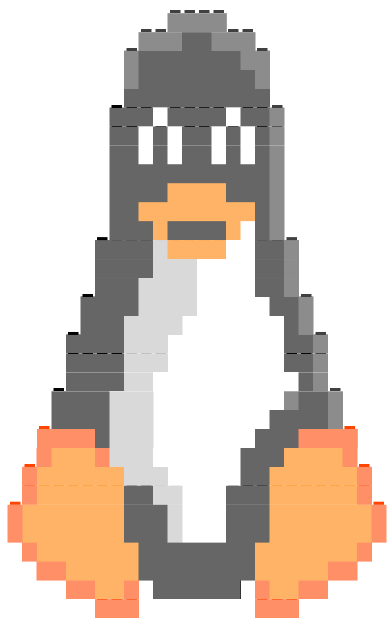
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΚΑΙ ΕΦΑΡΜΟΣΜΕΝΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΦΥΣΙΚΗΣ

ΦΥΣ 140 Εισαγωγή στην Επιστημονική Χρήση Υπολογιστών
Χειμερινό Εξάμηνο 2023

Φώτης Πτωχός και Αλέξανδρος Αττίκης

Φροντιστήριο 12

28 Νοεμβρίου 2023
15:00 - 17:00



Φροντιστήριο 12

Παράδειγμα 1 Στο παράδειγμα αυτό θα εφαρμόσουμε τη μέθοδο μετασχηματισμού Monte Carlo ώστε να κατασκευάσουμε 10^6 τυχαίους αριθμούς κατανεμημένους σύμφωνα με την πυκνότητα πιθανότητας $P(x) = e^x$. Οι τυχαίοι αριθμοί που κατασκευάζουμε θα πρέπει να έχουν τιμές στο διάστημα 1 έως 2. Θα πρέπει να κάνουμε το ιστόγραμμα των τυχαίων αριθμών που επιλέξαμε. Στη συγκεκριμένη άσκηση η πυκνότητα πιθανότητας αναφέρεται σε τιμές του x, όπου $x \in [a, b]$], και θα χρειαστεί να κανονικοποιηθεί πριν εφαρμόσουμε την μέθοδο. Η κανονικοποίηση επιτυγχάνεται ορίζοντας την συνάρτηση $P'(x)$ που ικανοποιεί τη σχέση:

$$P'(x) = \frac{P(x)}{\int\limits_a^b P(x)dx}. \tag{1}$$

Εφόσον το διάστημα των τιμών του y ξεκινά από την τιμή a, το ολοκλήρωμα για την εύρεση της συνάρτησης μετασχηματισμού θα είναι:

$$\text{CDF} = \int\limits_a^x P'(x)dx \tag{2}$$

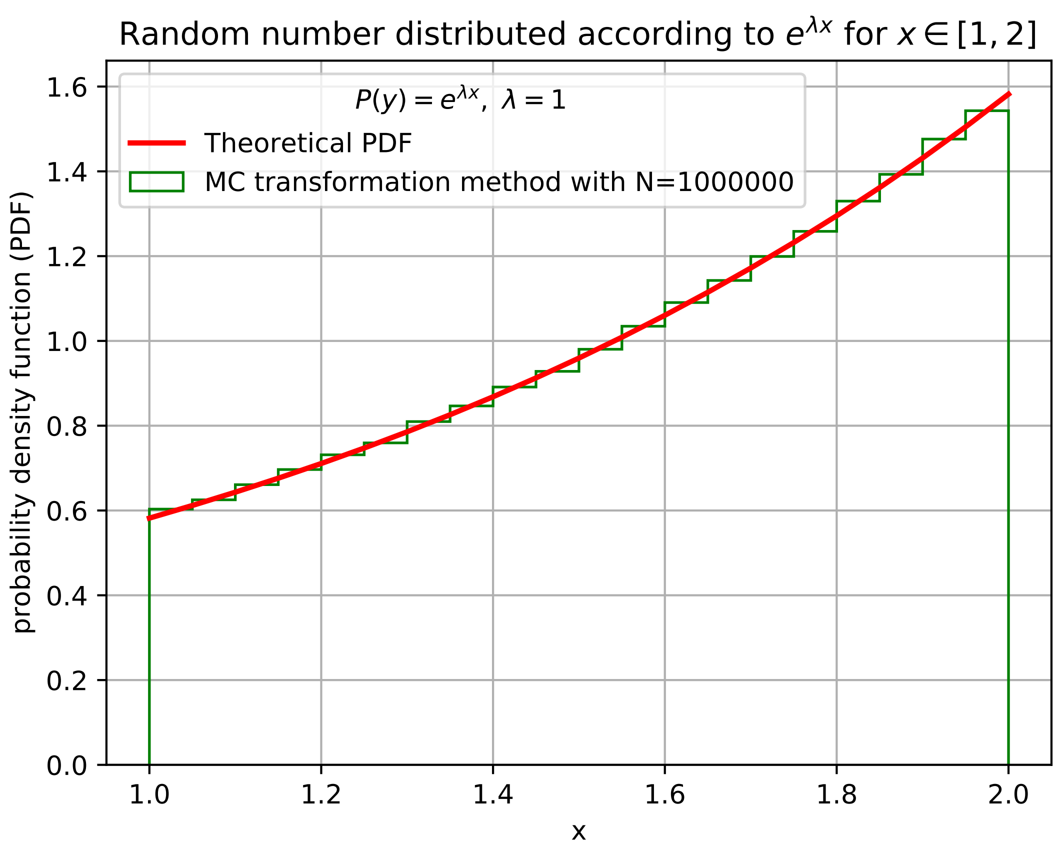
$$\Rightarrow y = \int\limits_a^x P'(x)dx \tag{3}$$

