Υπολογιστική Φυσική

Ακαδημαϊκό έτος 2022-2023
ΠΜΣ Φυσικής, ΕΚΠΑ

Ανδρέας-Νάουμαν Θάμπετ
Α.Μ. : 7110112200207

Κώδικας 1
Η διαδικασία του τυχαίου τηλεγράφου

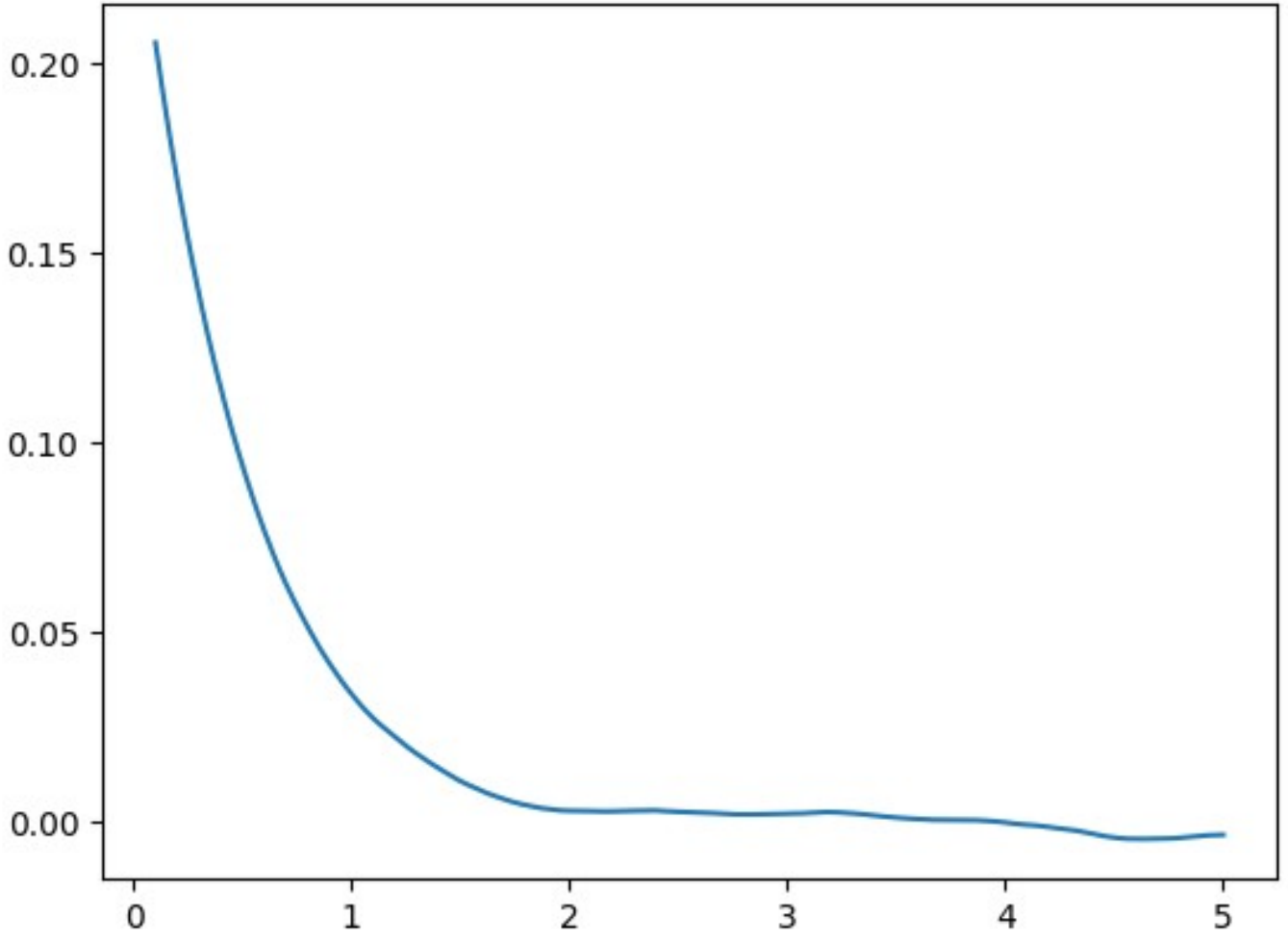
Αποτελεί την πιο απλή διαδικασία Markov με δύο δυνατές καταστάσεις n=0 και n=1.

Το προβλεπόμενο στάσιμο όριο για την μέση τιμή είναι α/(α+β), όπου α ο ρυθμός να πάμε από την κατάσταση 0 στην 1 και β ο ρυθμός να πάμε από την κατάσταση 1 στην 0, ενώ η συνάρτηση αυτοσυσχέτισης θα πρέπει να έχει εκθετική μορφή.

