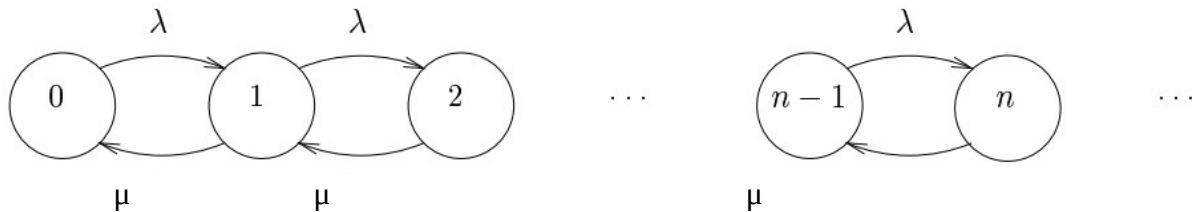


Συστήματα Αναμονής (Queuing Systems)

2ο Εργαστήριο
Θοδωρής Παπαρηγόπουλος (el18040)

Θεωρητική μελέτη της ουράς M/M/1

α) Προκειμένου να είναι η ουρά M/M/1 εργοδική πρέπει $\rho = \frac{\lambda}{\mu} (\text{erlangs}) < 1$



Έτσι, έχουμε Κατανομή Poisson με παράμετρο λ πελάτες ανά δευτερόλεπτο και οι εξυπηρετήσεις εκθετική κατανομή με παράμετρο μ πελάτες το δευτερόλεπτο.

$$\lambda_0 = \lambda_1 = \dots = \lambda_i = \dots = \lambda, i = 1, 2, 3, \dots$$

$$\text{και } \mu_0 = \mu_1 = \dots = \mu_i = \dots = \mu, i = 1, 2, 3, \dots$$

Χρησιμοποιώντας τις εξισώσεις ισορροπίας (τοπικές και γενικές) είναι $\lambda P_{i-1} = \mu P_i, i = 1, 2, 3, \dots$

$$\text{και } (\lambda_k + \mu_k) P_k = \lambda_{k-1} P_{k-1} + \mu_{k+1} P_{k+1}, k = 1, 2, 3, \dots$$

Επιπλέον έχουμε την κανονικοποίηση της εργοδοτικής πιθανότητας $P_0 + P_1 + \dots + P_N = 1$ έχουμε

$$\lambda P_0 = \mu P_1 \Rightarrow P_1 = \frac{\lambda}{\mu} P_0$$

$$(\lambda + \mu) P_1 = \lambda P_0 + \mu P_2 \Rightarrow P_2 = \left(\frac{\lambda}{\mu}\right)^2 P_0 = \rho^2 P_0 \Rightarrow \text{επαγωγικά προκύπτει } P_k = \rho^k P_0$$

$$\frac{1}{1-\rho} P_0 = 1 \Rightarrow P_0 = 1 - \rho \Rightarrow P_k = (1 - \rho) \rho^k, k > 0 \text{ and } P(n(t) > 0) = 1 - P_0 = \rho$$

β) Από τον Little's Law ξέρουμε πως ο μέσος χρόνος αναμονής ενός πελάτη στο σύστημα σε ισορροπία

