Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Η/Υ

Συστήματα Αναμονής - Ροή Δ

1η Ομάδα Ασκήσεων - Αναφορά

<u>**Ονοματεπώνυμα**</u>**:**
Σταθά Ευσταθία

<u>**Αριθμοί Μητρώου:**</u>
el16190

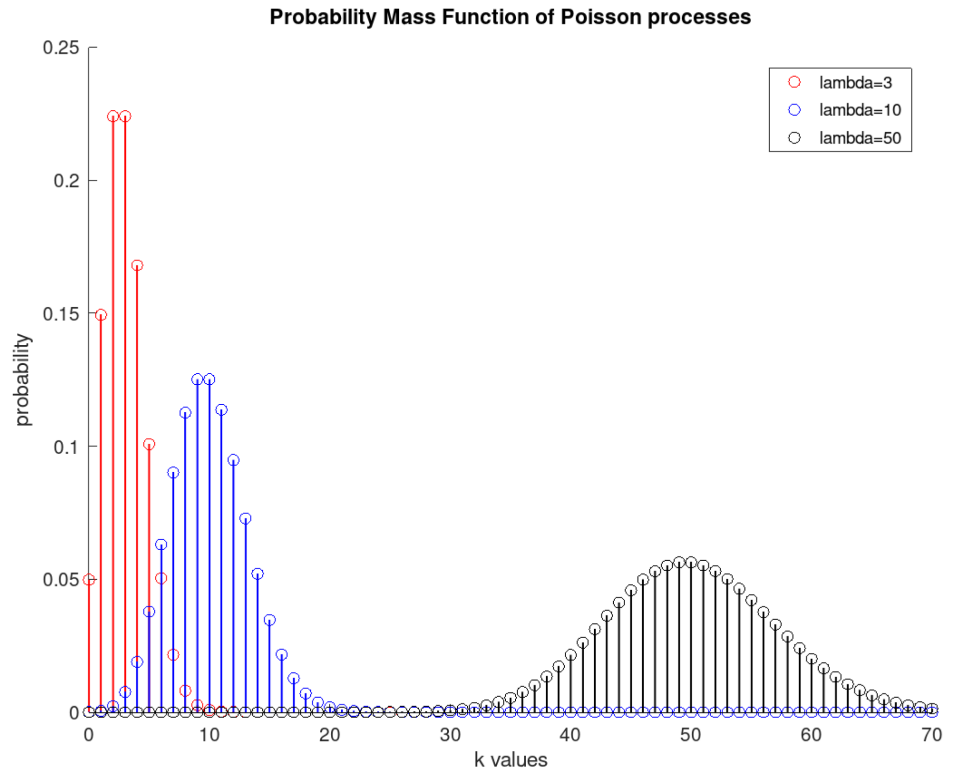
Πίνακας Περιεχομένων


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
    if
```

Άσκηση 1 - Κατανομή Poisson

Διαγράμματα - Σχόλια

Ερώτημα Α




```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

Όπως παρατηρούμε στο διάγραμμα, όσο αυξάνεται η τιμή της παραμέτρου λ της κατανομής Poisson, τόσο αυξάνεται και η διασπορά της. Το γεγονός αυτό επιβεβαιώνεται και από τη θεωρία, αφού γνωρίζουμε πως η διασπορά μιας τυχαίας μεταβλητής που ακολουθεί Κατανομή Poisson ισούται με λ. Ουσιαστικά, η ίδια παρουσιάζεται σαν να “απλώνει”, κατά μήκους του άξονα των x. Παράλληλα, μειώνεται και η μέγιστη τιμή της (το ύψος της), το οποίο παρατηρείται κοντά ή πάνω στην τιμή x = λ. Αυτό είναι λογικό, καθώς, όπως γνωρίζουμε, το άθροισμα της τιμής όλων των σημείων της συνάρτησης μάζας πιθανότητας της κατανομής Poisson (ως μιας διακριτής κατανομής) πρέπει να είναι 1. Ως εκ τούτου, όσο περισσότερα είναι τα σημεία της κατανομής, τόσο μικρότερες αναμένονται να είναι οι τιμές τους.


