ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

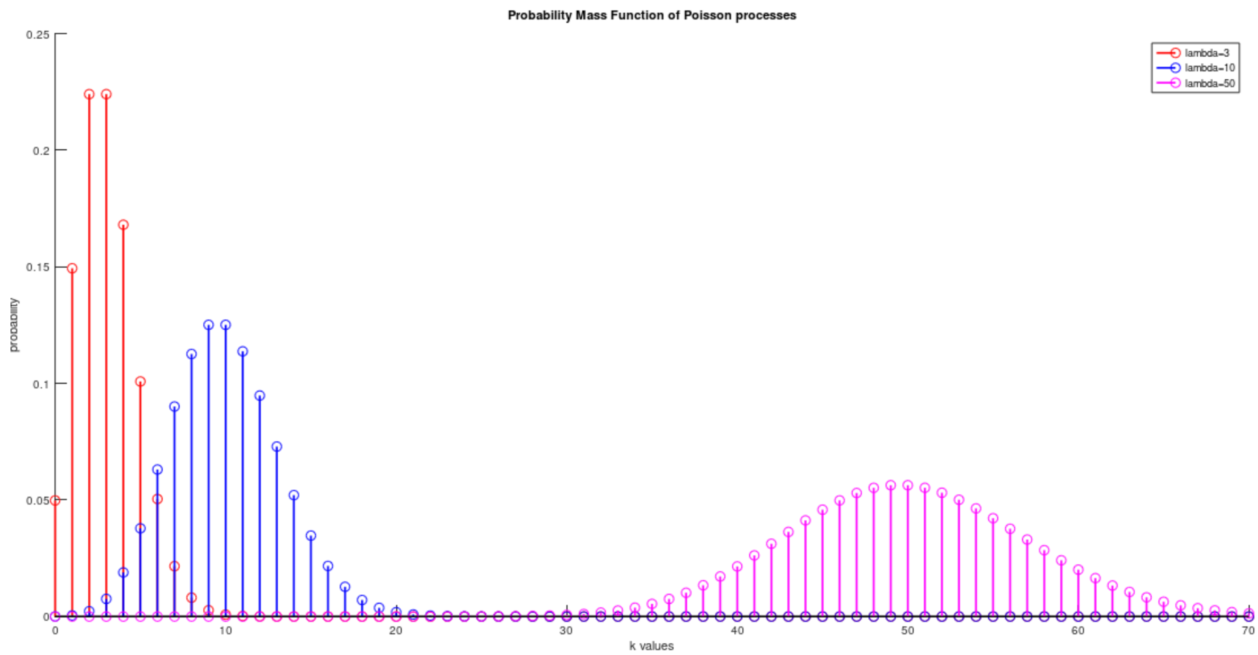
Ονοματεπώνυμο: Γκούμε Λαουρεντιάν
Α.Μ.: 031 18 014
Έτος/Εξάμηνο: 3ο/6ο

ΣΥΣΤΗΜΑΤΑ ΑΝΑΜΟΝΗΣ (Queuing Systems)

1η Σειρά Ασκήσεων

Κατανομή Poisson

Α) Παρουσιάζεται η Συνάρτηση Μάζας Πιθανότητας (PDF) της κατανομής Poisson για λ = {3, 10, 50}, όπου λ η μέση τιμή της κατανομής, καθώς γνωρίζουμε ότι για μία κατανομή X~Poisson(λ) => **E(X) = λ**, καθώς και **Var(X) = λ**.