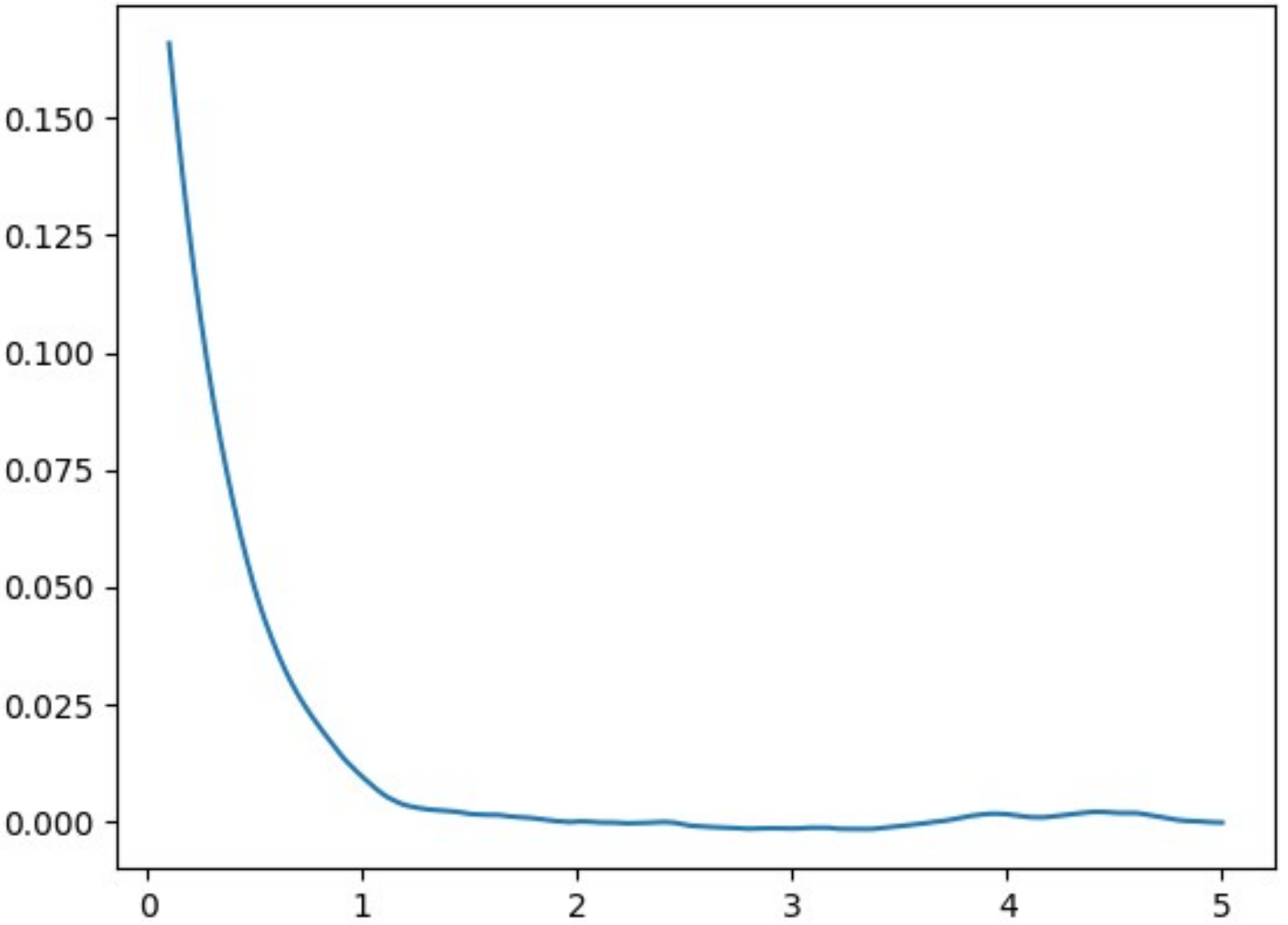
```python
import random
import numpy as np
import matplotlib.pyplot as plt
def exp_dist(b,r):
    return -np.log(1-r)/b
N=10000
a=1
b=1
sum_odd=0; sum_even=0
t=np.zeros(N)
t[0]=0
dt=np.zeros(N)
for m in range(N//2-1):
    r1=random.random()
    r2=random.random()
    t[2*m+1]=t[2*m]+exp_dist(b,r1)
    t[2*m+2]=t[2*m+1]+exp_dist(a,r2)
    dt[2*m+1]=t[2*m+1]-t[2*m]
    dt[2*m+2]=t[2*m+2]-t[2*m+1]
    sum_odd+=dt[2*m+1]
    sum_even+=dt[2*m+2]
mean= sum_odd/(sum_even+sum_odd)
print(mean)
print(((mean-a/(a+b))/a*(a+b))*100)
def kapa(l):
    sum=0
    for m in range(N//2-1):
        a=t[2*m]
        b=t[2*m+1]
        for k in range(m,N//2-1):
            c=t[2*k]-l
            d=t[2*k+1]-l
            if b>c:
                if a<d:

                    if c<=a:
                        if d<=b:
                            dias=d-a
                        else:
                            dias=b-a
                    else:
                        if d<=b:
                            dias=d-c
                        else:
                            dias=b-c
                    sum+=dias
                else:
                    sum+=0
            else:
                break
        mean2=sum/(sum_even+sum_odd)
    return mean2-mean**2

tau_values = np.linspace(0.1, 5, 1000)
k_values = [kapa(tau) for tau in tau_values]
plt.plot(tau_values, k_values)
plt.show()
```

```
6   N=10000
7   a=1
8   b=1
```

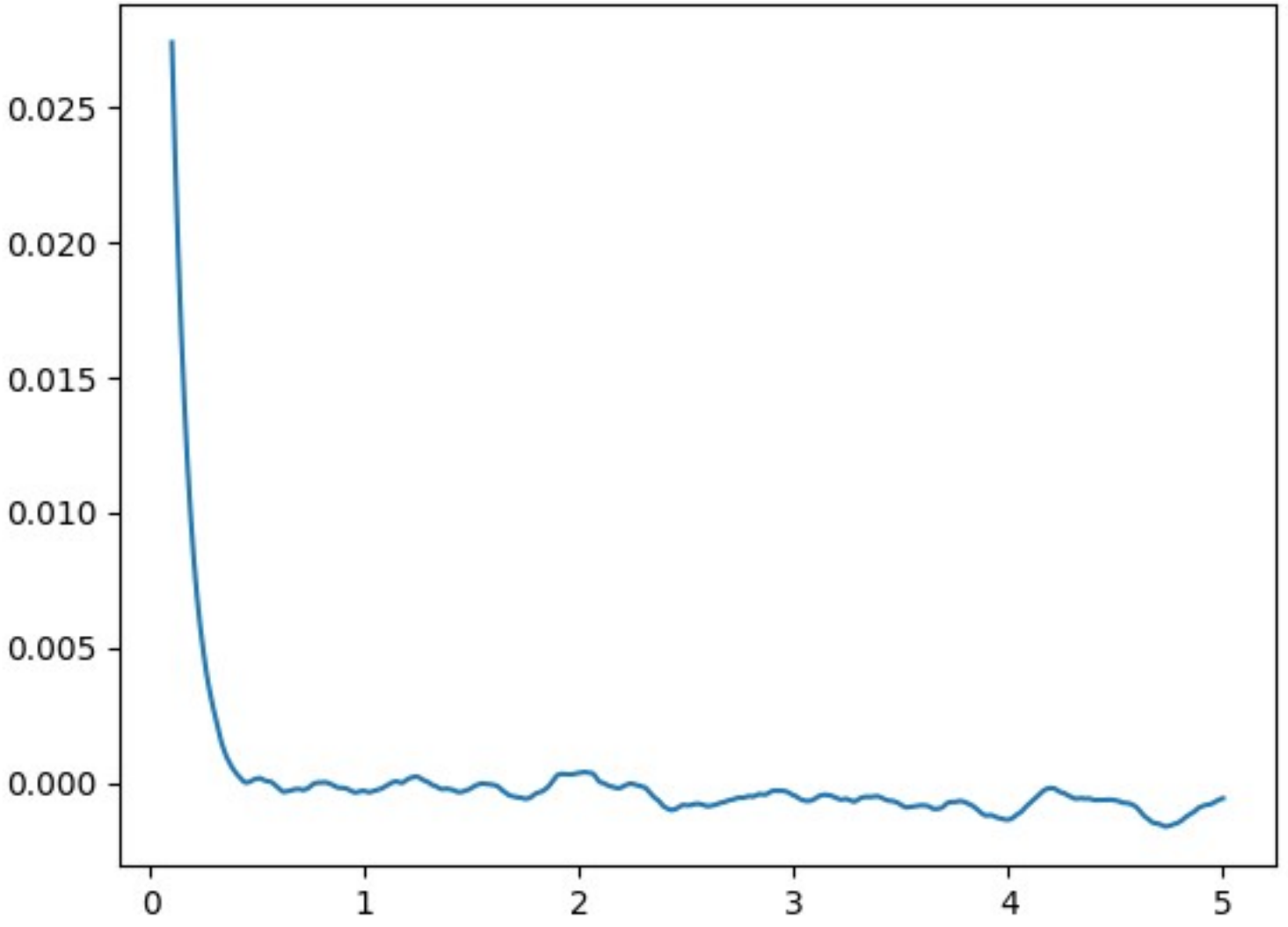
```
0.5012371319077883
0.24742638155765828
```



```
N=10000
a=2
b=1
0.6639721471009461
-0.40417793485807985
```

