A4_python

Son Min Sang

2021-01-04

차 례

page 11

```
from datetime import datetime
import numpy as np
np.random.seed(1234) #fix the random seed
N= 10**3
x = np.random.rand(N, 1)*2-1 # runif() generates U(0,1)
y = np.random.rand(N, 1)*2-1 # runif() generates U(0,1)
t = np.sqrt(x**2+y**2)
cbind=np.concatenate((x, y, t), axis=1)
print(cbind[:5]) #always display and check!
```

```
## [[-0.6169611  -0.19778718  0.64788947]
##  [ 0.24421754  0.8612288   0.8951856 ]
##  [-0.12454452  0.03067229  0.12826585]
##  [ 0.57071717  0.61916404  0.84207018]
##  [ 0.55995162  0.76354446  0.9468611 ]]
```

```
pi_hat = 4*np.sum(t <= 1)/N
print(pi_hat)
```

```
## 3.06
```

page 12

```
from datetime import datetime
import numpy as np
beg_time = datetime.now()
np.random.seed(1234)
N= 10**6

x = np.random.rand(N, 1)*2-1
y = np.random.rand(N, 1)*2-1
t = np.sqrt(x**2+y**2)

pi_hat = 4*np.sum(t <= 1)/N

end_time = datetime.now()
print(end_time - beg_time)
```

```
## 0:00:00.131650
```

```
from datetime import datetime
import numpy as np

beg_time = datetime.now()
np.random.seed(1234)
N= 10**6
count =0

for i in range(N):
    x_i = np.random.rand(1)*2-1
    y_i = np.random.rand(1)*2-1
    t_i = np.sqrt(x_i**2+y_i**2)
    if t_i<=1:
        count+=1

pi_hat=4*count/N
end_time = datetime.now()

print(end_time - beg_time)
```

```
## 0:00:10.734443
```

page 13

```
import numpy as np

def pi_simulator(N):
  np.random.seed(1234)
  x = np.random.rand(N,1)*2-1
  y = np.random.rand(N,1)*2-1
  t = np.sqrt(x**2+y**2)
  pi_hat=4*np.sum(t <= 1)/N
  return pi_hat
```

```
pi_simulator(100)
```

```
## 2.96
```

```
pi_simulator(1000)
```

```
## 3.06
```

```
pi_simulator(10000)
```

```
## 3.1352
```

```
pi_simulator(100000)
```

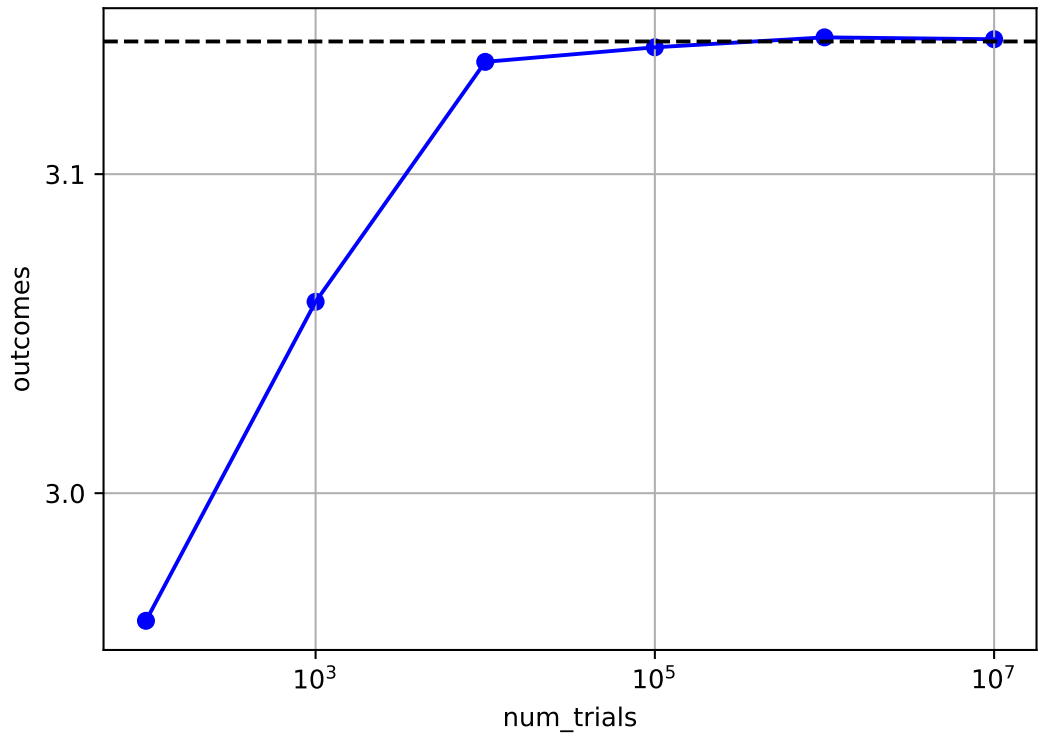
```
## 3.13976
```

```
import numpy as np

def pi_simulator(N):
  np.random.seed(1234)
  x = np.random.rand(N,1)*2-1
  y = np.random.rand(N,1)*2-1
  t = np.sqrt(x**2+y**2)
  pi_hat=4*np.sum(t <= 1)/N
  return pi_hat
num_trials = [10**i for i in range(2,8)]
outcomes=list(map(pi_simulator, num_trials))
results = dict(zip(num_trials,outcomes))
print(results)
```

```
## {100: 2.96, 1000: 3.06, 10000: 3.1352, 100000: 3.13976, 1000000: 3.142876, 10000000: 3.1422888}
```