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

Ερώτημα Β

```
mean value of Poisson with lambda 30 is
mean_value =  30.000
Variance of Poisson with lambda 30 is
variance =  30.000
```

Όπως φαίνεται στο στιγμιότυπο οθόνης που τοποθετήθηκε παραπάνω, η μέση τιμής και η διασπορά της κατανομής Poisson με λ=30 είναι ίσες με 30, δηλαδή ίσες με λ. Αυτό είναι κάτι που αναμέναμε, καθώς το γνωρίζουμε από τη θεωρία, ενώ η απόδειξη γίνεται εύκολα, αλλά απαιτεί κάποιο μαθηματικό φορμαλισμό.

Αρχικά, για τη μέση τιμή έχουμε:

$$E[n] = \sum_{n=0}^{\infty} nP_n(t) = \sum_{n=0}^{\infty} n\frac{(\lambda t)^n}{n!}e^{-\lambda t} = e^{-\lambda t}\sum_{k=0}^{\infty}\frac{(\lambda t)^{k+1}}{k!} = \lambda t e^{-\lambda t}\sum_{k=0}^{\infty}\frac{(\lambda t)^k}{k!} = \lambda t e^{-\lambda t}e^{\lambda t} = \lambda t$$

Ενώ, για τη διασπορά ισχύει:

$$\sigma_n^2 = E[n^2] + (E[n])^2 = \sum_{n=0}^{\infty} n^2 P_n(t) - (\lambda t)^2$$

Όμως,

$$E[n(n-1)] = E[n^2] - E[n] => E[n^2] = E[n(n-1)] + E[n]$$

και, καθώς,


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

$$E[n(n-1)] = \sum_{n=0}^{\infty} n(n-1)P_n(t) = \sum_{n=0}^{\infty} n(n-1)\frac{(\lambda t)^n}{n!}e^{-\lambda t} = e^{-\lambda t}\sum_{k=0}^{\infty}\frac{(\lambda t)^{k+2}}{k!} = \lambda t e^{-\lambda t}\sum_{k=0}^{\infty}\frac{(\lambda t)^k}{k!} = (\lambda t)^2 e^{-\lambda t}e^{\lambda t} = (\lambda t)^2$$

προκύπτει εύκολα πως

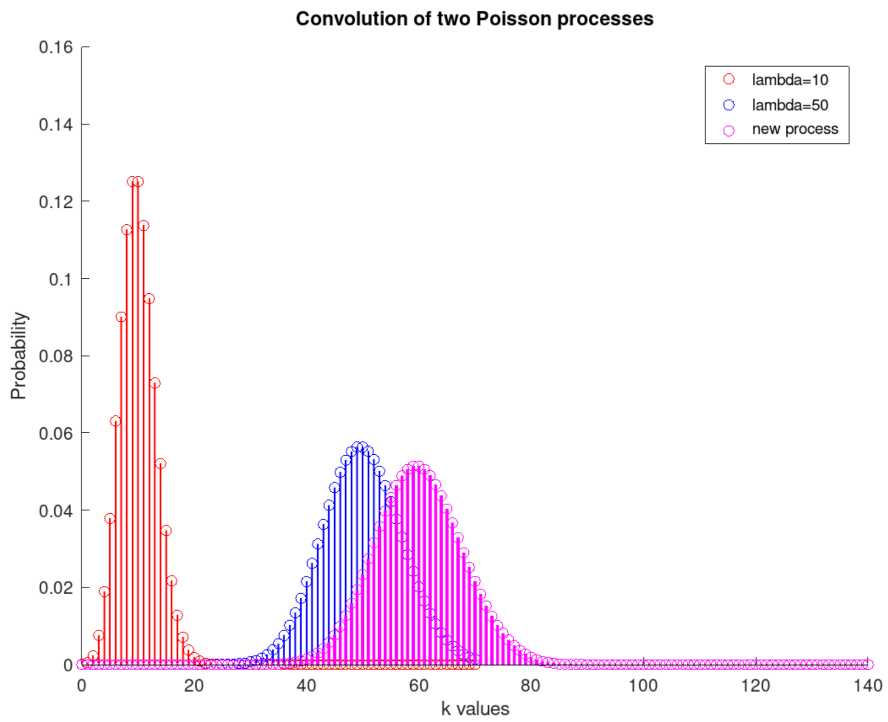
$$E[n^2] = (\lambda t)^2 + \lambda t \quad .$$