tutorial12/ex1.py

```python
#!/usr/root/python3
import numpy as np
import matplotlib.pyplot as plt
from random import seed, random, gauss

N = int(input("How many tries? "))
seed(12345678)

xList =[]
rate  = 1 # lambda
xMax  = 2
xMin  = 1
pdf   = []

# A * int^{xMax}_{xMin} exp(x) dx  = 1
# => A [exp(x)]^{xmax}_{xMin} = 1
# A [e^{x2}  - e^{x1}] = 1
# A = 1/[e^{x2} - e^{x1}]
norm  = 1/np.abs(np.exp(xMax) - np.exp(xMin))

for i in range(N):
    xList.append(np.log(np.exp(xMin) + random()/(rate*norm) ) )

```

```python
cont, xBins, intr=plt.hist(xList, bins=20, range=(1.,2.), density=True, histtype='step',color='g', label="MC transformation method with N=%d" % (N))
pdf = norm*np.exp( rate * xBins )

# Creat the plot
plt.plot(xBins, pdf, 'r-', lw = 2, label='Theoretical PDF')
plt.xlabel('x')
plt.ylabel('probability density function (PDF)')
plt.title(r'Random number distributed according to $e^{\lambda x}$ for $x\in[1, 2]$')
plt.grid(True)
plt.legend(title = r'$P(y)=e^{\lambda x},\ \lambda=1$')
for ext in [".png", ".pdf"]:
    plt.savefig(__file__.split(".")[0] + ext)
plt.show()
```

Αποτέλεσμα:



Δημιουργία τυχαίων αριθμών που κατανέμονται σύμφωνα με την εκθετική συνάρτηση e^{-bx} όπου $b > 0$.

Παράδειγμα 2 Ακόμα ένα παράδειγμα για τη Monte Carlo μέθοδο μετασχηματισμού. Χρησιμοποιώντας τη μέθοδο μετασχηματισμού, πρέπει να γράψουμε τον τρόπο με τον οποίο θα πάρουμε τυχαίους αριθμούς κατανεμημένους σύμφωνα με την:

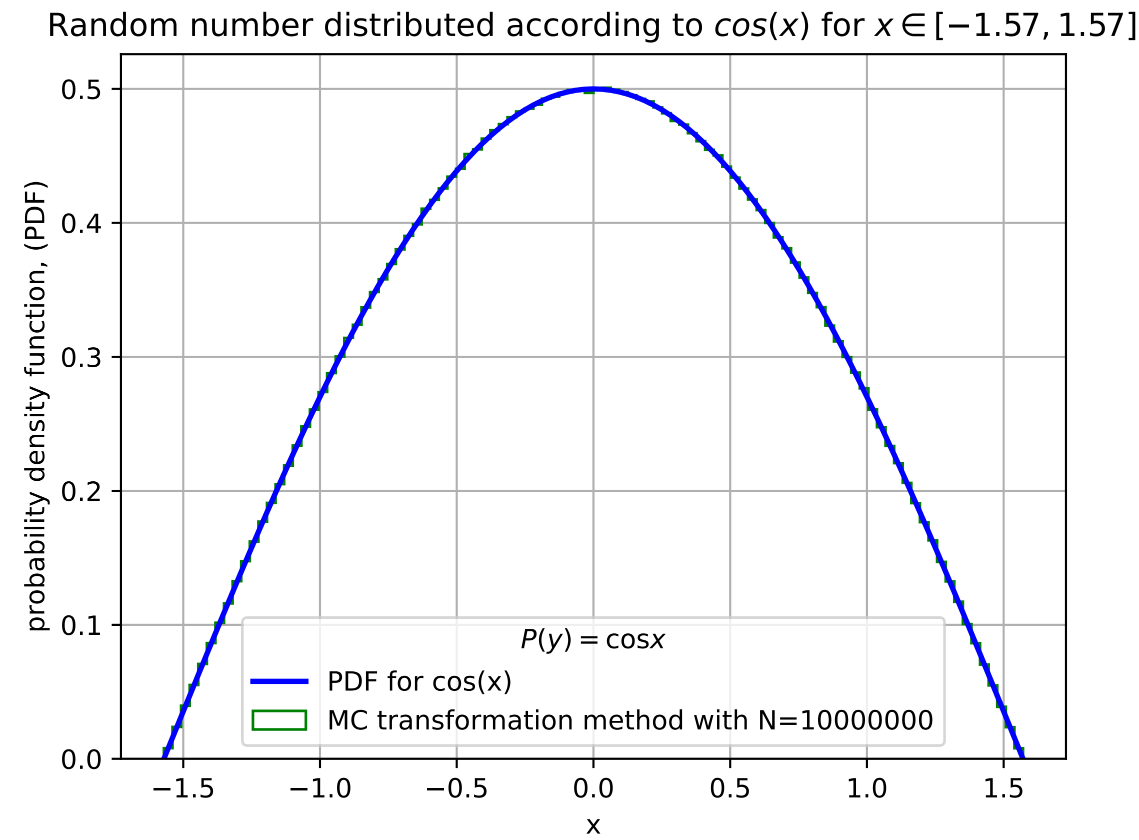
$$P(x) = \cos(x) \tag{4}$$

Ποιο διάστημα τιμών του x θα πρέπει να χρησιμοποιήσουμε;

tutorial12/ex2.py

```python
#!/usr/bin/python3
import numpy as np
from math import asin
import matplotlib.pyplot as plt
from random import seed, random

def PDF(x):
    return np.cos(x)

xMin = float(input("Give the lower value of the desired x-interval [-pi/2]: "))
xMax = float(input("Give the upper value of the desired x-interval [ pi/2]: "))
N    = int(input("How many random numbers to generate [100K]? "))

seed(123456)

yList = []
# A * int^{xMax}_{xMin} cos(x) dx  = 1
# => A [sin(x)]^{xmax}_{xMin} = 1
# A [sin(pi/2)  - sin(-pi/2)] = A [1- (-1)] = 1
# 2A = 1 => A=1/2
norm = (np.sin(xMax) - np.sin(xMin))

for itry in range(N):
    xv = random()             # random variable in interval [0,1)
    xv = norm*xv - 1          # sin(x) = 2y-1
    yList.append(asin(xv))     # x = arcsin(2y-1)

#plt.figure(figsize=(6,6))
cont, xBins, intr = plt.hist(yList, bins=100, range=(-np.pi/2,np.pi/2), density=True, histtype='step',color='g', label="MC transformation method with N=%d" % (N) )

pdf = PDF(xBins)/norm
plt.plot(xBins, pdf, 'b-', lw=2, label=r'PDF for cos(x)')
plt.xlabel('x')
plt.ylabel('probability density function, (PDF)')
plt.title(r'Random number distributed according to $cos(x)$ for $x \in [%.2f, %.2f]$' % (xMin, xMax) )
plt.grid(True)
plt.legend(title = r'$P(y)=\cos{x}$')
for ext in [".png", ".pdf"]:
    plt.savefig(__file__.split(".")[0] + ext)
plt.show()
```

Αποτέλεσμα :


Random number distributed according to $cos(x)$ for $x \in [-1.57, 1.57]$
probability density function, (PDF)
x
$P(y) = \cos x$
PDF for cos(x)
MC transformation method with N=10000000
0.0
0.1
0.2
0.3
0.4
0.5
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5

