Bayes

NarenSuri

September 8, 2016

```
myfunction <- function(sizeval){
## dice rolling

 x = sample(x = 1:6, size = sizeval, replace = TRUE)
 x
 # number of 1's 2's .... etc...  came of rolling
 b=table(x)
 b
 as.vector(b)

 # The real probability should be :
 (1/6)^sizeval
 a=c(1:6)

 # number of 1's
 a[1]= b[1]/sizeval

  # number of 2's
 a[2]=b[2]/sizeval

  # number of 3's
 a[3]=b[3]/sizeval

 # number of 4's
 a[4]=b[4]/sizeval

 # number of 5's
 a[5]=b[5]/sizeval

 # number of 6's
 a[6]=b[6]/sizeval

 return(a)
}
 # The real probability should be :
 (1/6)^50
```

```
## [1] 1.237193e-39
```

```
# repaaat this
ui = rep.int(50*log(1/6),6)
# in log scale
50*log(1/6)
```

```
## [1] -89.58797
```

```
i = myfunction(50)
i
```

```
## [1] 0.18 0.18 0.18 0.20 0.12 0.14
```

```
# log og results obained
li = log(i)
li
```

```
## [1] -1.714798 -1.714798 -1.714798 -1.609438 -2.120264 -1.966113
```

```
## Question B
# repeat experiment 300 times
 (1/6)^300
```

```
## [1] 3.586121e-234
```

```
# in log scale
300*log(1/6)
```

```
## [1] -537.5278
```

```
uj = rep.int(300*log(1/6),6)
j = myfunction(300)
j
```

```
## [1] 0.1933333 0.1700000 0.1366667 0.1800000 0.1500000 0.1700000
```

```
# log og results obained
lj =log(j)
lj
```

```