Επομένως,

$$\sigma_n^2 = (\lambda t)^2 + \lambda t - (\lambda t)^2 = \lambda t \quad .$$


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

Ερώτημα Γ

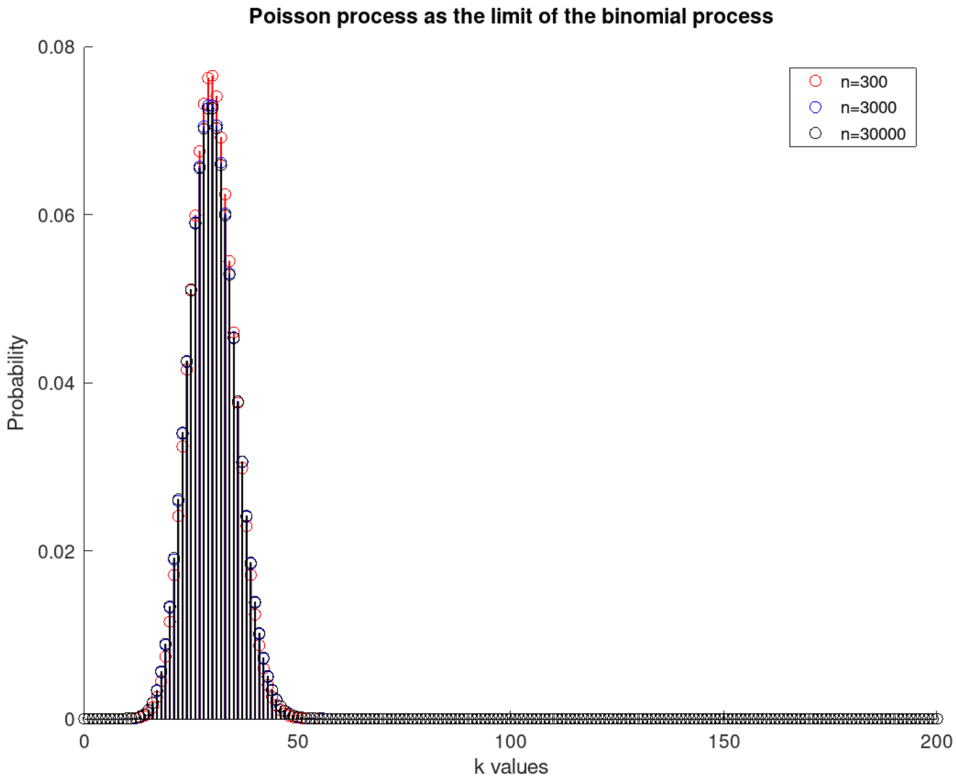


Όπως γνωρίζουμε από τη θεωρία, η κατανομή που προκύπτει από την υπέρθεση δύο ανεξάρτητων κατανομών Poisson με παραμέτρους λ_1 και λ_2 αντίστοιχα είναι, επίσης, μια κατανομή Poisson με παράμετρο $\lambda = \lambda_1 + \lambda_2$. Πράγματι, όπως βλέπουμε στο σχήμα η υπέρθεση των ανεξάρτητων κατανομών Poisson(10) και Poisson(50) μας έδωσε μια Poisson(60).