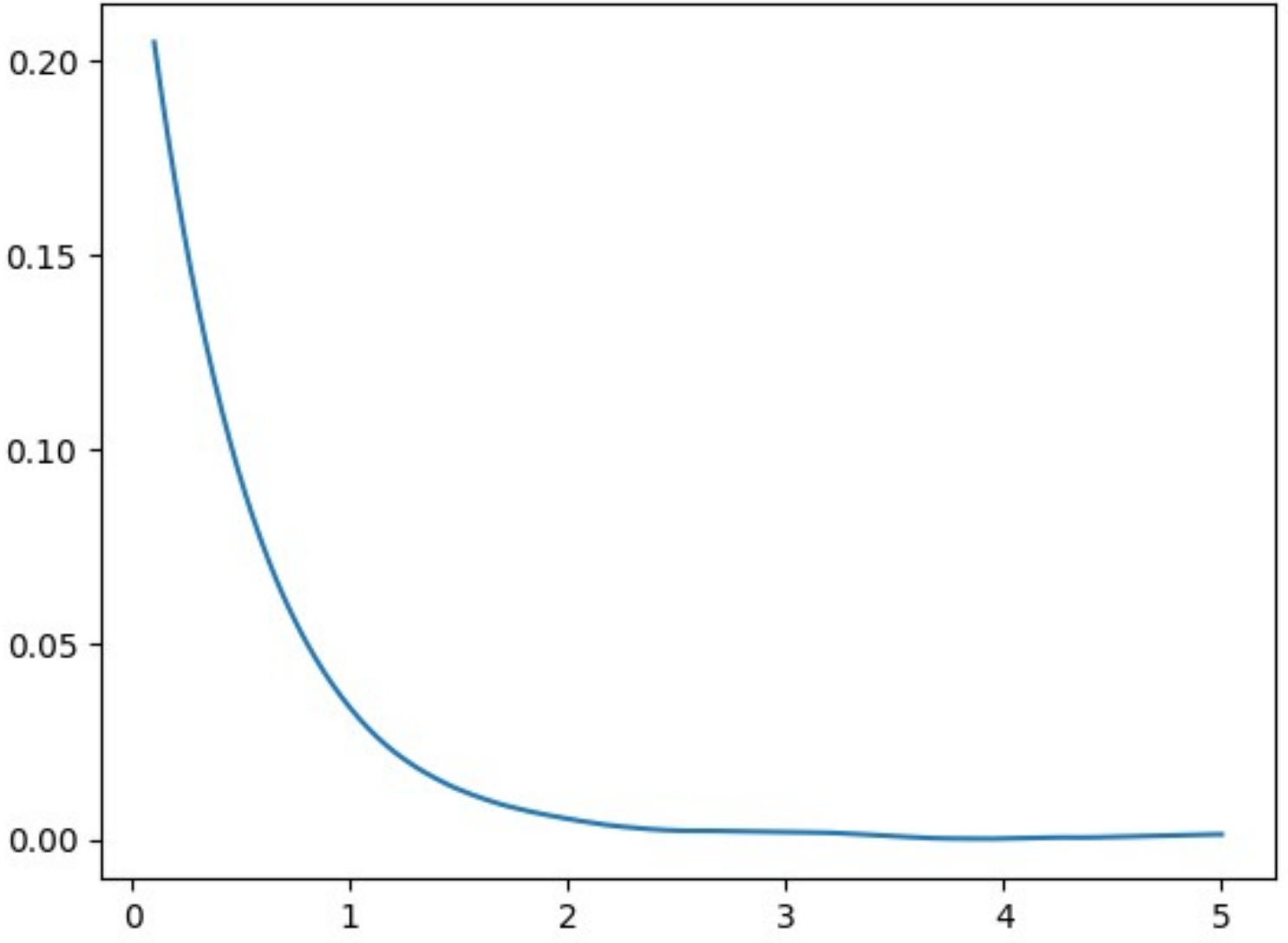


```
N=10000
a=10
b=1
0.9085510779153877
-0.05938142930735024
```



```
6   N=100000
7   a=1
8   b=1
```

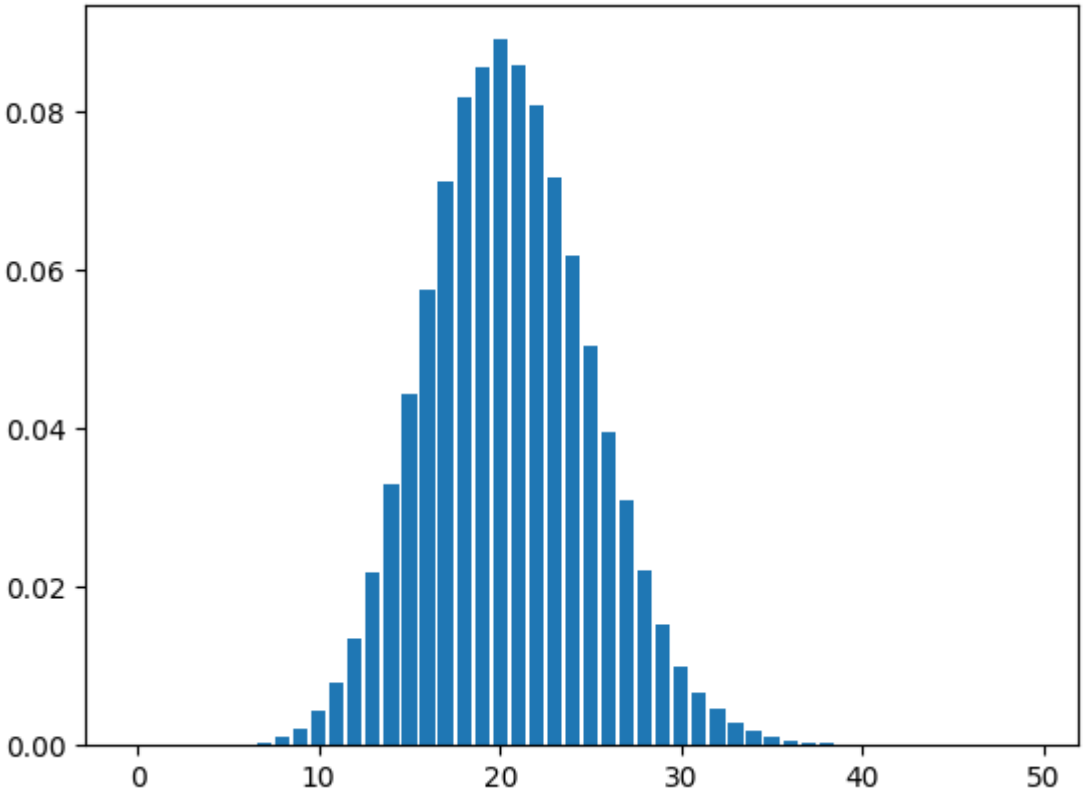
```
0.5002155699271722
0.04311398543443978
```