Αποτέλεσμα 1.Α

Ο κώδικας που αναπαράγει την ανωτέρω γραφική παράσταση:

```
pkg load statistics

clc;
clear all;
close all;
#Task1_1_A
#
k = 0:1:70;
lambda = [3, 10, 30, 50];

for i=1:columns(lambda)
  poisson(i,:) = poisspdf(k, lambda(i));          #for each lambda, an array with the pdf for each k is created
endfor

colors = "rbkm";
figure(1);
hold on;
for i=1:columns(lambda)                            #lambda columns traversal
  if(i != find(lambda == 30))                      #we do not print for lambda = 30
    stem(k,poisson(i,:),colors(i),"linewidth",1.2);  #stem is used for discrete functions
  endif
endfor
hold off;

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

Κώδικας 1.Α

Παρατηρούμε ότι όσο το λ μεγαλώνει, τόσο οι γραφικές μετατοπίζονται δεξιά και παρουσιάζονται πιο “απλωμένες”. Αυτό είναι αναμενόμενο, αφού όπως είπαμε η μέση τιμή (προκαλεί την μετατόπιση), καθώς και η διασπορά (καθορίζει το πόσο “απλωμένη” θα είναι η γραφική) είναι ίσες με το λ για διαδικασίες που ακολουθούν κατανομή Poisson. Διαισθητικά, επειδή το άθροισμα των πιθανοτήτων P[X = k] για κάθε k που ορίσαμε (από το 0 έως το 70 εδώ) θα πρέπει να αθροίζει στο 1, όσο “πέφτει” το μέγιστο της γραφικής, τόσο θα πρέπει να “απλωθεί”, ώστε να αντισταθμίσει τις απώλειες αυτές.

Β) Υπολογίζουμε τη μέση τιμή και τη διακύμανση μέσω των σχέσεων **E[X] = ∑(k * P[X = k])** και **V[X] = E[X^2] – (E[X])^2,** αντίστοιχα, για τα k που μας ενδιαφέρουν. Τα αποτελέσματα, με βάση τα όσα αναφέραμε στο προηγούμενο ερώτημα, αναμένεται να είναι ίσα μεταξύ τους και ίσα με λ = 30. Τα αποτελέσματα της προσομοίωσης, επιβεβαιώνουν το παραπάνω:

```
Mean value of Poisson with lambda 30 is: 30.000000
Variance of Poisson with lambda 30 is: 30.000000
```

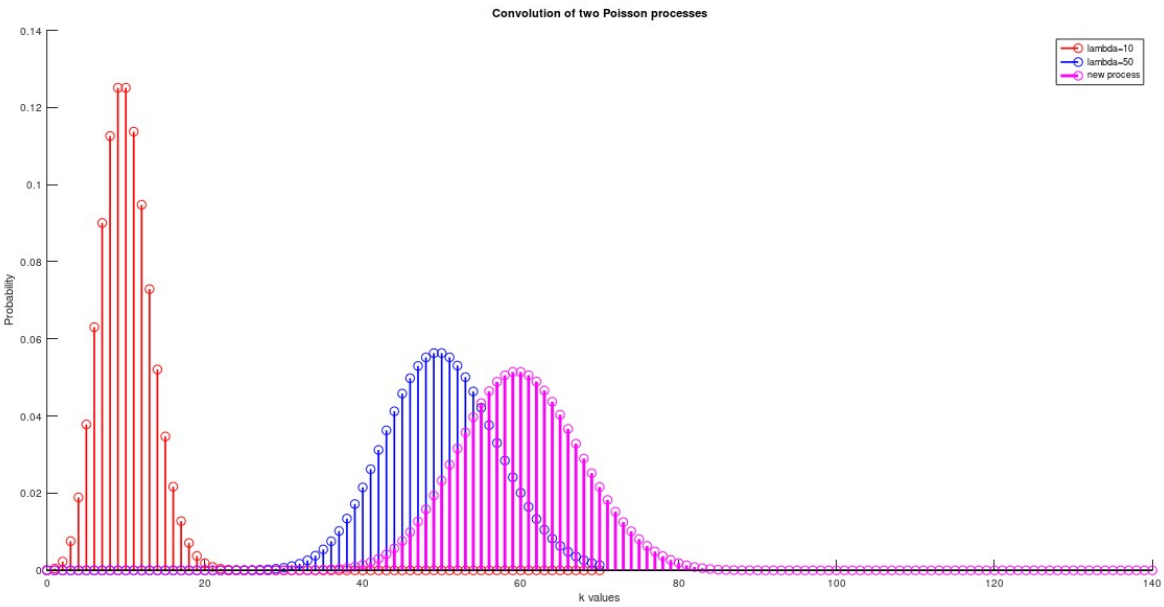
Αποτέλεσμα 1.Β

Ο κώδικας που διεκπεραιώνει τις παραπάνω διαδικασίες:

```
#Task1_1_B
#

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

#Columns(...) = 71, so we subtract 1, so i runs from 0 to 70 => 71 times
for i=0:(columns(poisson(index,:))-1)
  mean_value = mean_value + i.*poisson(index,i+1);
endfor

printf("Mean value of Poisson with lambda 30 is: %f\n", 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;
printf("Variance of Poisson with lambda 30 is: %f\n", variance);
```

<u>Κώδικας 1.Β</u>

Γ) Παρατηρούμε ότι από την υπέρθεση (μέσω της συνέλιξής τους) δύο κατανομών Poisson, προκύπτει μία **νέα κατανομή Poisson** με μεγαλύτερη μέση τιμή και διασπορά από αυτές που τη δημιούργησαν, καθώς ισχύει ότι $\boldsymbol{\lambda_{new} = \lambda_1 + \lambda_2}$, όπου λ_1, λ_2 οι παράμετροι των κατανομών από τις οποίες η νέα προήλθε. Απαραίτητη προϋπόθεση για να συμβαίνει αυτό, ωστόσο, είναι οι **υπερτιθέμενες κατανομές να είναι μεταξύ τους ανεξάρτητες**. Τα αποτελέσματα παρουσιάζονται παρακάτω:



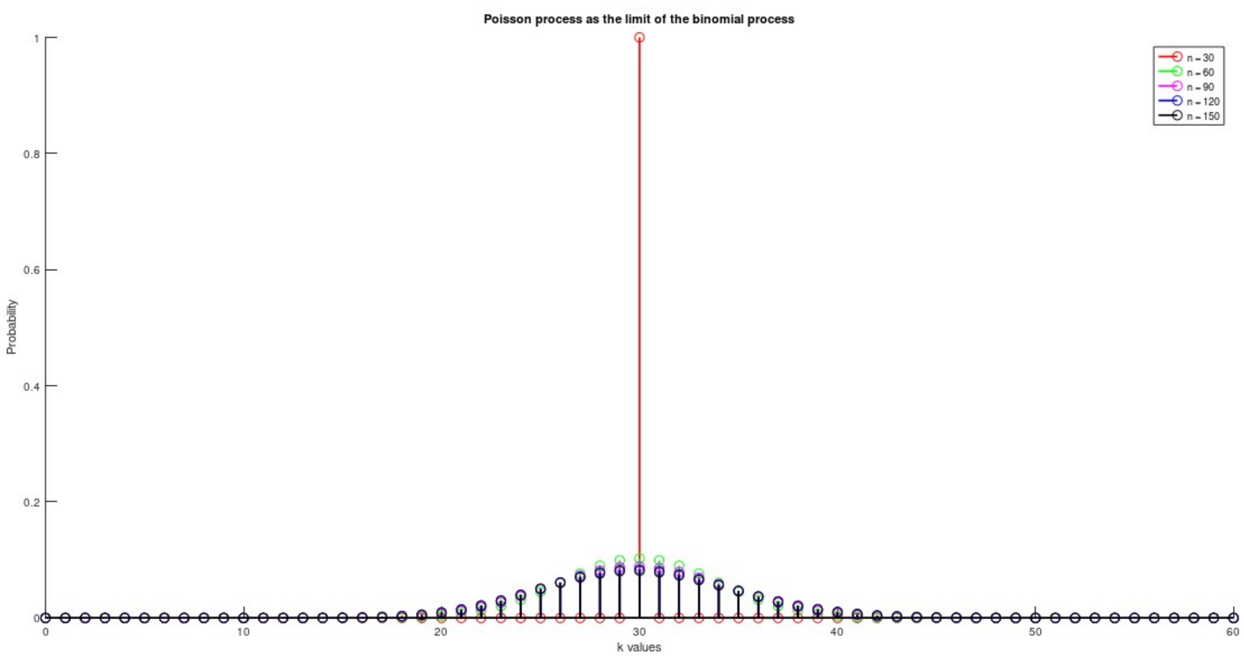
<u>Αποτέλεσμα 1.Γ</u>

Ο κώδικας που χρησιμοποιήθηκε για το συγκεκριμένο τμήμα:

```
# #Task1_1_C
#
#

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

composed = conv(poisson_first,poisson_second);        #convolution of two processes
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");
```

Κώδικας 1.Γ

Δ) Έστω λ μια δεδομένη θετική σταθερά και για κάθε n φυσικό μία Τυχαία Μεταβλητή Y_n με κατανομή Διωνυμική(n, p) και πυκνότητα $P_n(k) = P[Y_n = k]$, για k = 0, 1, 2…, n. Καθώς το n τείνει στο άπειρο, **οι τιμές της πυκνότητας $P_n(k)$ συγκλίνουν στις αντίστοιχες τιμές της πυκνότητας P(k) μίας Τ.Μ. Ζ με κατανομή Poisson με παράμετρο λ**. Συνεπώς όταν έχουμε πολλά γεγονότα ($n \geq 100$) και η πιθανότητα $p = (\lambda/n)$ είναι μικρή ($p < 0.04$) έτσι ώστε το γινόμενο np να είναι της τάξης του 1 μπορούμε με καλή ακρίβεια να προσεγγίσουμε μία διωνυμική κατανομή ως μία κατανομή Poisson. Μέσω της προσομοίωσης, μπορούμε να οπτικοποιήσουμε το αποτέλεσμα αυτό:



Αποτέλεσμα 1.Δ

Το τμήμα κώδικα που απαιτήθηκε για την αναπαράσταση:

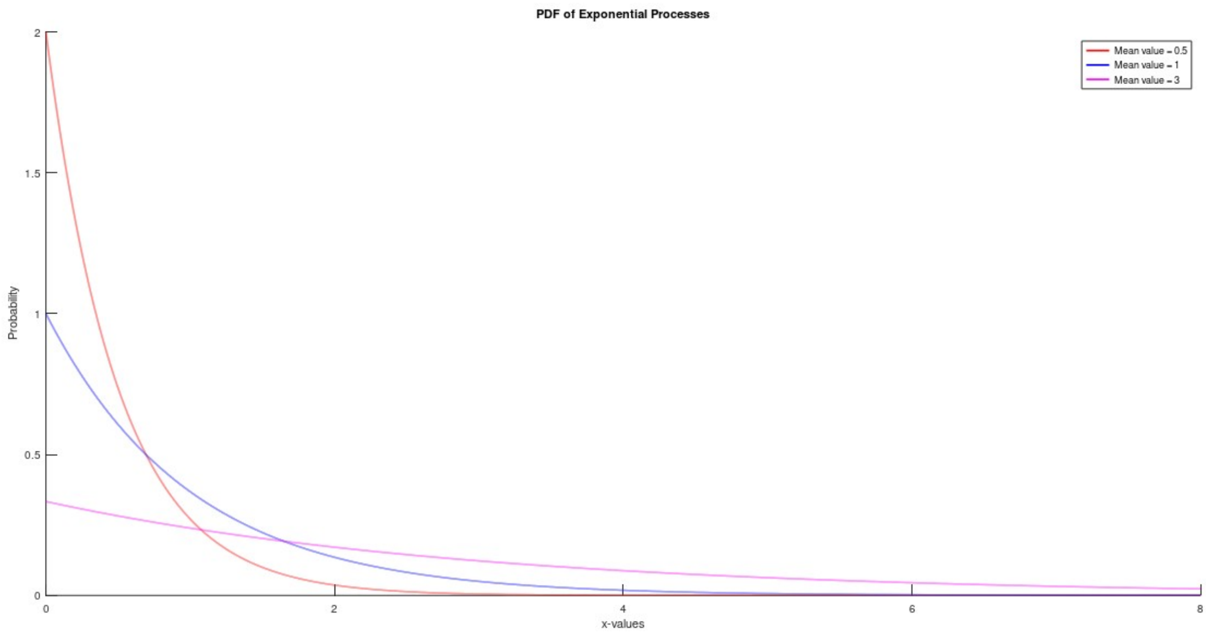
```
# Task1_1_D
#

k = 0:1:60;
# Define the desired Poisson Process
lambda = 30;
i = [1, 2, 3, 4, 5];
#Bin(n,p = lambda/n)
n = lambda.*i;
p = lambda./n;
colors = "rgmbk"