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

Ερώτημα Δ



Για μια τυχαία μεταβλητή Y ~ Διων(N, p), καθώς η παράμετρος N τείνει στο άπειρο, είναι γνωστό πως οι τιμές της συνάρτησης μάζας πιθανότητας της, συγκλίνουν στις τιμές της συνάρτησης μάζας πιθανότητας μιας τυχαίας μεταβλητής Z ~ Poisson(λ), όπου $\lambda = N * p$. Η προσέγγιση αυτή είναι καλή μόνο εφόσων $N \geq 100$. Επομένως, η ίδια η κατανομή Poisson είναι το όριο μιας διωνυμικής κατανομής παραμέτρων (n, p), όταν $n \to \infty$.


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

Κώδικας

```
pkg load statistics

clc;
clear all;
close all;

# TASK A: In a common diagram, design the Probability Mass Function of Poisson
Distribution
# with lambda parameters 3, 10, 30, 50. In the horizontal axes, choose k
parameters
# between 0 and 70.

k = 0:1:70;
lambda = [3, 10, 30, 50];
lambda_new = [3, 10, 50];

for i=1:columns(lambda_new)
  poisson(i,:) = poisspdf(k,lambda(i));
endfor

colors = "rbkm";
figure(1);
hold on;
for i=1:columns(lambda_new)
  stem(k,poisson(i,:),colors(i),"linewidth",1.2);
endfor
hold off;
```



```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

```
title("Probability Mass Function of Poisson processes");
xlabel("k values");
ylabel("probability");
legend("lambda=3","lambda=10","lambda=50");

# TASK B: regarding the poisson process with parameter lambda 30, compute its
mean
# value and variance

index = find(lambda == 30);
chosen = poisson(index,:);
mean_value = 0;

for i=0:(columns(poisson(index,:))-1)
  p_new = poisson(index, i+1);
  mean_value = mean_value + i.*poisson(index,i+1);
endfor

display("mean value of Poisson with lambda 30 is");
display(mean_value);

second_moment = 0;
for i=0:(columns(poisson(index,:))-1)
  second_moment = second_moment + i.*i.*poisson(index,i+1);
endfor

variance = second_moment - mean_value.^2;
display("Variance of Poisson with lambda 30 is");
display(variance);

# TASK C: consider the convolution of the Poisson distribution with lambda 20
```



```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

```
with
# the Poisson distribution with lambda 30.

first = find(lambda_new==10);
second = find(lambda_new==50);
poisson_first = poisson(first,:);
poisson_second = poisson(second,:);

composed = conv(poisson_first,poisson_second);
new_k = 0:1:(2*70);

figure(2);
hold on;
stem(k,poisson_first(:),colors(1),"linewidth",1.2);
stem(k,poisson_second(:),colors(2),"linewidth",1.2);
stem(new_k,composed,"mo","linewidth",2);
hold off;
title("Convolution of two Poisson processes");
xlabel("k values");
ylabel("Probability");
legend("lambda=10","lambda=50","new process");

# TASK D: show that Poisson process is the limit of the binomial distribution.
k = 0:1:200;
# Define the desired Poisson Process
lambda = 30;

n = [lambda.*10, lambda.*100, lambda.*1000];
p = lambda./n;

figure(3);
```



```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
    if
```

```
hold on;
for i=1:3
  binomial = binopdf(k,n(i),p(i));
  stem(k,binomial,colors(i),'linewidth',1.2);
endfor
hold off;
title("Poisson process as the limit of the binomial process");
xlabel("k values");
ylabel("Probability");
legend("n=300", "n=3000", "n=30000");
```