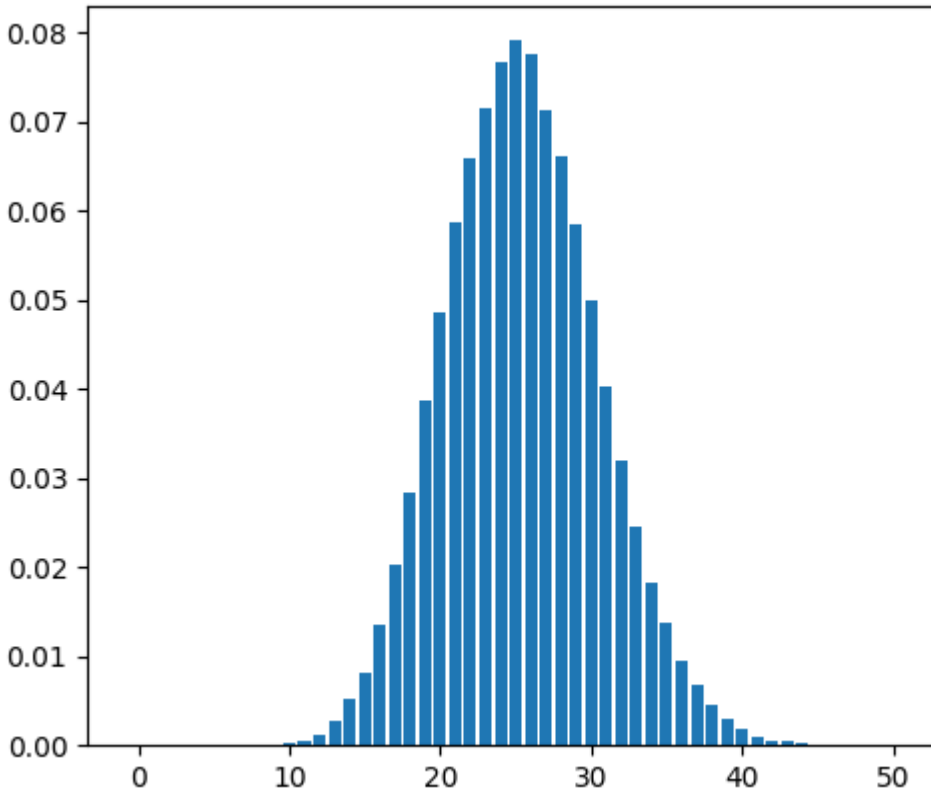
Κώδικας 2
Μεταβάσεις ατόμου/μορίου μεταξύ δύο καταστάσεων A/X (A “σταθερό”)

Θεωρούμε ότι η πιθανότητα μετάβασης είναι ανάλογη του πληθυσμού των ατόμων στην αντίστοιχη αρχική κατάσταση. Η προβλεπόμενη κατανομή για τον πληθυσμό στην X κατάσταση ακολουθεί την κατανομή Poisson με μέση τιμή ασυμπτωματικά τον πληθυσμό A.

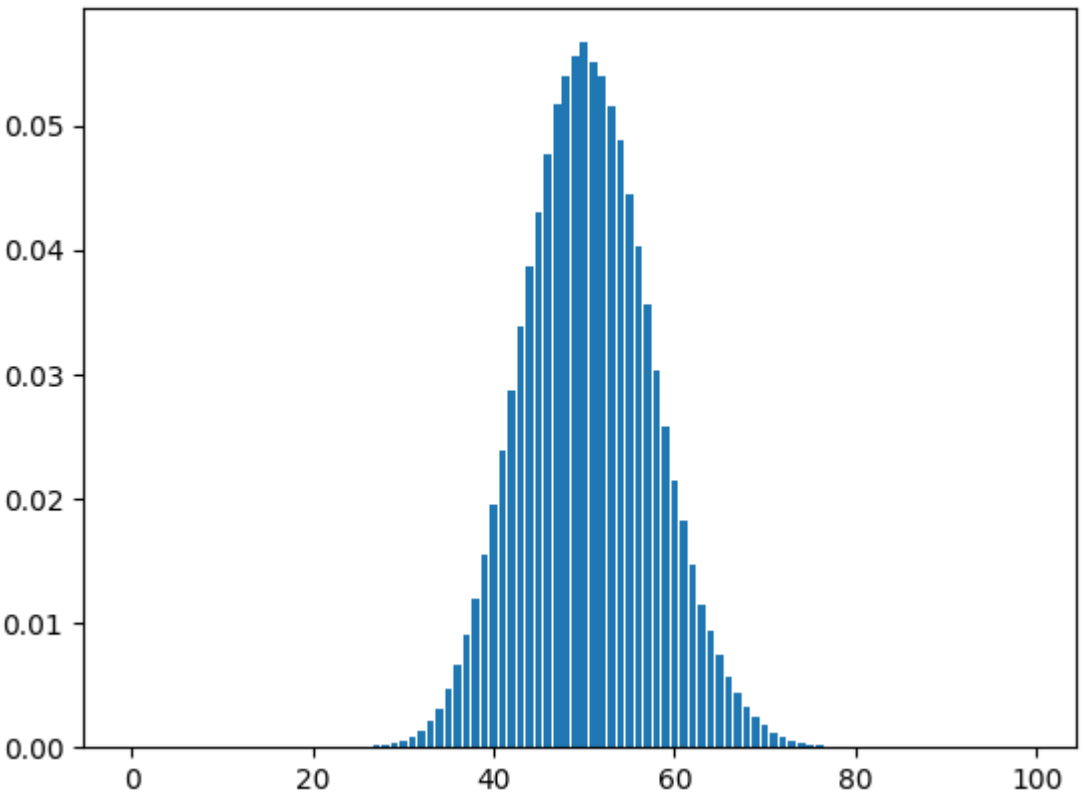
```python
import random
import numpy as np
import matplotlib.pyplot as plt
t=0
A=20
T=10000
N=A
k=1
W=k*(N+A)
tn=np.zeros(50)
def Dt(W,r):
    return -np.log(1-r)/W
while t<T:
    r1=random.random()
    r2=random.random()
    l= Dt(W,r1)
    t+= l
    tn[N]= tn[N]+l
    ratio=A/(A+N)
    if r2>ratio:
        N=N-1
    else:
        N=N+1
plt.bar(range(0,50),tn/t)
plt.show()
```



```
A=25
T=10000
```