figure(3);
title("Poisson process as the limit of the binomial process");
xlabel("k values");
ylabel("Probability");
hold on;
for i = 1 : columns(i)
  binomial = binopdf(k,n(i),p(i));
  stem(k, binomial, colors(i), 'linewidth', 1.2);
endfor
legend("n = 30", "n = 60", "n = 90", "n = 120", "n = 150")
hold off;
```

Κώδικας 1.Δ

Σημείωση: Οι ανωτέρω κώδικες έχουν γραφτεί στο ίδιο script με τη σειρά κατά την οποία παρουσιάστηκαν

Εκθετική Κατανομή

Α) Προσομοιώνουμε τη **Συνάρτηση Πυκνότητας Πιθανότητας για εκθετική κατανομή** με μέσο όρο (1/λ) = {0.5, 1, 3}:



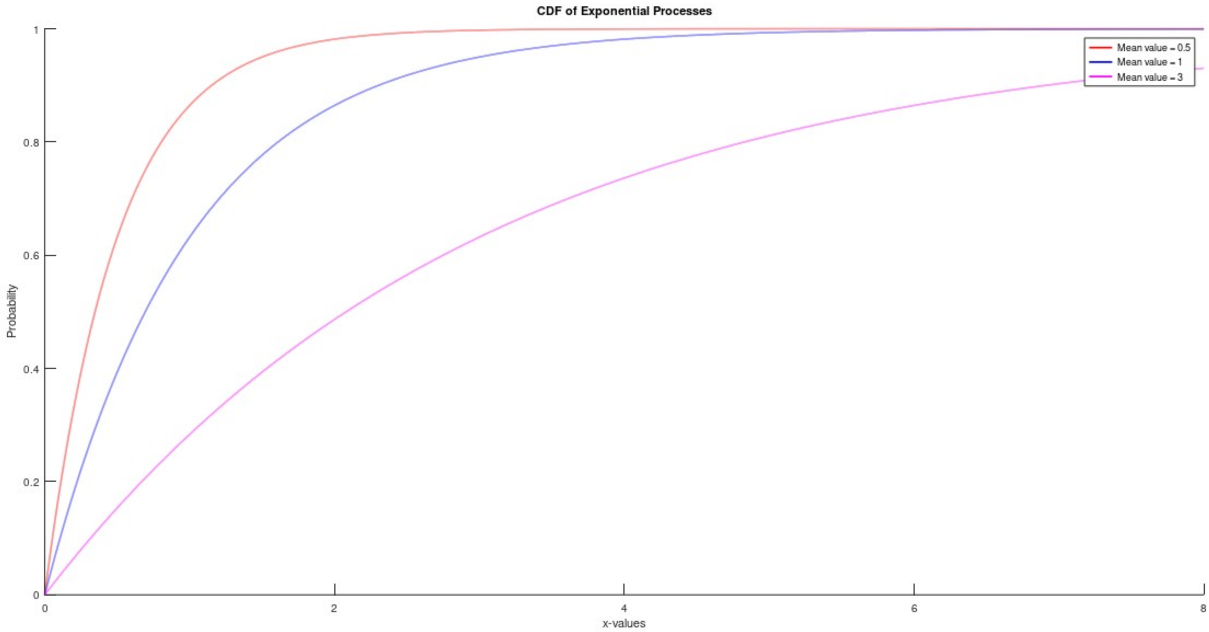
<u>Αποτέλεσμα 2.Α</u>

Ο κώδικας που υλοποιεί το παραπάνω αποτέλεσμα:

```
pkg load statistics

clc;
clear all;
close all;

#Task1_2_A
k = 0 : 0.0001 : 8;
mean_value = [0.5, 1, 3];     #(1/lambda)

for i = 1 : columns(mean_value)
  exponential_pdf(i, :) = exppdf(k, mean_value(i));
endfor

colors = "rbm";
figure(1);
hold on;
for i = 1 : columns(mean_value)
  plot(k, exponential_pdf(i, :), colors(i), "linewidth", 1.2);
endfor

hold off
title("PDF of Exponential Processes");
xlabel("x-values");
ylabel("Probability");
legend("Mean value = 0.5", "Mean value = 1", "Mean value = 3");

```

<u>Κώδικας 2.Α</u>

B) Αντίστοιχα με τη Σ.Π.Π. (PDF), προσομοιώνουμε την **Συνάρτηση Κατανομής Πιθανότητας (CDF)** για ίδιες τιμές του (1/λ):



Αποτέλεσμα 2.Β

Ο κώδικας που απαιτήθηκε:

```
#Task1_2_B

for i = 1 : columns(mean_value)
  exponential_cdf(i, :) = expcdf(k, mean_value(i));
endfor

figure(2);
hold on
for i = 1 : columns(mean_value)
  plot(k, exponential_cdf(i, :), colors(i), "linewidth", 1.2);
endfor
hold off

title("CDF of Exponential Processes");
xlabel("x-values");
ylabel("Probability");
legend("Mean value = 0.5", "Mean value = 1", "Mean value = 3");
```

Κώδικας 2.Β

Σημείωση: Οι ανωτέρω κώδικες έχουν γραφτεί στο ίδιο script με τη σειρά κατά την οποία παρουσιάστηκαν

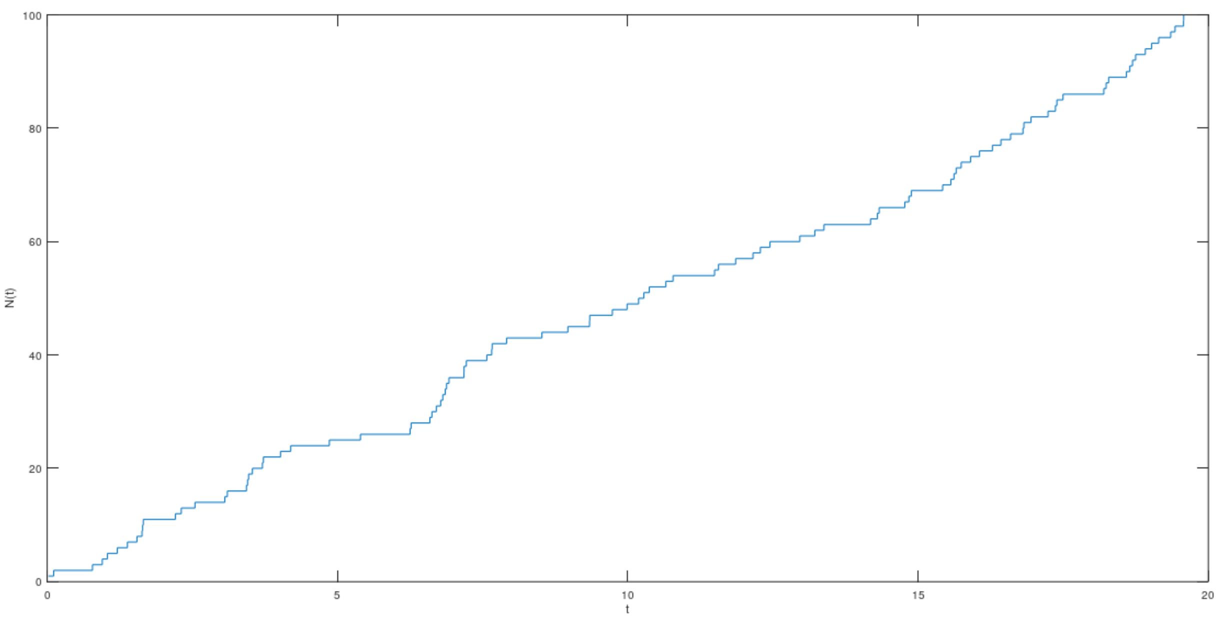
Γ) Για τις ζητούμενες πιθανότητες έχουμε (όπου $F[x] = 1 - e^{-\lambda x}$ για $x \geq 0$):

- $P[X > 30.000] = 1 - P[X \leq 30.000] = 1 - F[30.000] = 1 - (1 - e^{-(2.5)*30.000}) = e^{-75.000}$
- $P[X > 50.000 \mid X > 20.000) = \dfrac{P[X>50.000 \wedge X>20.000]}{P[X>20.000]} = \dfrac{P[X>50.000]}{P[X>20.000]} = \dfrac{1-P[X\leq 50.000]}{1-P[X\leq 20.000]} = \dfrac{1-F[50.000]}{1-F[20.000]} = \dfrac{\exp(-(2.5)*50.000)}{\exp(-(2.5)*20.000)} = e^{-75.000}$

Παρατηρούμε ότι οι 2 πιθανότητες προέκυψαν ίσες μεταξύ τους. Αυτό ήταν αναμενόμενο, λόγω της ιδιότητας έλλειψης μνήμης της εκθετικής κατανομής, η οποία μας λέει το εξής:
Αν έχω a, b >0 (στην περίπτωσή μας 20.000 και 30.000 αντίστοιχα), τότε **P[X ≥ a + b | X ≥ a] = P[X ≥ b]**. Επομένως, η δεσμευμένη αυτή πιθανότητα είναι ανεξάρτητη του a και ίση με εκείνη που αντιστοιχεί στο a = 0. Συγκεκριμένα, για τα δικά μας δεδομένα: P[X ≥ 50.000 | X ≥ 20.000] = P[X ≥ 20.000 + 30.000 | X ≥ 20.000] = P[X ≥ 30.000].

<u>Διαδικασία Καταμέτρησης Poisson</u>

Α) Γνωρίζουμε ότι οι **χρόνοι που μεσολαβούν μεταξύ δύο διαδοχικών γεγονότων Poisson ακολουθούν εκθετική κατανομή**, είναι επομένως ανεξάρτητοι από το παρελθόν. Παρακάτω προσομοιώνουμε μια διαδικασία καταμέτρησης Poisson 100 τυχαίων γεγονότων, όπου θεωρούμε πως σε κάθε γεγονός έχουμε μοναδιαία αύξηση (μία προσθήκη στην ουρά):



<u>Αποτέλεσμα 3.Α</u>

Β) Γνωρίζουμε ότι το πλήθος των εμφανίσεων σε ένα διάστημα (t, t+T) είναι μια διακριτή Τ.Μ. ίση με ν = N(t+T) – N(t) και για περιπτώσεις όπου τα γεγονότα εμφανίζονται ανεξάρτητα από το παρελθόν και δεν επηρεάζουν το μέλλον (όπως και συμβαίνει εδώ με την εκθετική κατανομή που ακολουθούν οι ενδιάμεσοι χρόνοι) η **ν ακολουθεί κατανομή Poisson**. Ο μέσος αριθμός εμφανίσεων σε αυτή την περίπτωση είναι ίσος με λΤ, επομένως, αναμένουμε ο μέσος αριθμός εμφανίσεων στη μονάδα του χρόνου να είναι ίσος με (λΤ)/Τ = λ = 5 γεγονότα/sec. Η προσομοίωση επιβεβαιώνει την πρόβλεψη αυτή, **με αποτέλεσμα πλησιέστερο στο λ, όσο μεγαλώνει το πλήθος των γεγονότων**:

```
Mean number of arrivals for N = 100 in a time period T is: 5.066759
Mean number of arrivals for N = 200 in a time period T is: 5.611935
Mean number of arrivals for N = 300 in a time period T is: 5.563723
Mean number of arrivals for N = 500 in a time period T is: 5.055238
Mean number of arrivals for N = 1000 in a time period T is: 4.854101
Mean number of arrivals for N = 10000 in a time period T is: 5.003253
```

Αποτέλεσμα 3.Β

Ο κώδικας που χρησιμοποιήθηκε για τα παραπάνω 2 ερωτήματα:

```
pkg load statistics

clc;
clear all;
close all;

#Task1_3_A and Task1_3_B
lambda = 5;
mean = 1/lambda;
N = [100,  200, 300, 500, 1000, 10000];              #Number of arrivals

for i = 1 : columns(N)
  arrivals = exprnd(mean, 1, N(i));                  #random samples
  increased_arrivals = arrivals;
  for j = 1 : (N(i)-1)                               #increasing random samples
    increased_arrivals(1, j+1) = increased_arrivals(1, j) + arrivals(1, j+1);
  endfor
  if(i == 1)                                         #plot for Task1_3_A
    y = 1 : N(i);
    stairs(increased_arrivals(1,:), y);
    xlabel("t");
    ylabel("N(t)");
   endif
   mean_arrivals = N(i)/increased_arrivals(1, N(i));    #means display
   printf("Mean number of arrivals for N = %d in a time period T is: %f \n", N(i), mean_arrivals)
endfor
```

Κώδικας 3.Α και 3.Β