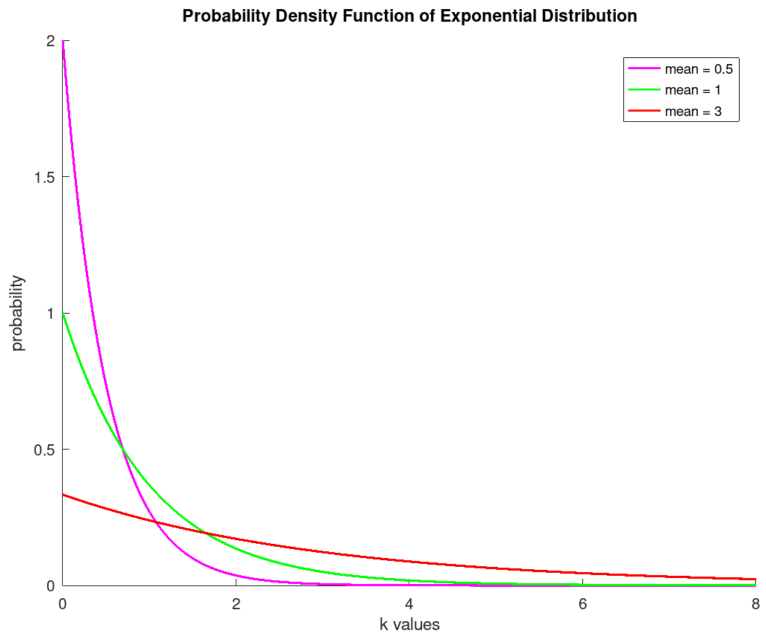


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

Άσκηση 2 - Εκθετική Κατανομή

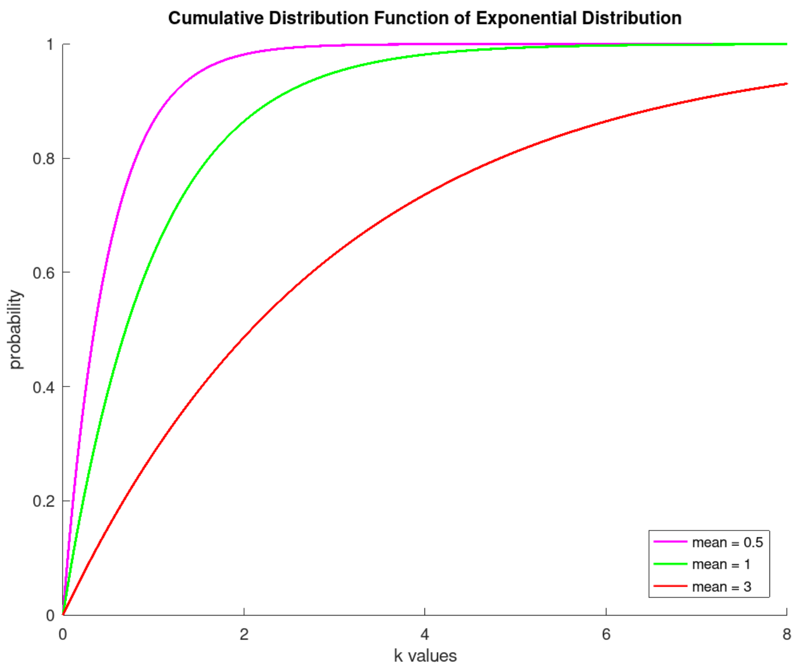
Διαγράμματα - Σχόλια

Ερώτημα Α



Ερώτημα Β


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```



Ερώτημα Γ

```
P(X>30000) when X~Exp(1/2.5) is
probab_1 =  0.88692
P(X>50000|X>20000) when X~Exp(1/2.5) is
probab_2 =  0.88692
```

Όπως παρατηρούμε οι δύο πιθανότητες είναι ίδιες, ακριβώς όπως περιμέναμε με βάση τη θεωρία. Συγκεκριμένα, γνωρίζουμε πως η Εκθετική Κατανομή χαρακτηρίζεται από την ιδιότητα έλλειψης