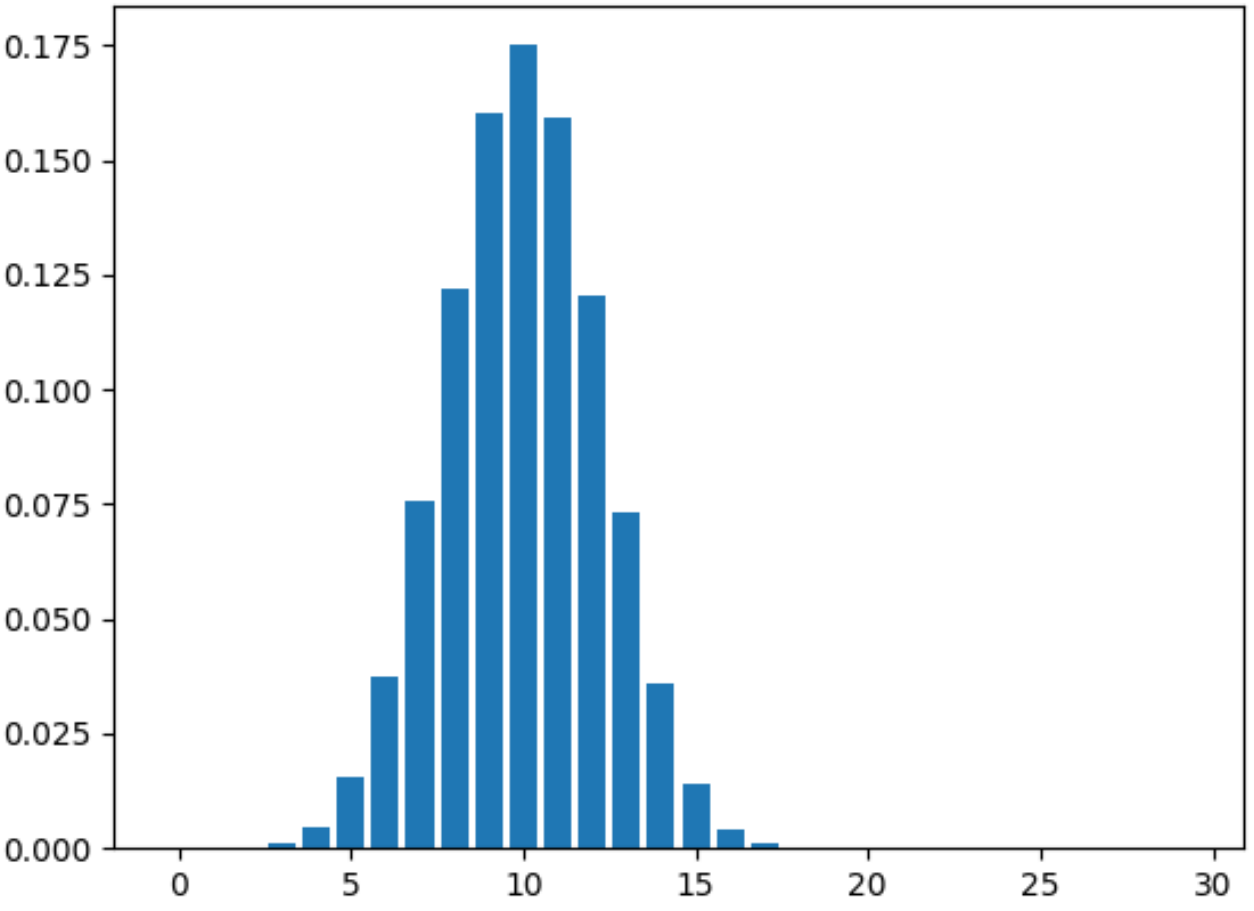

```
A=50
T=10000
```



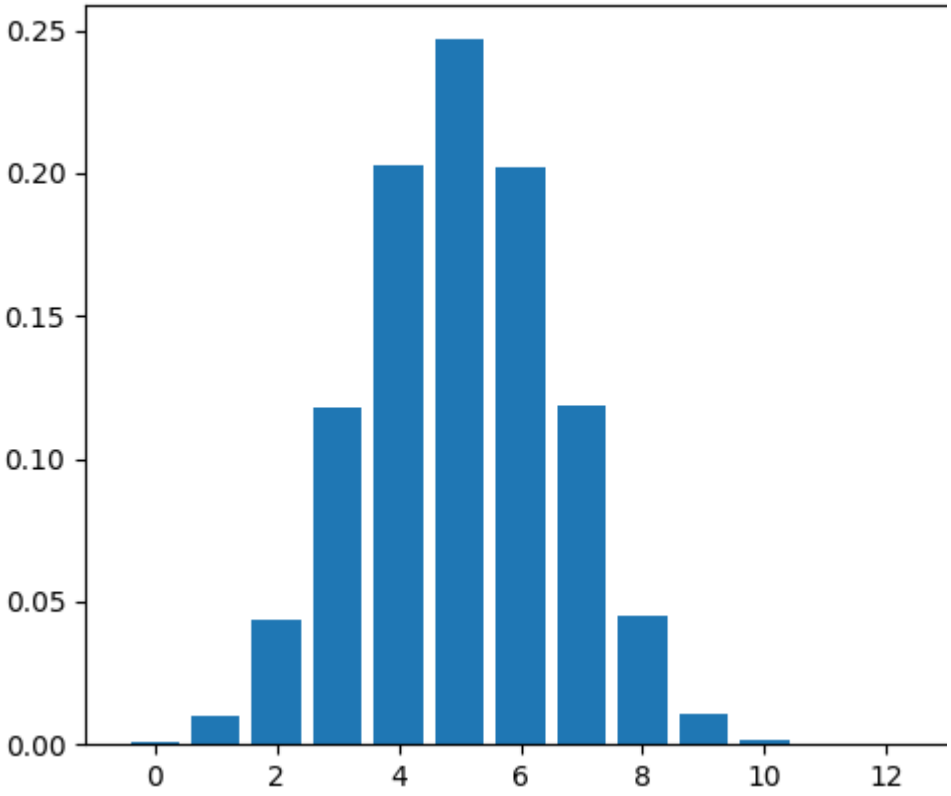
Κώδικας 3
Μεταβάσεις ατόμου/μορίου μεταξύ δύο καταστάσεων A/X (A μεταβλητό)

Σε αυτήν την περίπτωση, η προβλεπόμενη κατανομή προκύπτει ότι είναι διωνυμική με μέση τιμή A/2.

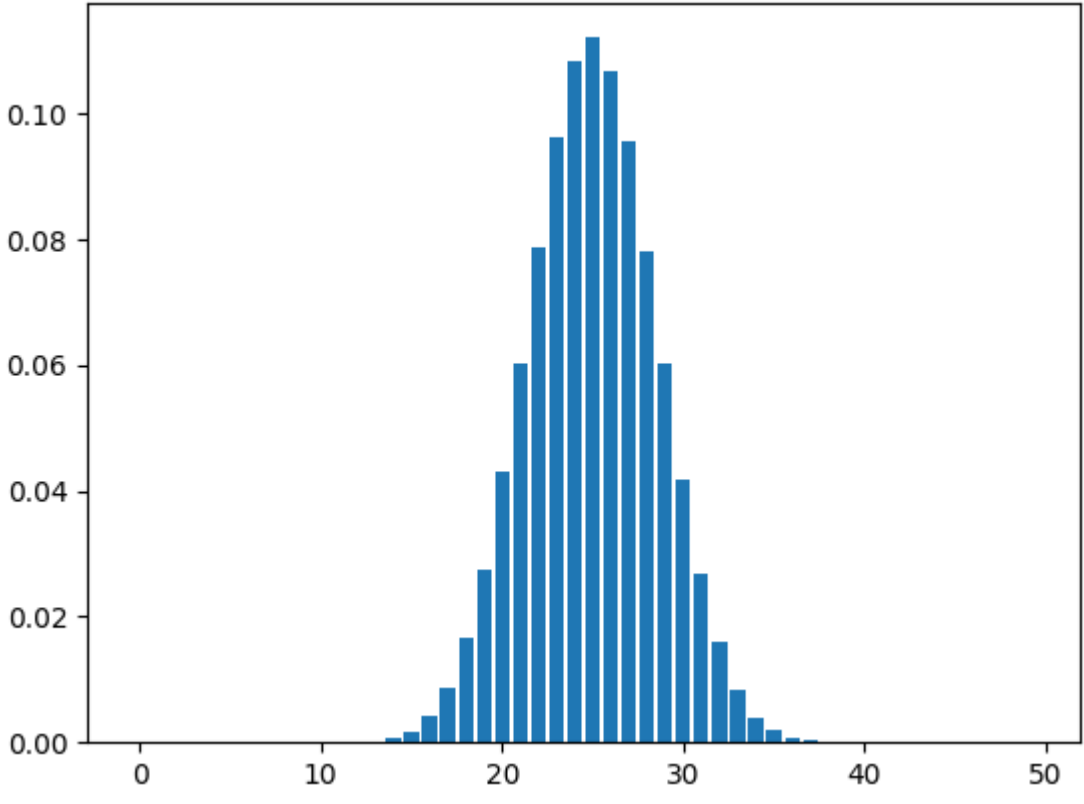
```python
import random
import numpy as np
import matplotlib.pyplot as plt
t=0
A=20
N=0
T=10000
k=1
W=k*(N+A)
tn=np.zeros(30)
def Dt(W,r):
    return -np.log(1-r)/W
while t<T:
    r1=random.random()
    r2=random.random()
    l= Dt(W,r1)
    t+= l
    tn[N]= tn[N]+l
    ratio=A/(A+N)
    if r2>ratio:
        N=N-1
        A=A+1
    else:
        N=N+1
        A=A-1
plt.bar(range(0,30),tn/t)
plt.show()
```



```
A=10
N=0
T=10000
```




```
A=50
N=0
T=10000
```

