A4 python ver

Lee SungHo

2021-01-04

⚜ ⚜

page 11 : Implementation with 1000 repetitions

```
import numpy as np

np.random.seed(1234) # fix the random seed
N = 10**3
x = np.random.rand(N,1)*2 - 1
y = np.random.rand(N,1)*2 - 1
t = np.sqrt(x**2+y**2)

df = np.concatenate((x, y, t), axis=1) # always display and check!
print(df[0:6])
```

```
## [[-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 ]
##  [-0.45481479  0.52533556  0.69486254]]
```

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

```
## 3.06
```

page 12 : From thre previous slide

```
import numpy as np
import time

beg_time = time.time()
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)

df = np.concatenate((x, y, t), axis=1)

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

end_time = time.time()
print('Time difference of', end_time - beg_time,'secs')
```

```
## Time difference of 0.09773635864257812 secs
```

page 12 : From thre previous slide

```
import numpy as np
import time

beg_time = time.time()
np.random.seed(1234)
N = 10**6
count = 0
for i in range(N):
  x_i = np.random.rand()*2 - 1
  y_i = np.random.rand()*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 = time.time()
print('Time difference of', end_time - beg_time,'secs')
```

```
## Time difference of 2.3734118938446045 secs
```

page 13 : Approach with a customer function

```
import numpy as np
import time

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
```

page 13 : Approach with a customer function

```
num_trials = [10**i for i in range(2,8)]
outcomes = [pi_simulator(i) for i in num_trials]

result = dict(zip(num_trials,outcomes))
result
```

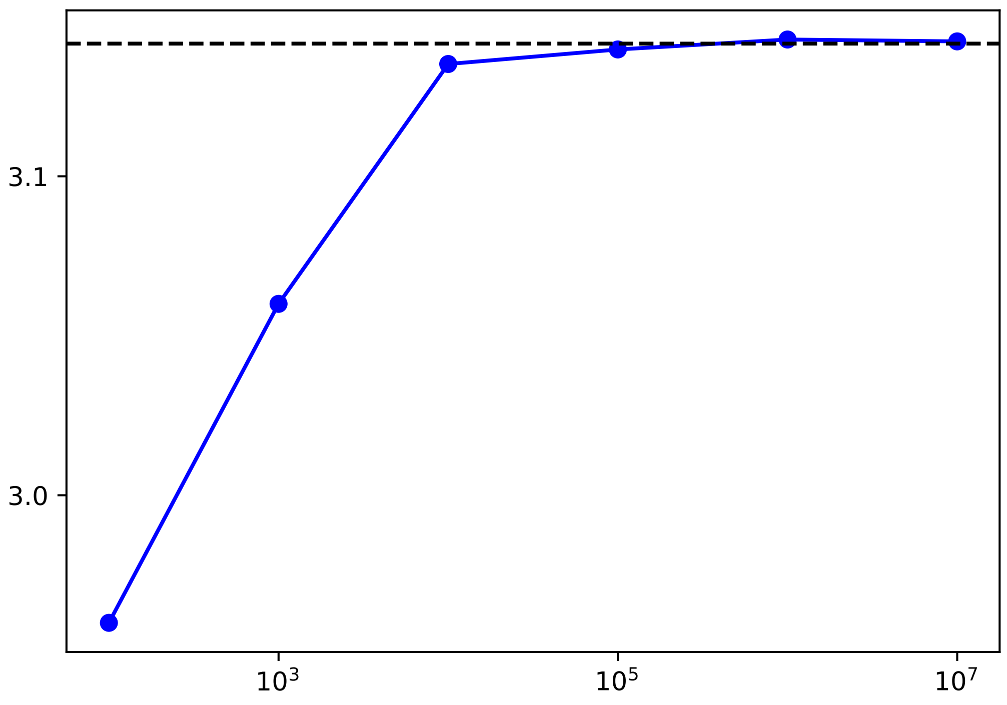
```
## {100: 2.96, 1000: 3.06, 10000: 3.1352, 100000: 3.13976, 1000000: 3.142876, 10000000: 3.1422888}
```

page 15

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


plt.plot(result.keys(), result.values(), c='blue',  marker='o')
plt.axhline(3.14159, 0, 1, color='black', linestyle='--')
plt.xscale('log')
plt.xlabel=('num_trials')
plt.ylabel=('outcomes')
plt.rc('font',size=25)

plt.show()
```



page 17

```
import numpy as np
import time

def pi_simulator2(N):
  beg_time = time.time() #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 = time.time() #newly added

  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))
```

```
## 0.0009968280792236328
## 0.0
## 0.0
## 0.002991914749145508
## 0.03684091567993164
## 0.35669827461242676
## [2.96, 3.06, 3.1352, 3.13976, 3.142876, 3.1422888]
```

page 22

```python
import numpy as np

import time


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 = []


for i in range(n):

  samples.append(pi_simulator3(MC_N))


print(samples[:5])
```

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

page 23

```
import scipy.stats as st

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 :  0.052713059735388881
```

```
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

```python
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
```

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

```
## ub : 3.1434762594884726
```

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

```
## ub-lb : 0.0020109189769454497
```