## [1] -1.643339 -1.771957 -1.990210 -1.714798 -1.897120 -1.771957
```

```
## Question C
# repeat experiment 300 times
 (1/6)^1000
```

```
## [1] 0
```

```
# in log scale
1000*log(1/6)
```

```
## [1] -1791.759
```

```
# repaaat this
uk = rep.int(1000*log(1/6),6)
k = myfunction(1000)
k
```

```
## [1] 0.190 0.162 0.173 0.181 0.140 0.154
```

```
# log og results obained
lk = log(k)
lk
```

```
## [1] -1.660731 -1.820159 -1.754464 -1.709258 -1.966113 -1.870803
```

```
## Question D # not asked but just did
# repeat experiment 300 times
 (1/6)^10000
```

```
## [1] 0
```

```
# in log scale
10000*log(1/6)
```

```
## [1] -17917.59
```

```
# repaaat this
ul = rep.int(10000*log(1/6),6)
l = myfunction(10000)
l
```

```
## [1] 0.1713 0.1706 0.1622 0.1714 0.1646 0.1599
```

```
# log og results obained
ll =log(l)
ll
```

```
## [1] -1.764339 -1.768434 -1.818925 -1.763755 -1.804237 -1.833207
```

```
###############
#create a matrix
# matProb=matrix(c(ui,uj,uk,ul), nrow=4, ncol=6,byrow=TRUE)
# matProb

# matFreq=matrix(c(li, lj, lk), nrow=8, ncol=6,byrow=TRUE)
#

# dimnames(matFreq) = list(c(50,300,1000,10000),# row names
# +   c(1:6)) # column names
# df = data.frame(matFreq)
df
```

```