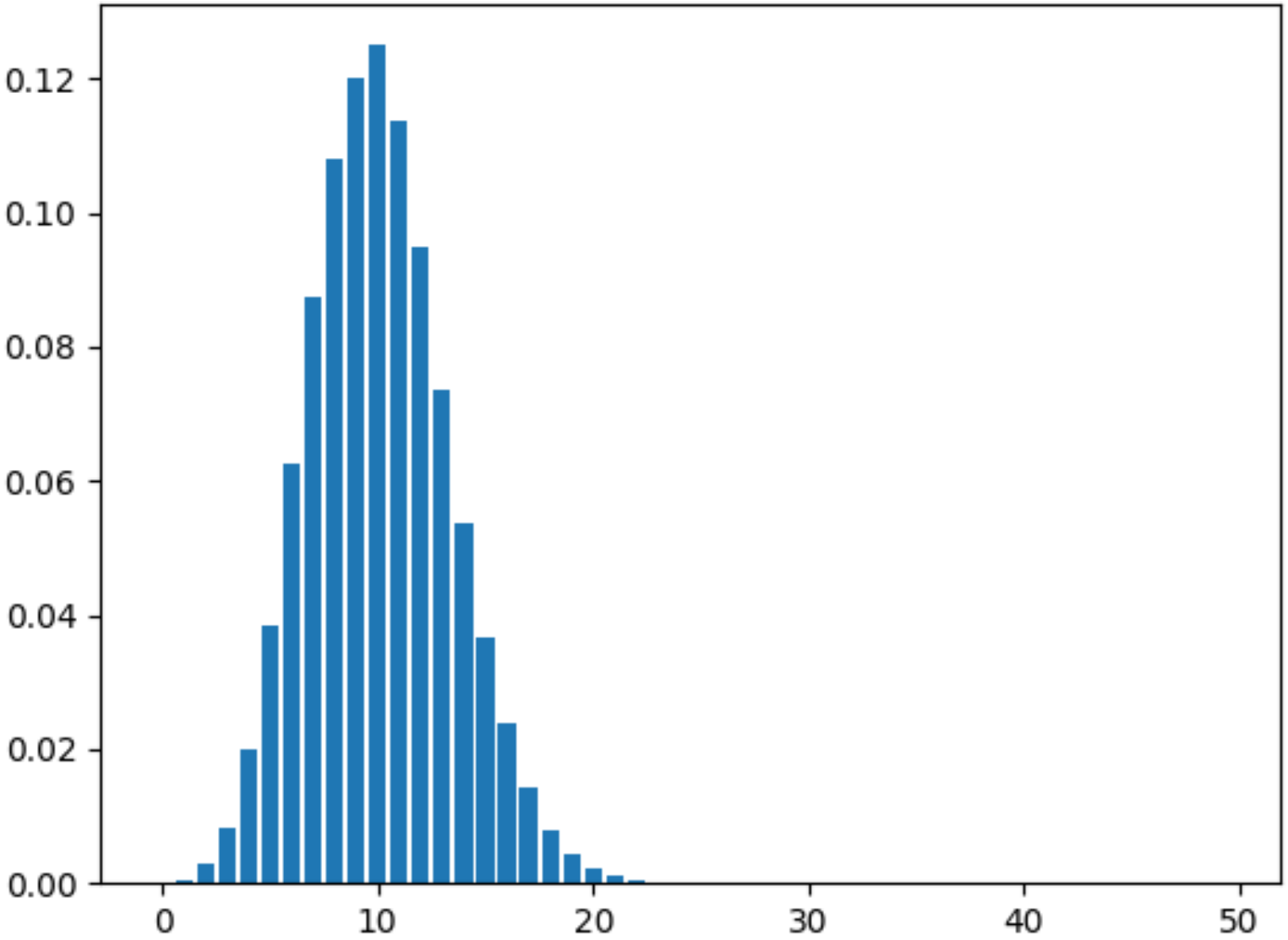


Κώδικας 4
Αντίδραση Michelson-Menten

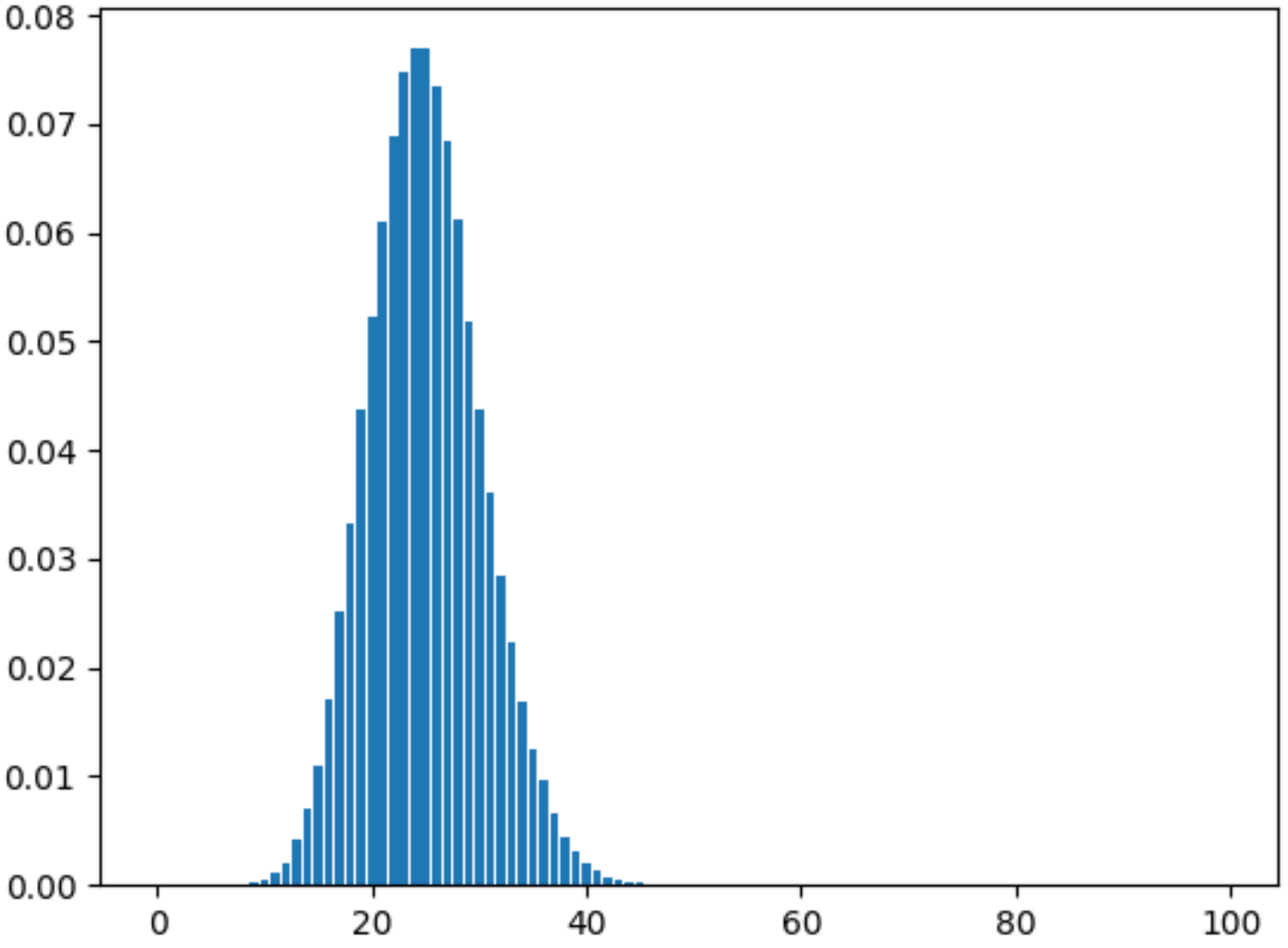
Περιγράφει την εξέλιξη του πληθυσμού του προϊόντος Χ και την κατάσταση του καταλύτη s. Όταν ο καταλύτης είναι ενεργός επιτρέπεται η μετάβαση από το Α στο Χ, ενώ ανεξάρτητα της καταστάσεως του ενζύμου/καταλύτη διασπάται αυθόρμητα το Χ στο Α. Η μέση τιμή προβλέπεται στα Α*α/(α+β), όπου α είναι ο ρυθμός ενεργοποίησης του καταλύτη και β ο ρυθμός απενεργοποίησης αντίστοιχα.

```python
import random
import numpy as np
import matplotlib.pyplot as plt
t=0
A=20
N=A
T=10000
b=100
a=100
k=1
