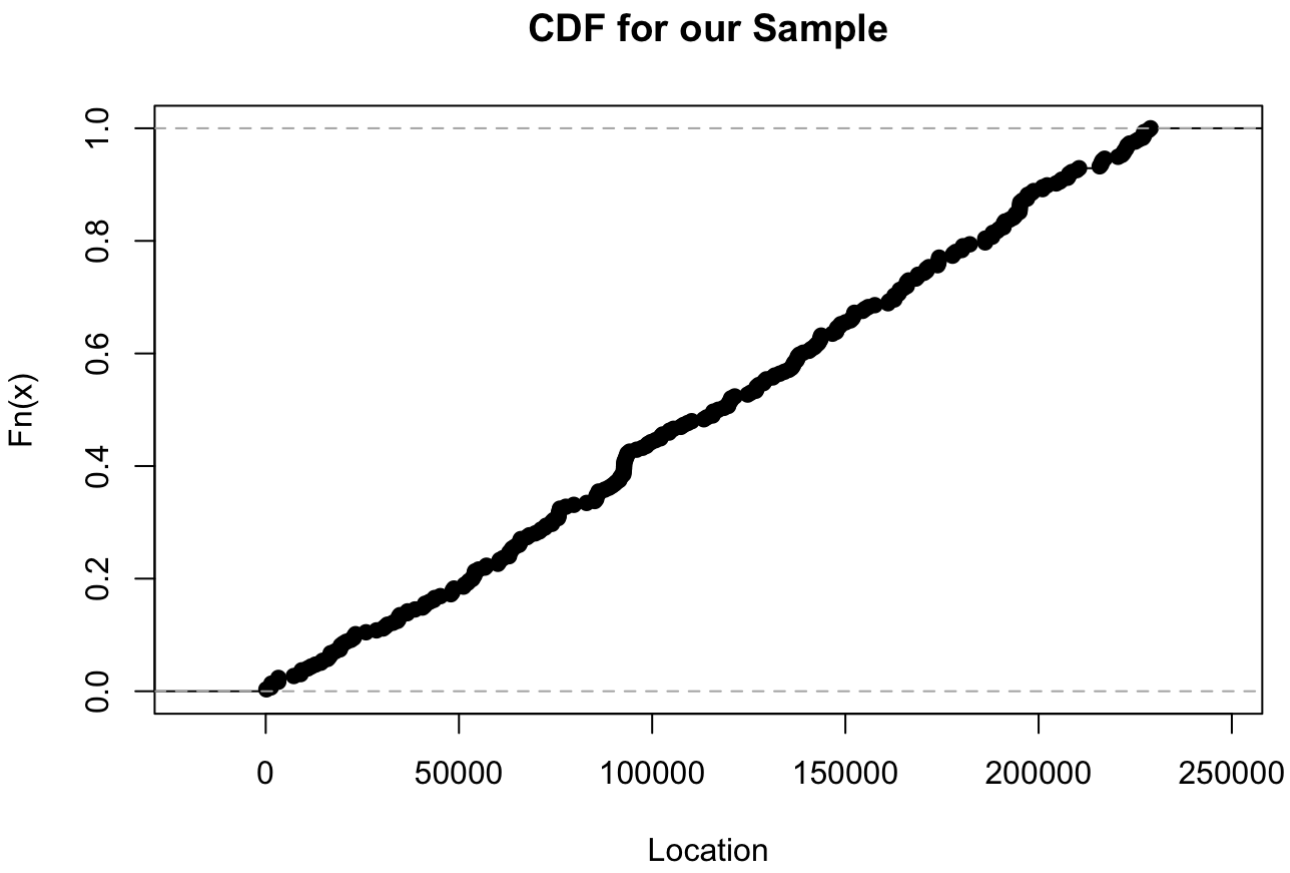

```
chisq.test(O1.trunc,p=p,simulate.p.value=TRUE)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value (based
##  on 2000 replicates)
##
## data:  O1.trunc
## X-squared = 880.02, df = NA, p-value = 0.0004998
```

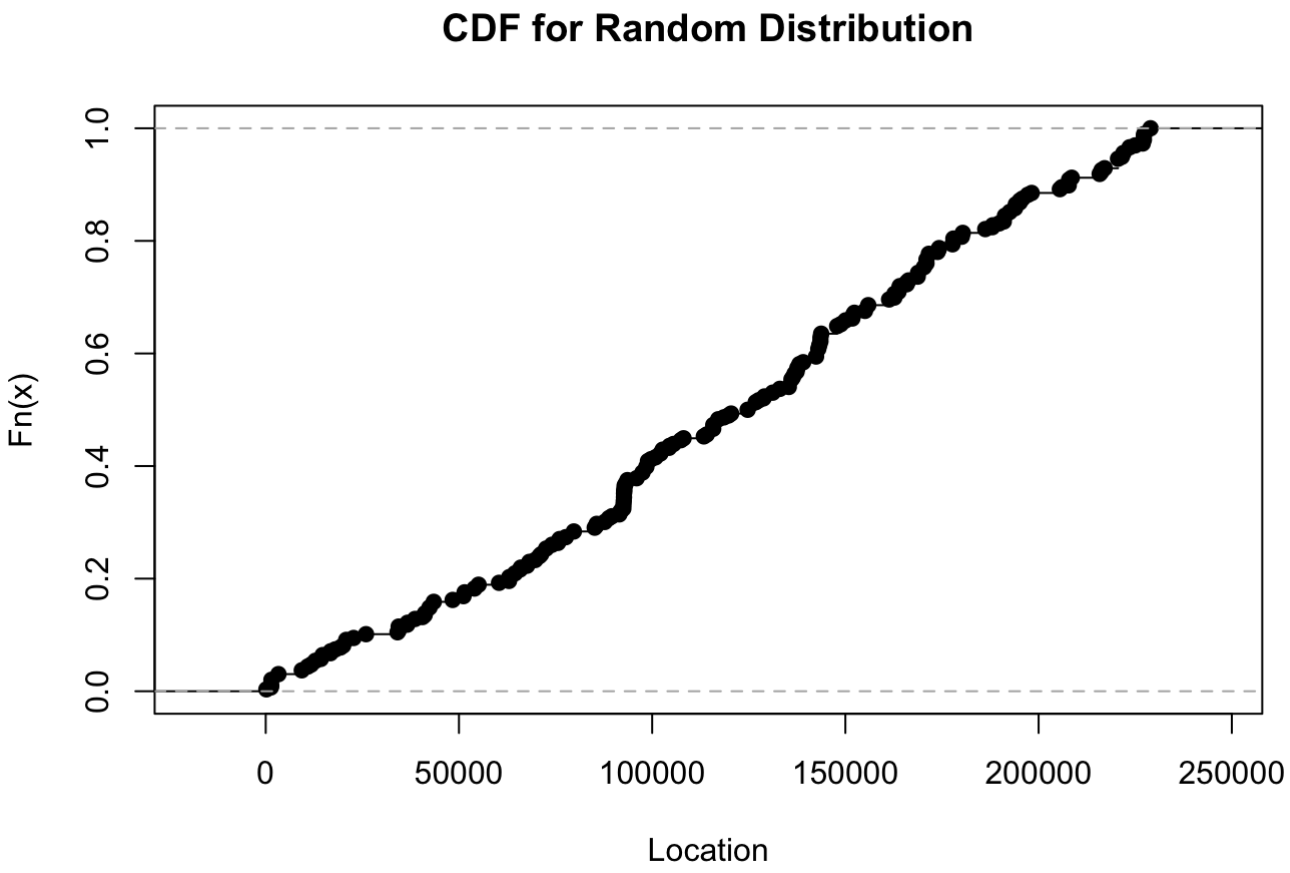
Advanced Analysis

```