## function (x, df1, df2, ncp, log = FALSE)
## {
##     if (missing(ncp))
##         .Call(C_df, x, df1, df2, log)
##     else .Call(C_dnf, x, df1, df2, ncp, log)
## }
## <bytecode: 0x00000000145cf150>
## <environment: namespace:stats>
```

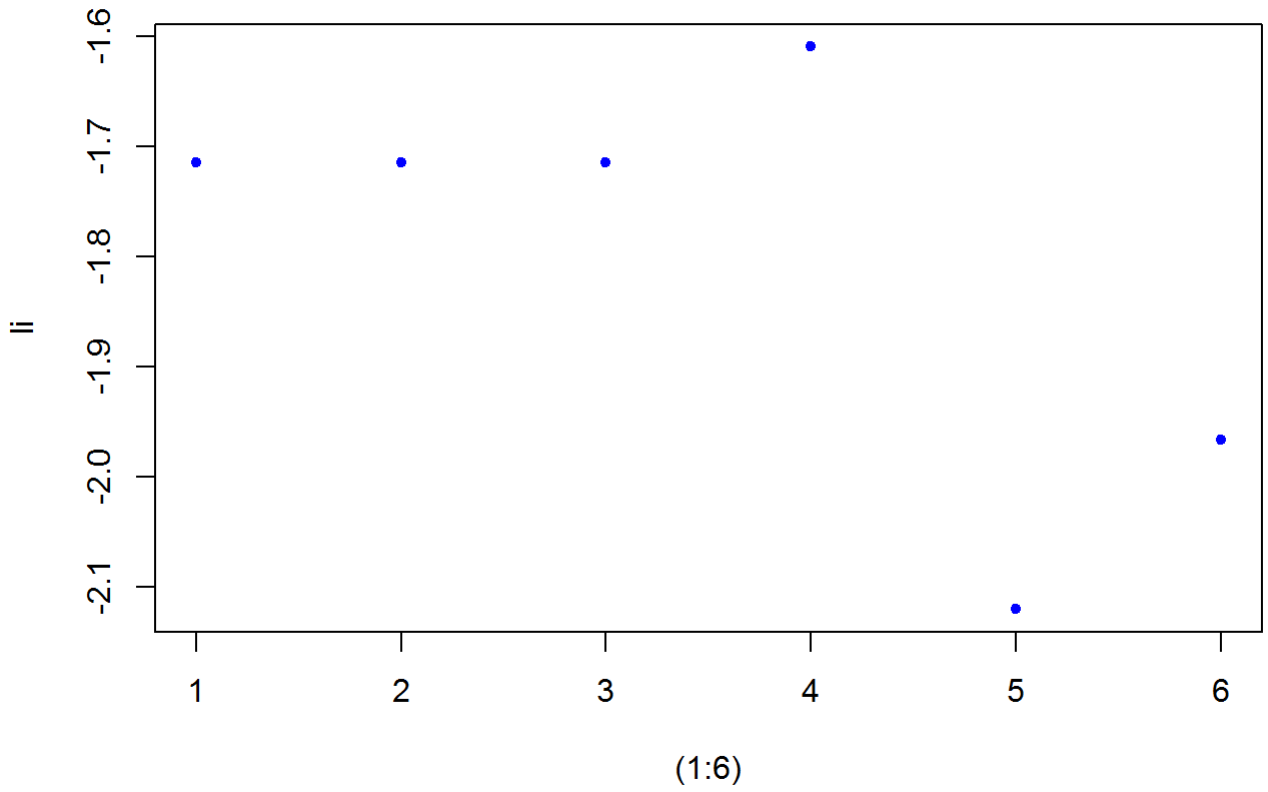
```
#   cdf = cbind.data.frame(sample=c(50,300,1000,10000),df)
# plot
 library("ggplot2", lib.loc="~/R/win-library/3.2")
```

```
## Warning: package 'ggplot2' was built under R version 3.2.5
```

```
# ggplot(data = cdf, aes(x=sample,y=) + geom_line(aes(colour=1:8)))
# plotmatrix(mat, mapping = aes(), colour = 1:8)
# sample 50
li
```

```
## [1] -1.714798 -1.714798 -1.714798 -1.609438 -2.120264 -1.966113
```