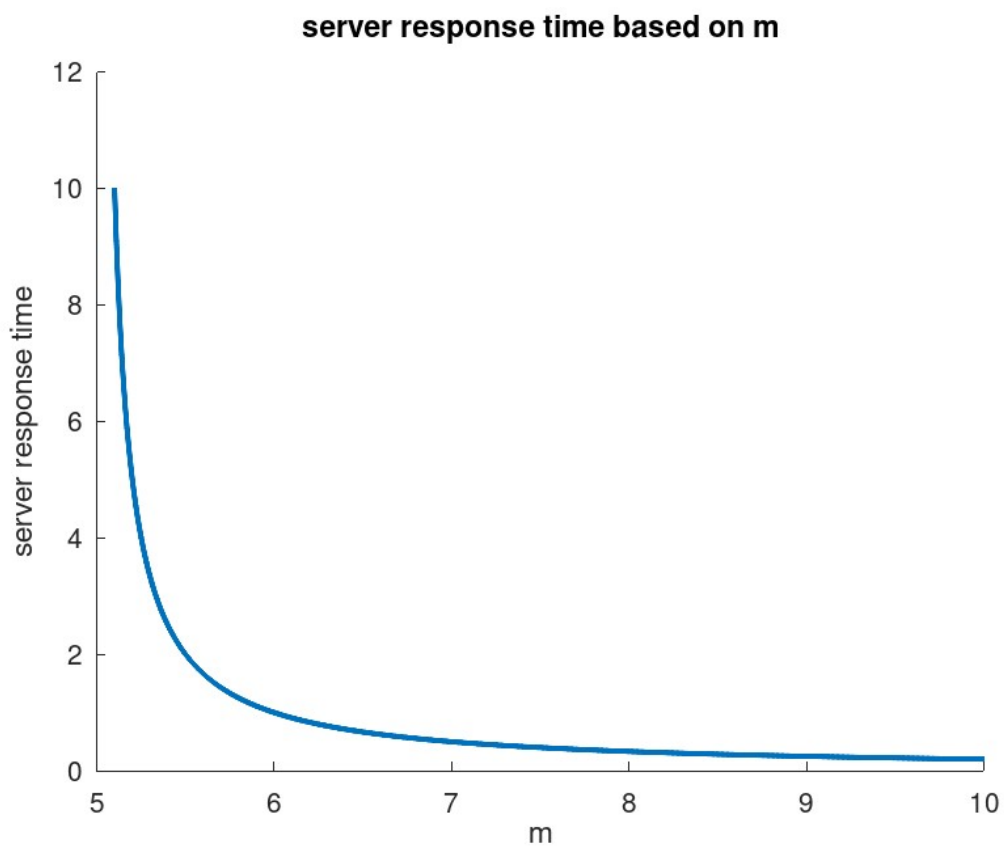
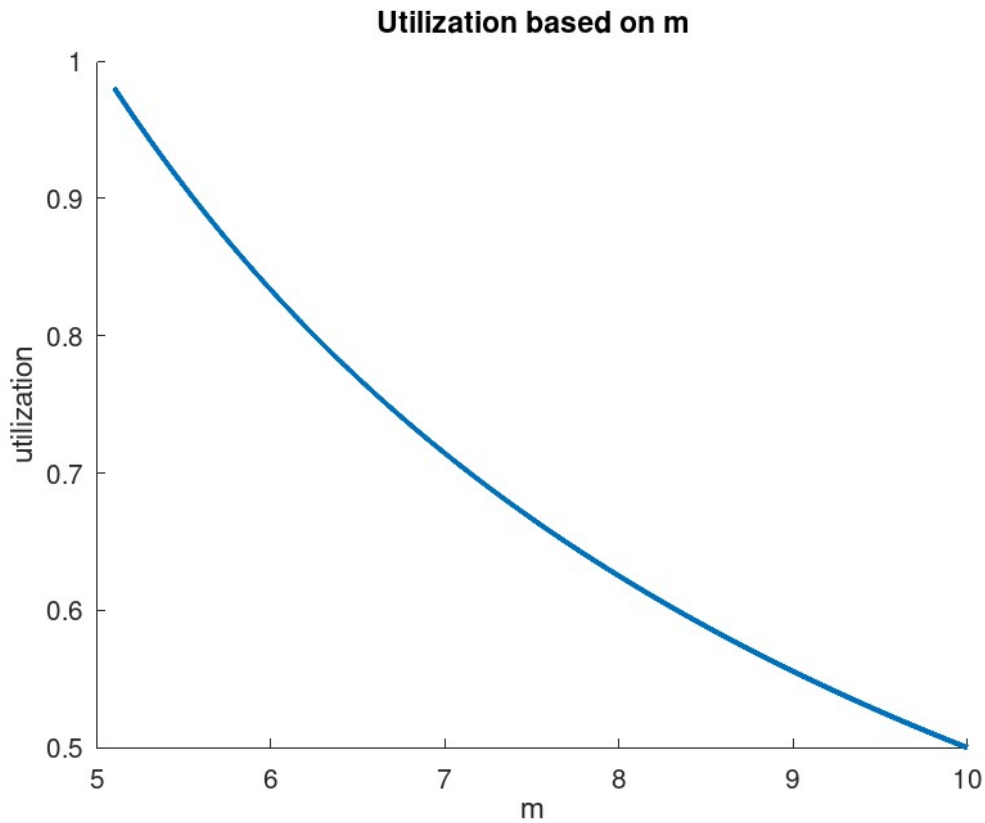
$$\text{είναι } E(T) = E\left[\frac{n(t)}{\gamma}\right] = E\left[\frac{n(t)}{\lambda}\right] = \frac{1}{\mu(1-\rho)}$$

γ) Για $k = 57$ είναι $P_{57} = (1 - \rho) \rho^{57} > 0$ που μας οδηγεί σε μια πιθανότητα θετική που είναι να έχουμε 57 πελάτες.