s=1
W=k*(N+A)+b
tn=np.zeros(50)
def Dt(W,r):
    return -np.log(1-r)/W
while t<T:
    r1=random.random()
    r2=random.random()
    l= Dt(W,r1)
    t+= l
    tn[N]= tn[N]+l
    R=r2*W
    R=R-k*N
    if R<0:
        N=N-1
        W=W-k
        continue
    if s==0:
            s=1
            W=W+k*A+b-a
    else:
            R=R-k*A
            if R<0:
                N=N+1
                W=W+k
            else:
                s=0
                W=W-k*A+a-b
    print(W)

plt.bar(range(0,50),tn/t)
plt.show()
```

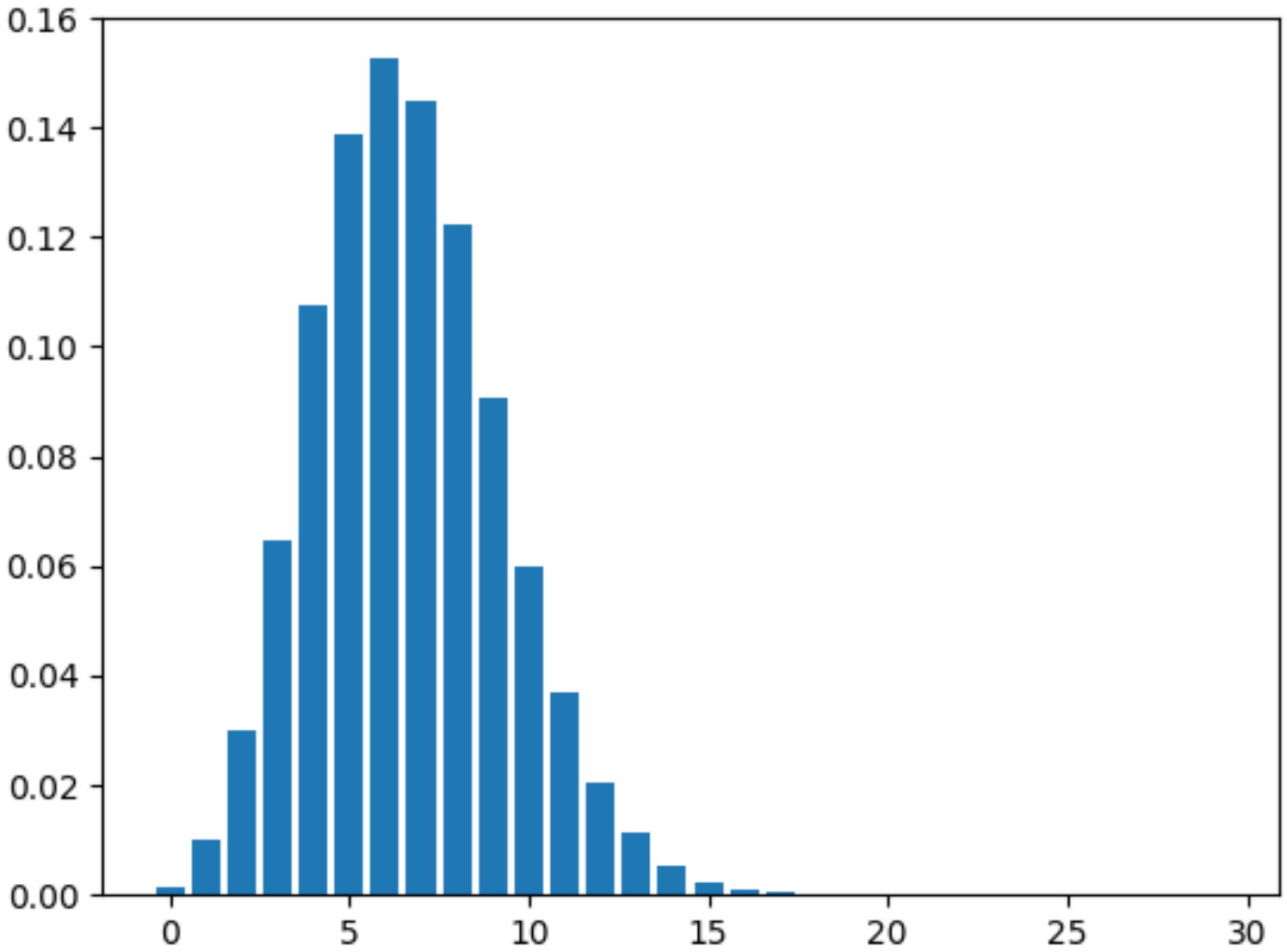
```
A=20
N=A
T=10000
b=100
a=100
k=1
s=1
```