page 15

```
import numpy as np
from datetime import datetime
import matplotlib.pyplot as plt

plt.scatter(num_trials,outcomes, c='blue')
plt.plot(num_trials,outcomes, c='blue')
plt.axhline(3.14159,0,1,color='black',linestyle='--')
plt.xscale('log')
plt.grid(True,axis='both')
plt.xlabel('num_trials')
plt.ylabel('outcomes')
plt.rc('font', size=25)
plt.show()
```



page 17

```
import numpy as np
from datetime import datetime
def pi_simulator2(N): # name change
  beg_time= datetime.now() # newly added
  np.random.seed(1234)
  x = np.random.rand(N,1)*2-1
  y = np.random.rand(N,1)*2-1
  t = np.sqrt(x**2+y**2)
  pi_hat=4*np.sum(t <= 1)/N
  end_time=datetime.now() # newly added
  print(N)
  print(end_time-beg_time) # newly added
  return pi_hat
num_trials = [10**i for i in range(2,8)]
list(map(pi_simulator2, num_trials))
```

```
## 100
## 0:00:00
## 1000
## 0:00:00
## 10000
## 0:00:00
## 100000
## 0:00:00.002992
## 1000000
## 0:00:00.062339
## 10000000
## 0:00:00.607213
## [2.96, 3.06, 3.1352, 3.13976, 3.142876, 3.1422888]
```

page 22

```
import numpy as np
import scipy.stats as st
from datetime import datetime
def pi_simulator3(N):#name change
  #np.random.seed(1234) #seed must not be fixed
  x = np.random.rand(N,1)*2-1
  y = np.random.rand(N,1)*2-1
  t = np.sqrt(x**2+y**2)
  pi_hat=4*np.sum(t <= 1)/N
  return pi_hat
n = 100 # number of experiments to repeat
MC_N = 1000 # number of simulation repetition in a single experiment
np.random.seed(1234)
samples = list(range(0,n))
for i in range(n): #create an empty Zero vector
  samples[i] = pi_simulator3(MC_N) # do this for MC_N times
print(samples[:5])
```

```
## [3.06, 3.184, 3.12, 3.228, 3.124]
```

page 23

```
X_bar = np.mean(samples)
s = np.sqrt(sum((X_bar-samples)**2)/(n-1))
t= st.t.ppf(0.975, n-1)

print("X_bar :",X_bar)
```

```
## X_bar : 3.1412000000000004
```

```
print("s :",s)
```

```
## s : 0.05271305973538881
```

```
print("t :",t)
```

```
## t : 1.9842169515086827
```

page 24

```
n = 100 # number of experiments to repeat
MC_N = 1000 # number of simulation repetition in a single experiment
np.random.seed(1234)
samples = list(range(0,n))
for i in range(n):
  samples[i] = pi_simulator3(MC_N)
X_bar = np.mean(samples)
s = np.sqrt(sum((X_bar-samples)**2)/(n-1))
t= st.t.ppf(0.975, n-1)
lb = X_bar-t*s/np.sqrt(n) # lower bound
ub = X_bar+t*s/np.sqrt(n) # upper bound
```

page 25

```
n = 1000 # number of experiments to repeat
MC_N = 10000 # number of simulation repetition in a single experiment
np.random.seed(1234)
samples = list(range(0,n))
for i in range(n):
  samples[i] = pi_simulator3(MC_N)
X_bar = np.mean(samples)
s = np.sqrt(sum((X_bar-samples)**2)/(n-1))
t= st.t.ppf(0.975, n-1)

lb = X_bar-t*s/np.sqrt(n) # lower bound
ub = X_bar+t*s/np.sqrt(n) # upper bound
print("lb :",lb)
```

```
## lb : 3.141465340511527
```

```
print("ub :",ub)
```

```
## ub : 3.1434762594884726
```

```
print("ub-lb :",ub-lb)
```

```
## ub-lb : 0.002010918976945497
```