Παράδειγμα 3 Σε αυτό το παράδειγμα θα γράψουμε ένα πρόγραμμα το οποίο υπολογίζει την πιθανότητα ρίχνοντας 3 ζάρια να πάρουμε ακριβώς την ίδια τιμή σε 2 από αυτά:

tutorial12/ex3.py

```
#!/usr/bin/python3
'''
USAGE:
   chmod +x ex3.py
   python3 ex3.py
   script -q ex3.log python3 -i ex3.py
'''
import numpy as np
from random import seed, random, randint

seed(123456)

N = int(input("How many tries? "))
S = 0 # success

# Main simulation loop over N experiments
for itry in range(N):

    # Create a random number in [1,6]. Do that 3 times and store values in list
    rollList = [randint(1,6) for k in range(3)]  # simulate 3 rolls of the dice exactly 2/3 have same number
    matches  = 0 # will be used to keep track of the instances that we get

    # Check each pair of dice rolls for matches.
    for i in range(len(rollList)-1): # -1 because the nested loop will be i+1

        for j in range(i+1, len(rollList) ): # i+1 because external loop covers "i"->len-1

            # If two rolls have the same number, matches is incremented
            if rollList[i] == rollList[j]:
                matches +=1

    # Success only if exactly 1 match is found
    if matches == 1:
        S +=1

# Evaluate probabilities
pExpected   = (S/N)*100
pExperiment = (5/12)*100
msg="The probability of getting exactly two same dice when rolling 3 dice:"
msg+= "\n\tP(MC) = %4f" % (pExperiment)
msg+= "\n\tP(Theory) = %4f"  % (pExpected)
print(msg)
```

Αποτέλεσμα:

```
How many tries? 100000
The probability of getting exactly two dice with same face when rolling 3
   dice:
  P(MC) = 41.666667
  P(Theory) = 41.625000
```

Παράδειγμα 4 Υποθέστε ότι ένας περιπατητής κάνει βήματα μήκους ίσο με τη μονάδα (αυθαίρετες διαστάσεις) κατά μήκος μιας ευθείας μήκους 2ℓ. Τη χρονική στιγμή $t = 0$, ο περιπατητής ξεκινά από το σημείο $x = 0$, που βρίσκεται στο μέσο της ευθείας, και κάθε βήμα του μπορεί να είναι είτε προς τα δεξιά ή προς τα αριστερά (θετική ή αρνητική κατεύθυνση) με την ίδια πιθανότητα 0.5. Υπολογίστε το μέσο αριθμό βημάτων που χρειάζεται να κάνει ο περιπατητής για να βρεθεί έξω από την περιοχή $[-\ell, +\ell]$:

tutorial12/ex4.py

```
#!/usr/bin/python3
'''
USAGE:
   chmod +x ex4.py
   python3 ex4.py
   script -q ex4.log python3 -i ex4.py
'''
import numpy as np
import matplotlib.pyplot as plt
import random as rndm

ipower = int(input('Give the max power of 10 for MC simulations: '))
dist   = int(input('Give the maximum length (l): '))

rndm.seed(123456)
prob = 0.5 # probability for step left or right

# Repeat experiment for various number of N
for iw in range(ipower):
    mxwalks = 10**(iw+1)
    totSteps = 0

    # Do multiple walks so that we can get an average of steps needed to cover distance
    for iwalks in range(mxwalks):

        # Initialise number of steps taken and initial position
        nSteps = 0
        xPos   = 0

        # Walk until you get outside the region of interest (dist)
        while np.abs(xPos) < dist:
            if rndm.random() > prob:
                # update position with a step to the right
                xPos += 1
            else:
                # update position with a step to the left
                xPos -= 1

            # Increment number of steps taken
            nSteps += 1

        # Increment number of steps taken
        totSteps += nSteps
```

```python

    # Evaluate the average steps taken
    aveSteps = totSteps/float(mxwalks)

    if iw == 0 :
        print('%10s  %10s' % ('Ntries', '<Steps>'))
    print(" %8d  %15.6f" % (mxwalks, aveSteps))
```

Αποτέλεσμα:

```
Give the max power of 10 for MC simulations: 5
Give the maximum length (l): 3
    Ntries     <Steps>
        10        9.200000
       100        7.880000
      1000        9.352000
     10000        8.979200
    100000        8.981820
```

Στην περίπτωση αυτή, μία δοκιμή είναι ένα περίπατος μέχρις ότου ο περιπατητής βρεθεί εκτός των ορίων της ευθείας. Θα πρέπει να μετρήσουμε για την περίπτωση αυτή πόσα βήματα χρειάστηκαν ώστε να βγει εκτός της ευθείας. Θα πρέπει να κάνουμε ένα μεγάλο τέτοιων δοκιμών και κάθε φορά να μετράτε τον αριθμό των βημάτων που χρειάστηκαν για να βγει εκτός της ευθείας. Για να βρείτε τον μέσο αριθμό βημάτων, θα πρέπει να εκτελέσετε πολλά πειράματα και να βρείτε τον μέσο όρο των βημάτων που χρειάστηκαν για όλα τα πειράματα.