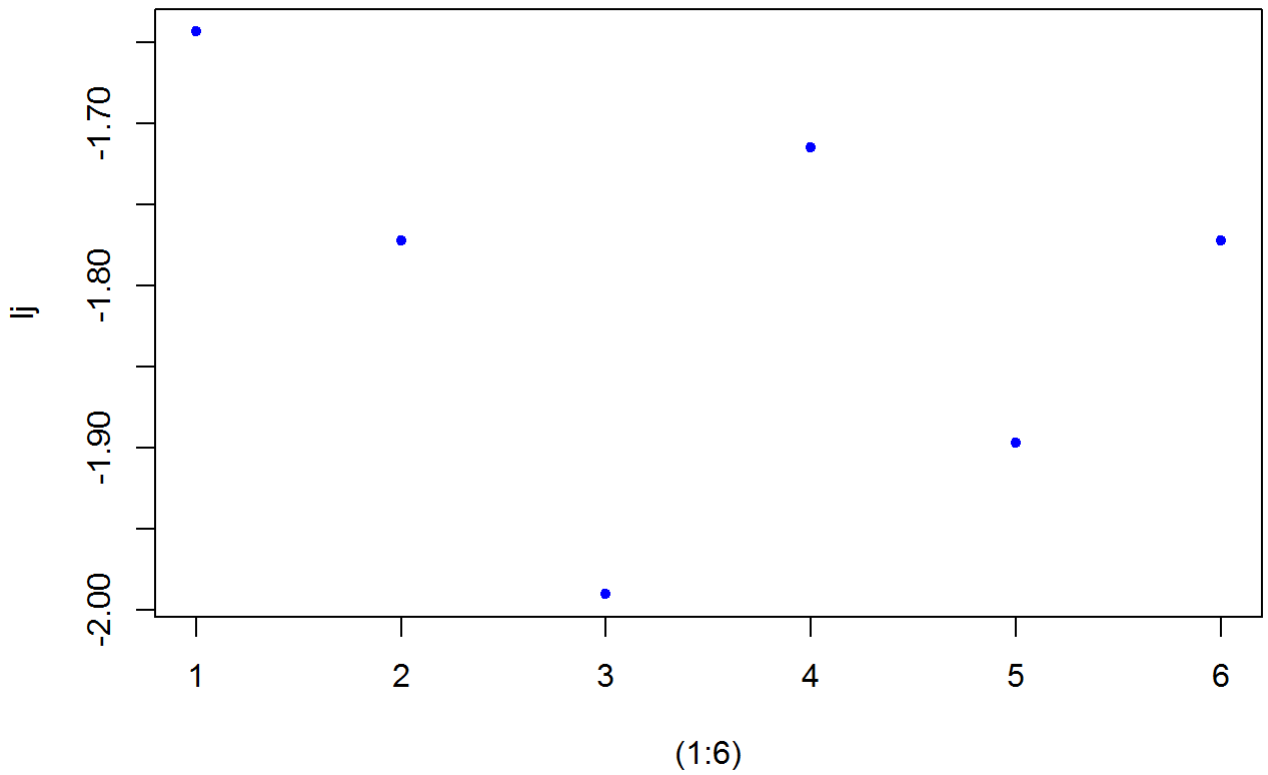
```
plot((1:6),li, col = "blue", pch = 20)
```




```
# sample 300
lj
```

```
## [1] -1.643339 -1.771957 -1.990210 -1.714798 -1.897120 -1.771957
```

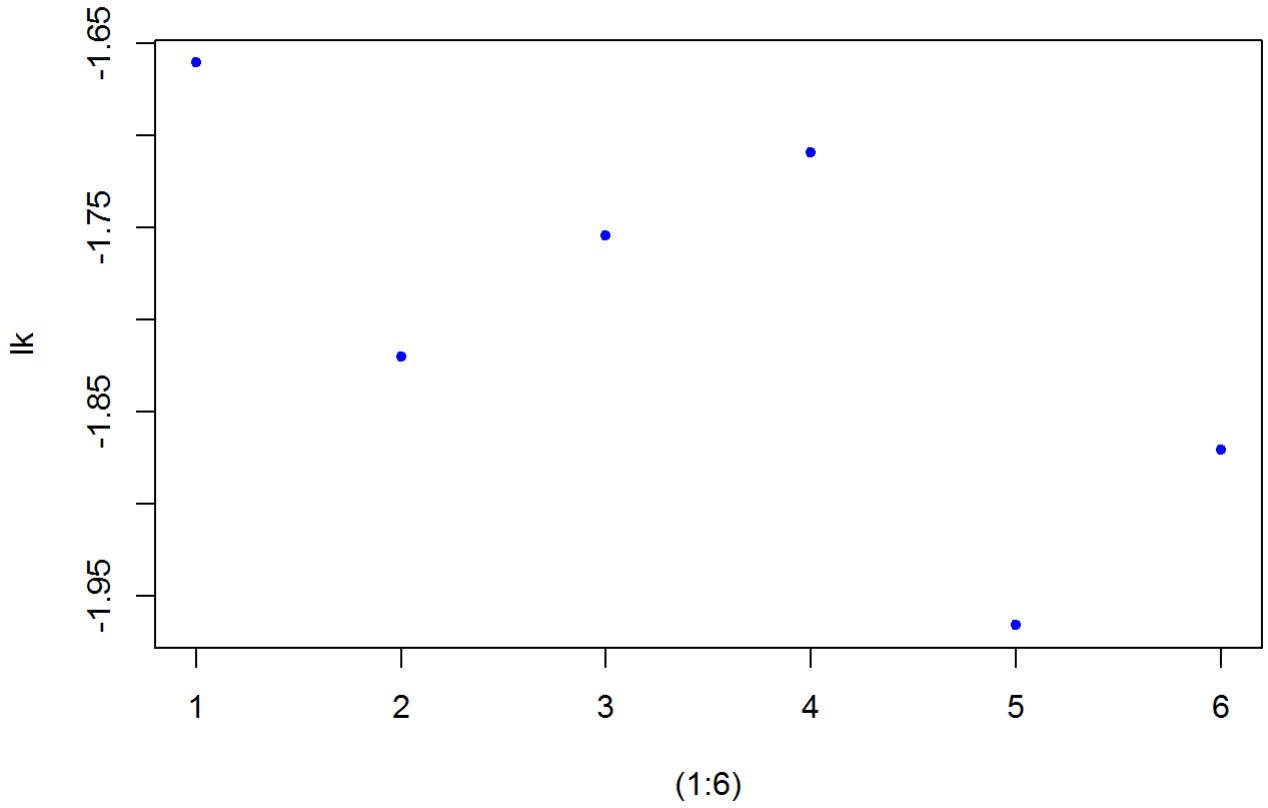
```
plot((1:6),lj, col = "blue", pch = 20)
```




```
# sample 1000
lk
```

```
## [1] -1.660731 -1.820159 -1.754464 -1.709258 -1.966113 -1.870803
```

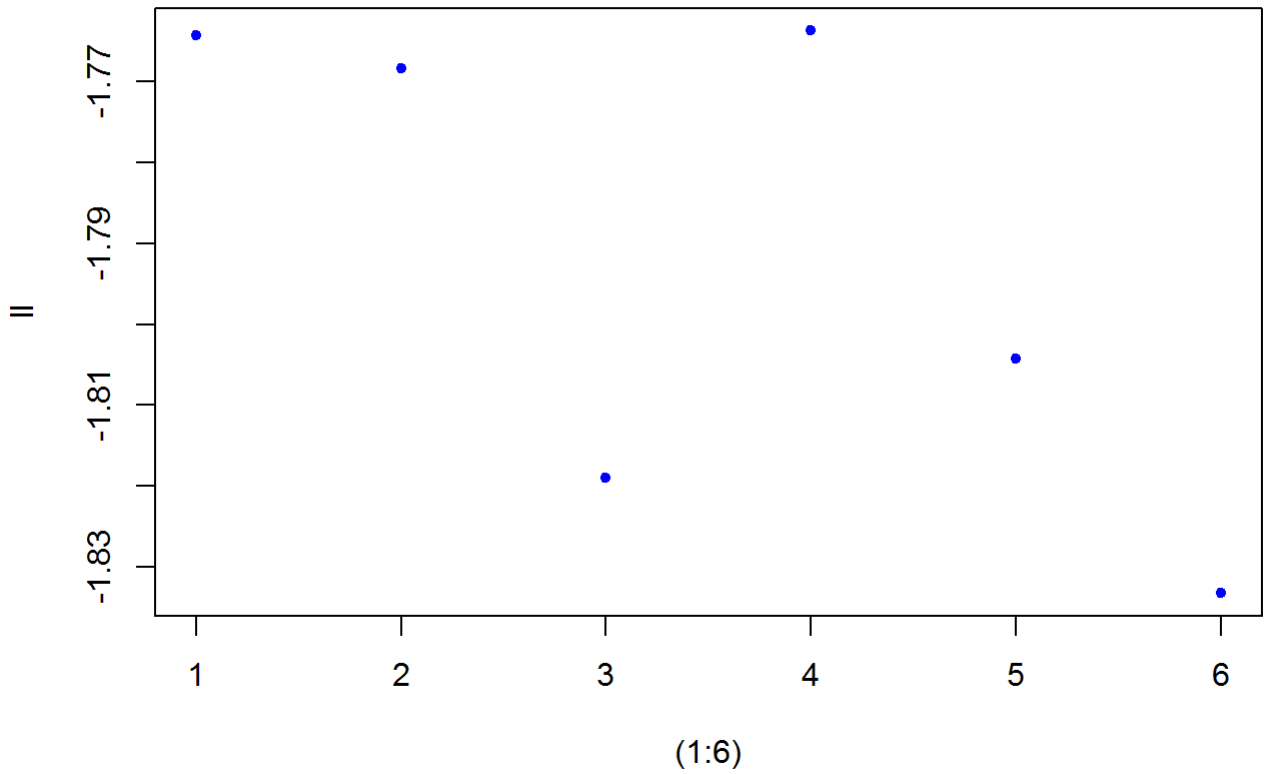
```
plot((1:6),lk, col = "blue", pch = 20)
```




```
# sample 10000
ll
```

```
## [1] -1.764339 -1.768434 -1.818925 -1.763755 -1.804237 -1.833207
```

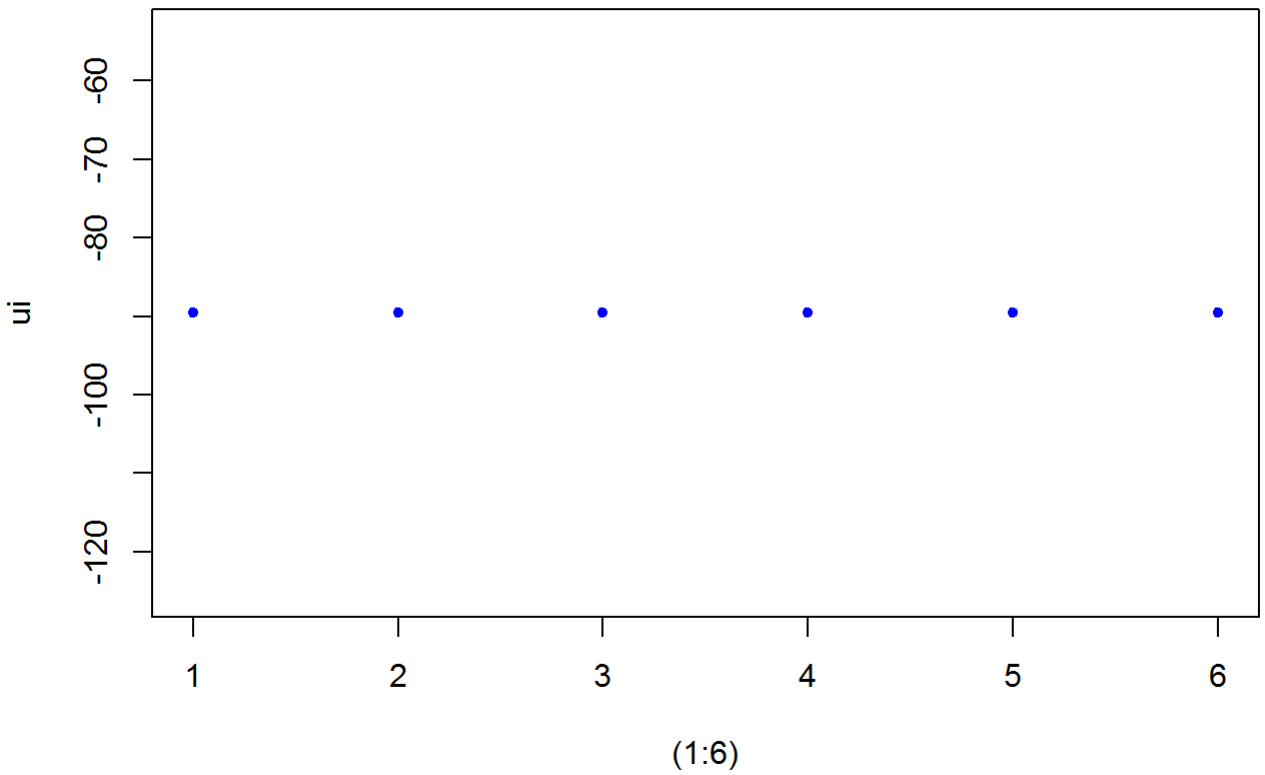
```