data <- read.table("hcmv.txt", header=TRUE)
```

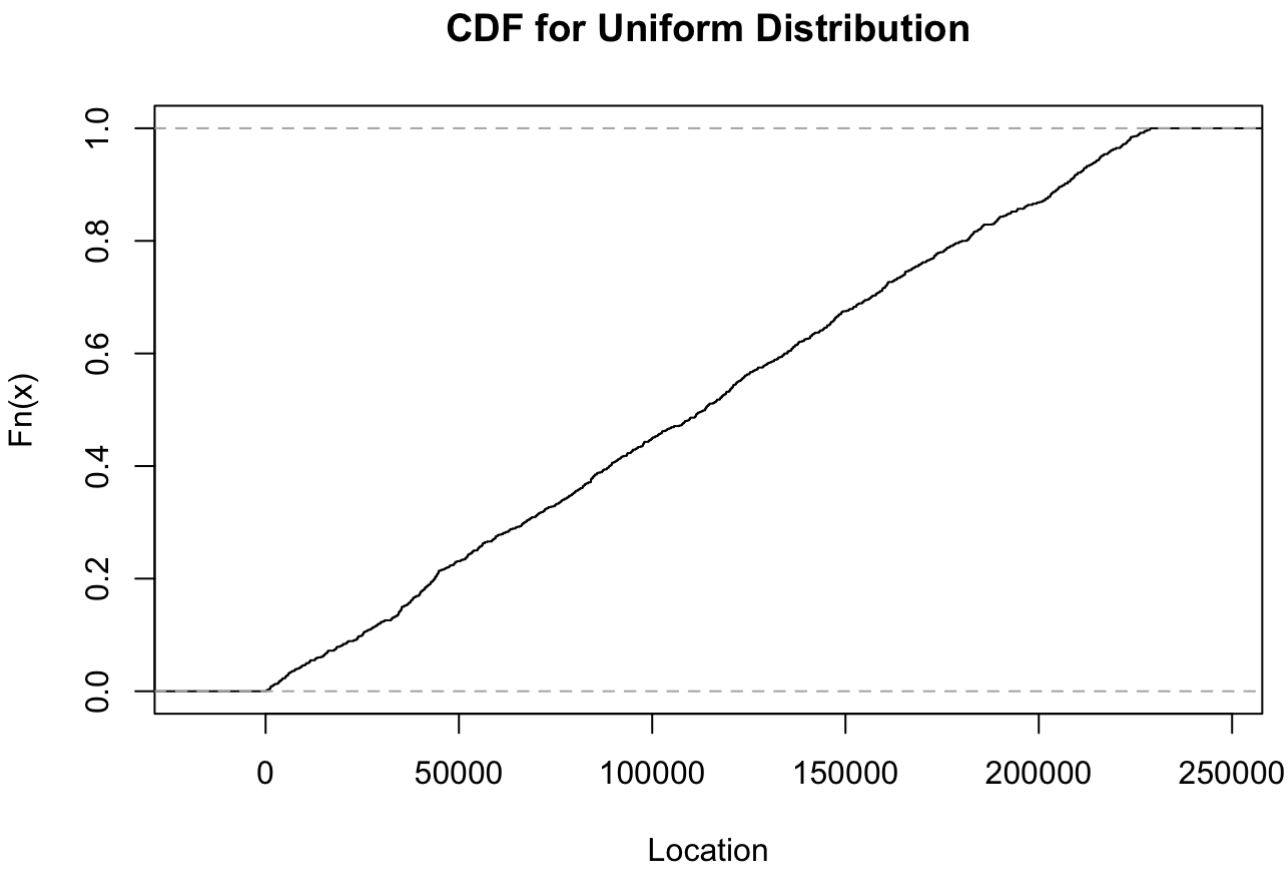
```
cdf.data = ecdf(data$location)
plot(cdf.data,main="CDF for our Sample",xlab="Location")
```




```
set.seed(100)
random_sample = sample(x=data$location, size=296, replace=TRUE)
cdf.random = ecdf(random_sample)
plot(cdf.random,main="CDF for Random Distribution",xlab="Location")
```




```
set.seed(2017)
n <- 1000
data_unif <- runif(n, min=177, max=228953)
cdf.unif = ecdf(data_unif)
plot(cdf.unif,main="CDF for Uniform Distribution",xlab="Location")
```


CDF for Uniform Distribution
Fn(x)
0.0
0.2
0.4
0.6
0.8
1.0
0
50000
100000
150000
200000
250000
Location