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

μνήμης, μάλιστα είναι η μόνη κατανομή συνεχούς μεταβλητής με αυτή την ιδιότητα. Μπορούμε να εξηγήσουμε καλύτερα το αποτέλεσμα αυτό, χρησιμοποιώντας λίγο μαθηματικό φορμαλισμό:

Έστω X ~ Εκθ(λ) και s,t >0 τότε:

$$P(X > t + s | X > s) = \frac{P(X > t + s \bigcap X > s)}{P(X > s)} = \frac{P(X > t + s)}{P(X > s)} = e^{-\lambda(t+s)}e^{\lambda s} = e^{-\lambda t}$$

Κώδικας

```
pkg load statistics

clc;
clear all;
close all;
nfig = 1;
colors = "mgrbk";

# TASK A: In a common diagram, design the Probability Density Function of
Exponential Distribution
# with mean 0.5, 1, 3. In the horizontal axes, choose k parameters between 0 and
8.

k = 0:0.00001:8;
mean = [0.5, 1, 3];

for i = 1:3
  exp(i,:) = exppdf(k, mean(i));
endfor
```



```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

```
figure(nfig);
hold on;
for i = 1:3
  plot(k, exp(i,:), colors(i), 'LineWidth', 1.5);
endfor
hold off;

title("Probability Density Function of Exponential Distribution");
xlabel("k values");
ylabel("probability");
legend("mean = 0.5", "mean = 1", "mean = 3");

# TASK B: In a common diagram, design the Cumulative Distribution Function of
Exponential Distribution
# with mean 0.5, 1, 3. In the horizontal axes, choose k parameters between 0 and
8.

for i = 1:3
  exp(i,:) = expcdf(k, mean(i));
endfor

nfig = nfig + 1;
figure(nfig);
hold on;
for i = 1:3
  plot(k, exp(i,:), colors(i), 'LineWidth', 1.5);
endfor
hold off;
```