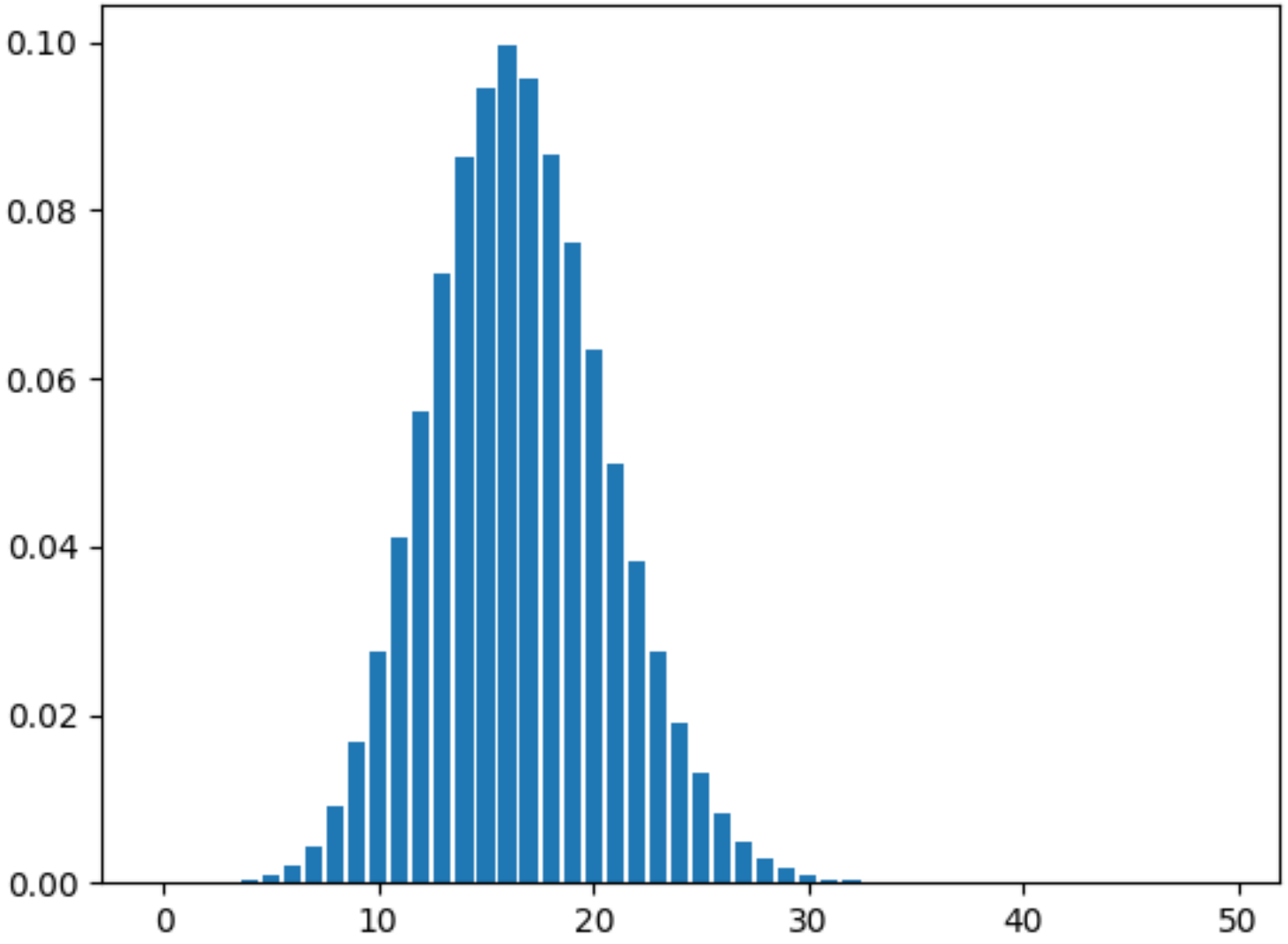
```
A=50
N=A
T=10000
b=100
a=100
k=1
s=1
```



```
A=20
N=A
T=10000
b=200
a=100
k=1
s=1
```



```
A=20
N=A
T=10000
b=100
a=500
k=1
s=1
```



Κώδικας 5
Το μοντέλο Schlogl

Μελετάει αντιδράσεις της μορφής

A+X<=>2X, X<=>B

Ζητούμενο είναι ο υπολογισμός της διασποράς της μέσης τιμής του πληθυσμού N του X, που προβλέπεται θεωρητικά στο α*V, όπου α είναι η παράμετρος ελέγχου και V ο όγκος.

```python
import random
import numpy as np
import matplotlib.pyplot as plt
a=10000
V=1
Ns=V*(a-1)
e=1e-8
imax=1000

s2=np.zeros(imax)
s0=np.zeros(imax)
P=np.zeros(imax+Ns)
s0[0]=1
P[Ns]=1
d=1
N=Ns
i=0

while d>e and i<imax-1:
    P[N+1]=(a*N)*P[N]/(N+1)/(1+N/V)
    N=N+1
    s2[i+1]=s2[i]+(N-Ns)**2*P[N]
    s0[i+1]=s0[i]+P[N]
    d=s2[i+1]-s2[i]
    i=i+1
print(i)
S2=s2[i]
S0=s0[i]

s2=np.zeros(imax)
s0=np.zeros(imax)
P=np.zeros(Ns+1)
s0[0]=1
P[Ns]=1
d=1
N=Ns
i=0


while d>e and i<imax-1 and N>=1:
    P[N-1]=P[N]*N*(1+(N-1)/V)/a/(N-1)
    N=N-1
    s2[i+1]=s2[i]+(N-Ns)**2*P[N]
    s0[i+1]=s0[i]+P[N]
    d=s2[i+1]-s2[i]
    i=i+1
print(i)
S2=s2[i]+S2
S0=s0[i]+S0
var=S2/S0
varth=a*V
div=(var-varth)/varth
print(var)
print(div*100)
```

a=2

```
16
0
0.9999999998703336
-50.000000006483326
```

a=50

```
60
42
47.35699810090164
-5.286003798196717
```

a=500

```
179
160
491.23920713379806
-1.7521585732403877
```

a=100000

```
2621
2599
99874.0024431764
-0.1259975568235968
```

a=5

```
23
3
3.8360404337913483
-23.279191324173034
```

a=100

```
82
65
96.17742002910562
-3.822579970894381
```

a=1000

```
252
233
987.5435415849297
-1.2456458415070302
```

a=1000000

```
8255
8233
999601.2168452921
-0.03987831547078677
```

a=10

```
30
8
9.10147670578659
-8.9852329421341
```

a=200

```
115
96
194.5202905770551
-2.7398547114724465
```

a=10000

```
808
787
9960.264724370702
-0.3973527562929849
```

a=10000000

```
25615
25594
9998738.592884341
-0.01261407115658745
```