plot((1:6),ll, col = "blue", pch = 20)
```




```
# sample 50 prob
ui
```

```
## [1] -89.58797 -89.58797 -89.58797 -89.58797 -89.58797 -89.58797
```

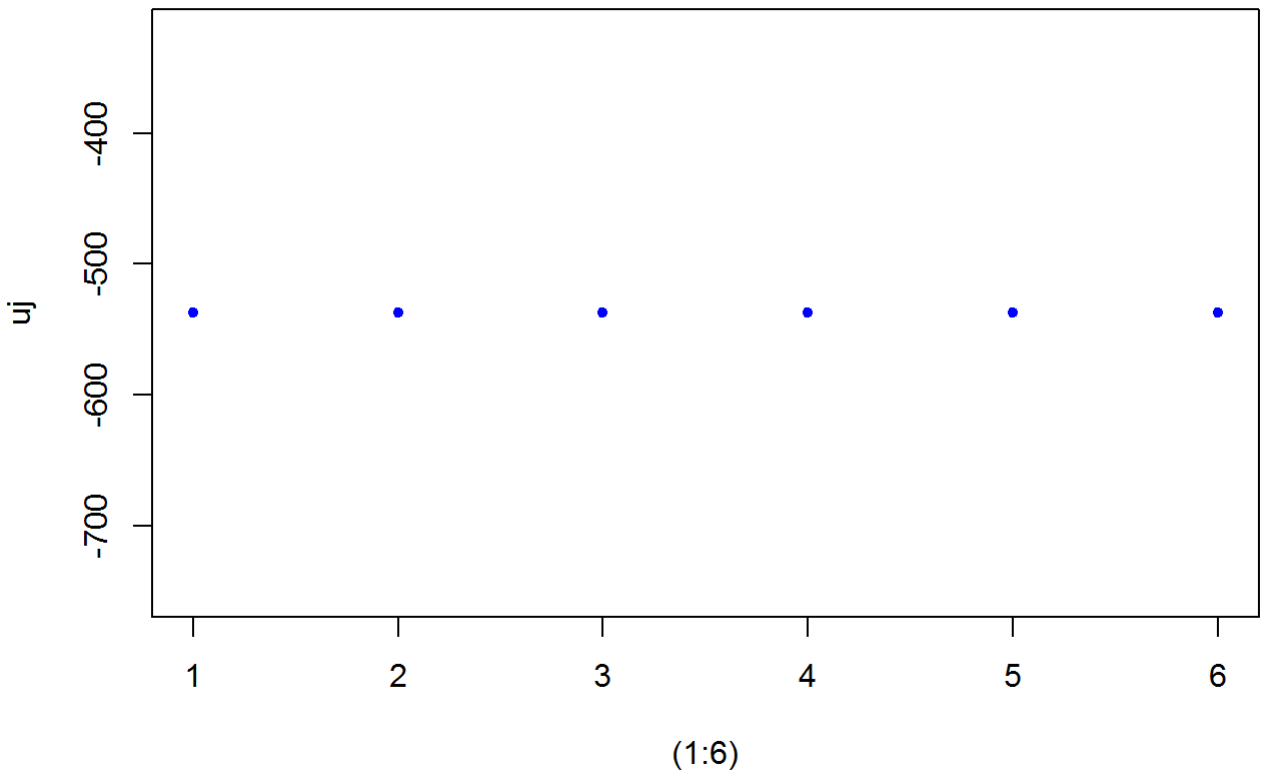
```
plot((1:6),ui, col = "blue", pch = 20)
```




```
# sample 300 prob
uj
```

```
## [1] -537.5278 -537.5278 -537.5278 -537.5278 -537.5278 -537.5278
```

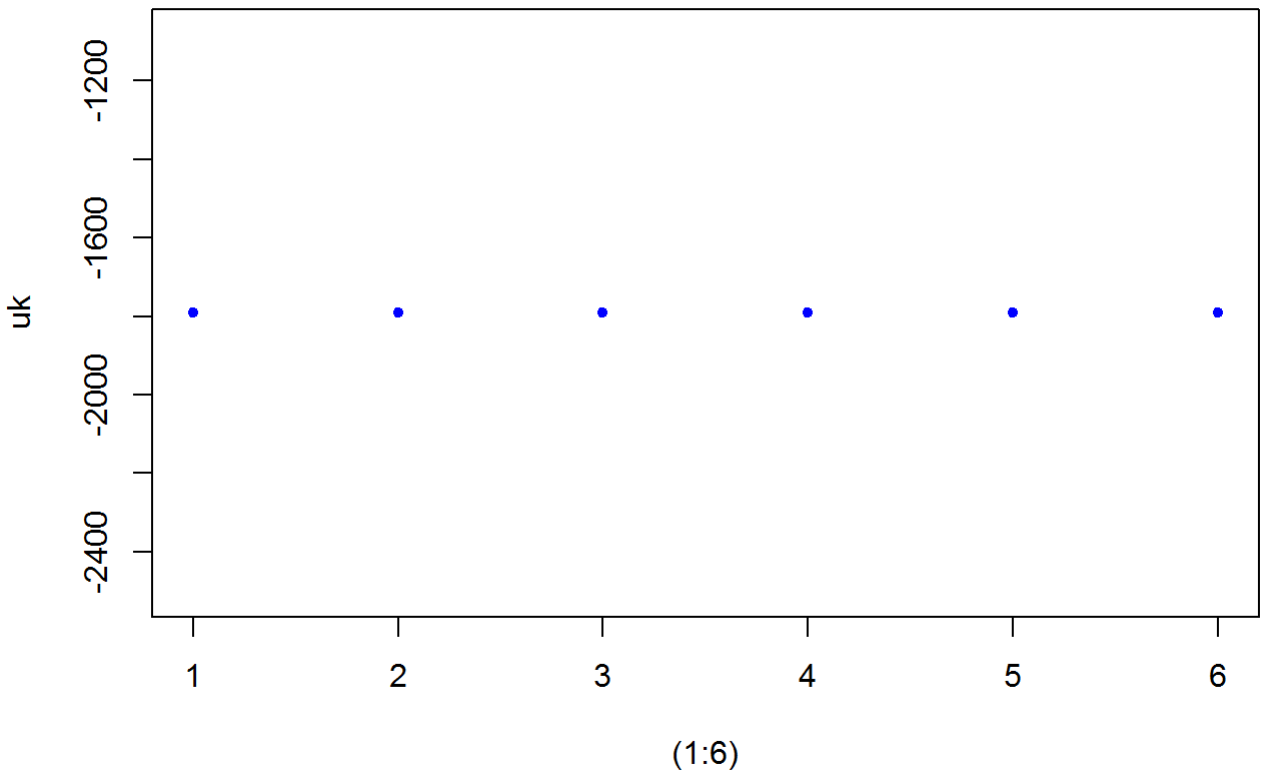
```
plot((1:6),uj, col = "blue", pch = 20)
```




```
# sample 1000 prob
uk
```

```
## [1] -1791.759 -1791.759 -1791.759 -1791.759 -1791.759 -1791.759
```

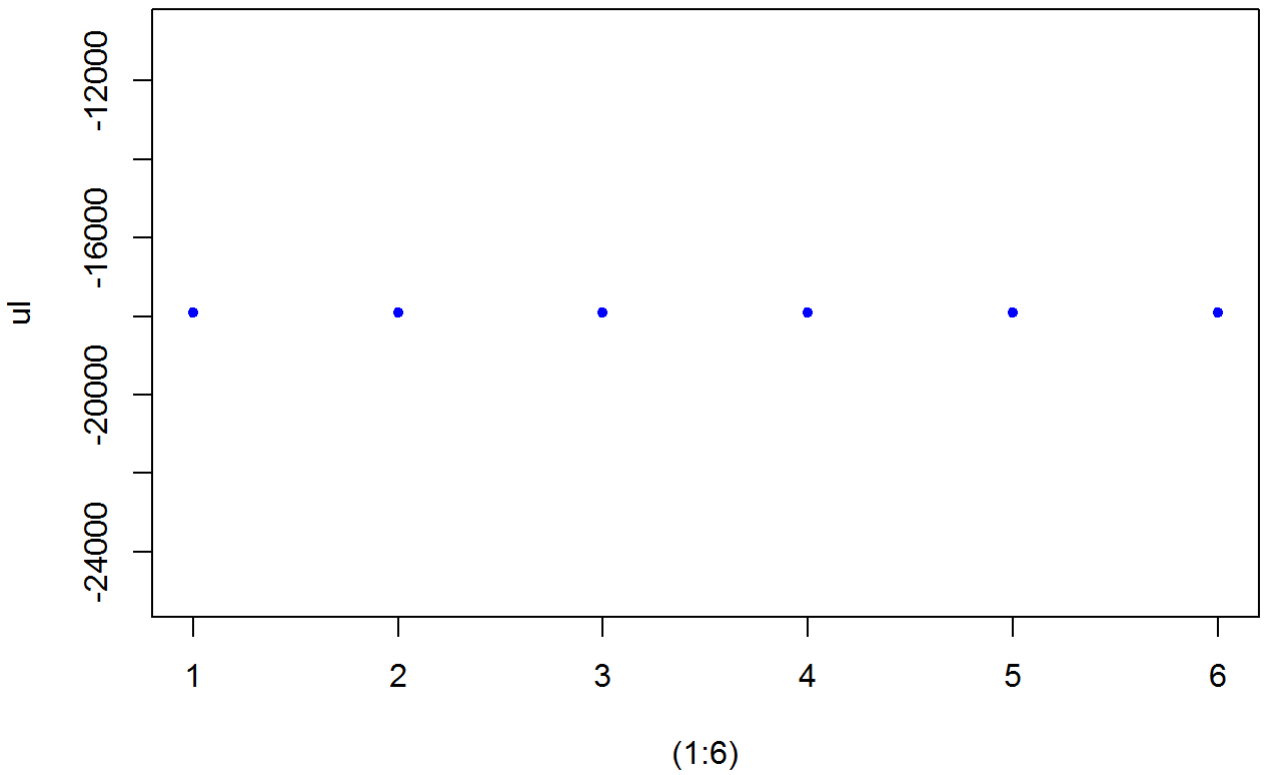
```
plot((1:6),uk, col = "blue", pch = 20)
```




```
# sample 10000 prob
ul
```

```
## [1] -17917.59 -17917.59 -17917.59 -17917.59 -17917.59 -17917.59
```

```
plot((1:6),ul, col = "blue", pch = 20)
```




```
# The observations made are,when i selected the data from sample the chance of getting the values is equally likely among th
e sample of 50, when increased the sample size the chance maintained in and around the same for the sampling method - frequn
cy method

# where as when i have the probalisitic method , with the increase in size the probability started going down a lot. so the
 probability of getting 1 in 10000 dices is not as same in sampling frequency method approach.
```