```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

```
title("Cumulative Distribution Function of Exponential Distribution");
xlabel("k values");
ylabel("probability");
legend("mean = 0.5", "mean = 1", "mean = 3", 'location', 'southeast');

# TASK C: calculate tha probabilities P(X>30000) and P(X>50000|X>20000) with
1/lamda = 2.5 sec
# to show memory loss property of exponential distribution

exp_c = expcdf(k, 2.5);

probab_1 = 1 - exp_c(30000);
display("P(X>30000) when X~Exp(1/2.5) is")
display(probab_1)

probab_2 = (1 - exp_c(50000))/(1 - exp_c(20000));
display("P(X>50000|X>20000) when X~Exp(1/2.5) is")
display(probab_2)
```



```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

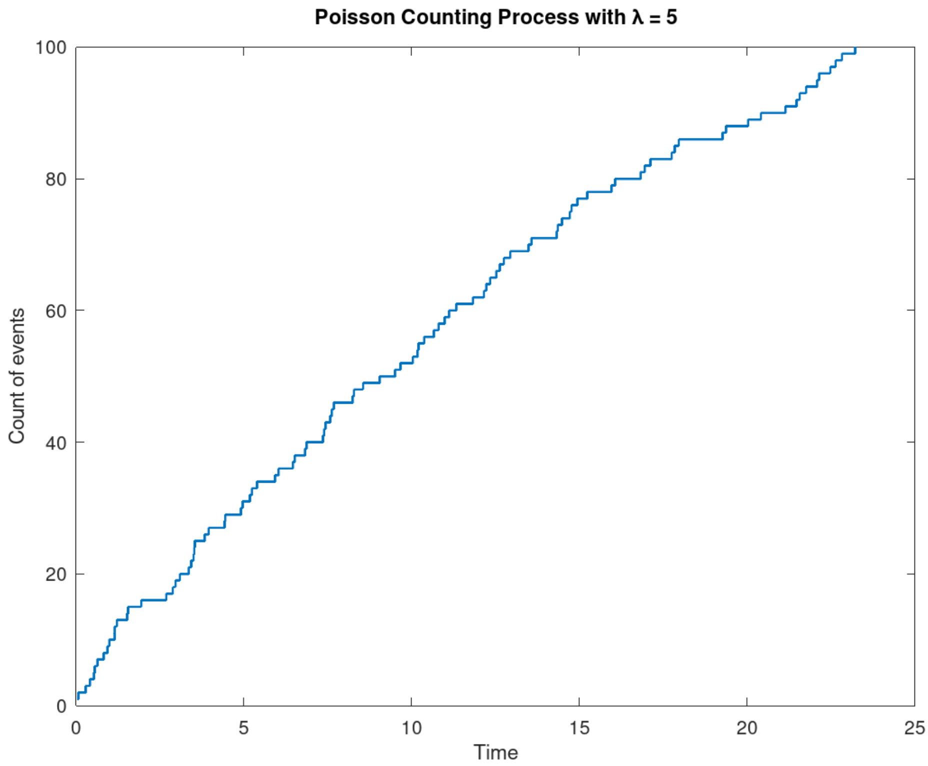
Άσκηση 3 - Διαδικασία Καταμέτρησης Poisson

Διαγράμματα - Σχόλια

Ερώτημα Α

Όπως γνωρίζουμε, οι χρόνοι που μεσολαβούν ανάμεσα στην εμφάνιση δύο διαδοχικών γεγονότων Poisson ακολουθούν την εκθετική κατανομή. Δημιουργήσαμε 100 τέτοια διαδοχικά τυχαία γεγονότα και παρακάτω φαίνεται το διάγραμμα που αντιστοιχεί στη διαδικασία καταμέτρησης Poisson.


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```



Ερώτημα Β

Ο αριθμός γεγονότων Poisson(λ) σε ένα χρονικό παράθυρο $\Delta T = T_1 - T_2$ συνιστά μια τυχαία μεταβλητή ν, η οποία ακολουθεί κατανομή Poisson με μέση τιμή $E_T[\nu] = \lambda * \Delta T$. Όπως αναφέραμε νωρίτερα, η μέση τιμή μιας κατανομής Poisson με παράμετρο λ είναι και η ίδια λ, επομένως περιμένουμε ανεξαρτήτως του πλήθους των τυχαίων γεγονότων, ο μέσος αριθμός γεγονότων που


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```

εμφανίζονται στη μονάδα του χρόνου να συγκλίνει στη μέση τιμή της κατανομής που ο ίδιος ακολουθεί, δηλαδή στην τιμή $E_T[v] = \lambda * \Delta T = 5 * 1 = 5$. Πράγματι, κάνοντας όλα τα απαιτούμενα πειράματα παρατηρήσαμε πως αυτό ισχύει, όπως φαίνεται και στο στιγμιότυπο οθόνης που λήφθηκε από την κονσόλα του Octave.

```
Mean value of poisson events in unit of time for 100 events, is 4.098.

Mean value of poisson events in unit of time for 200 events, is 4.6718.

Mean value of poisson events in unit of time for 300 events, is 5.0328.

Mean value of poisson events in unit of time for 500 events, is 4.867.

Mean value of poisson events in unit of time for 1000 events, is 5.0361.

Mean value of poisson events in unit of time for 100000 events, is 4.9704.
```

Κώδικας


```
pkg load statistics

clc;
clear all;
close all;

nfig = 1;

number_of_events = [100, 200, 300, 500, 1000, 100000];

for j=1:6
  random_intervals = exprnd(1/5, 1, number_of_events(j));

  event_time = zeros(1, number_of_events(j));
  counts = ones(1, number_of_events(j));
  event_time(1) = random_intervals(1);

  for i=2:number_of_events(j)
    event_time(i) = event_time(i-1) + random_intervals(i);
    counts(i) = counts(i-1) + 1;
```