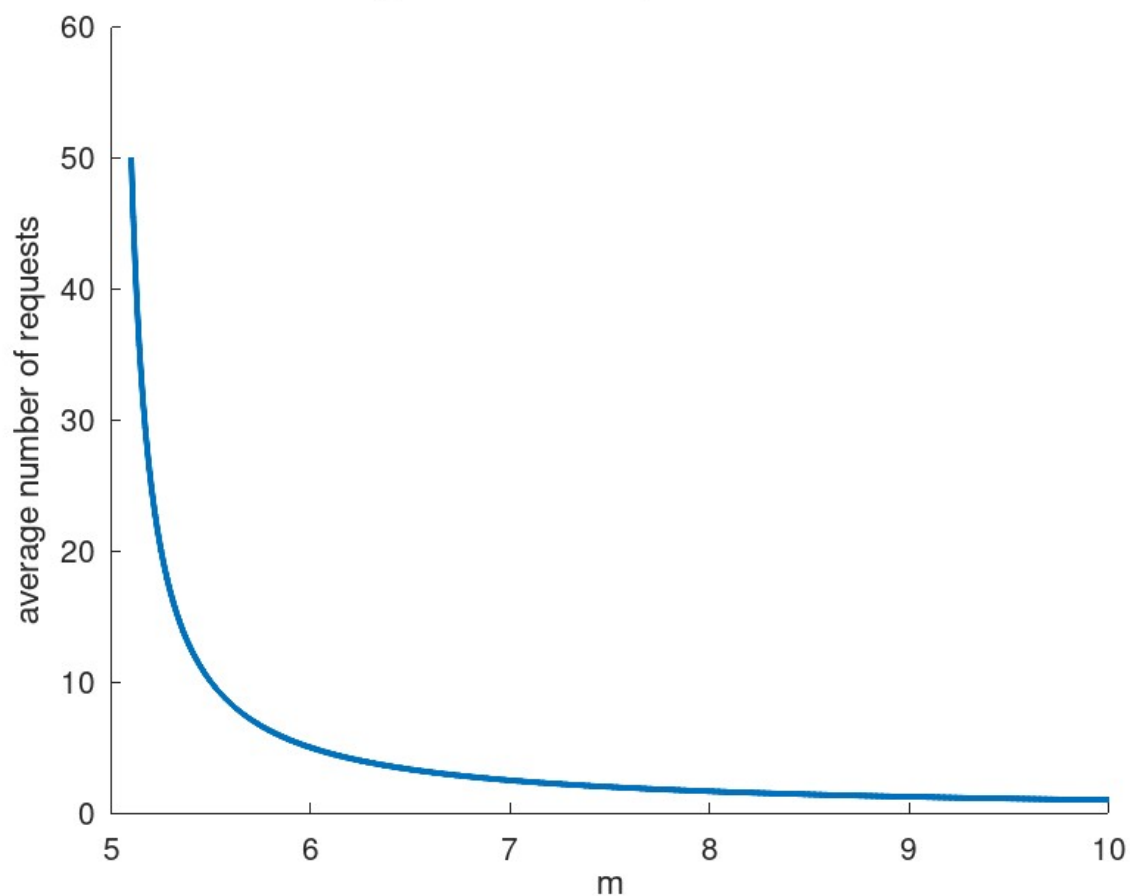
Ανάλυση ουράς M/M/1 με Octave

α) Από πριν έχουμε πως πρέπει $\lambda < \mu$ για να είναι εργοδοτικό. Επιλέγουμε 50 δείγματα που οδηγούν σε 10 πελάτες το λεπτό. Οι ρυθμοί εξυπηρέτησης θα είναι: $\mu = [6, 9.99](\rho < 1)$.

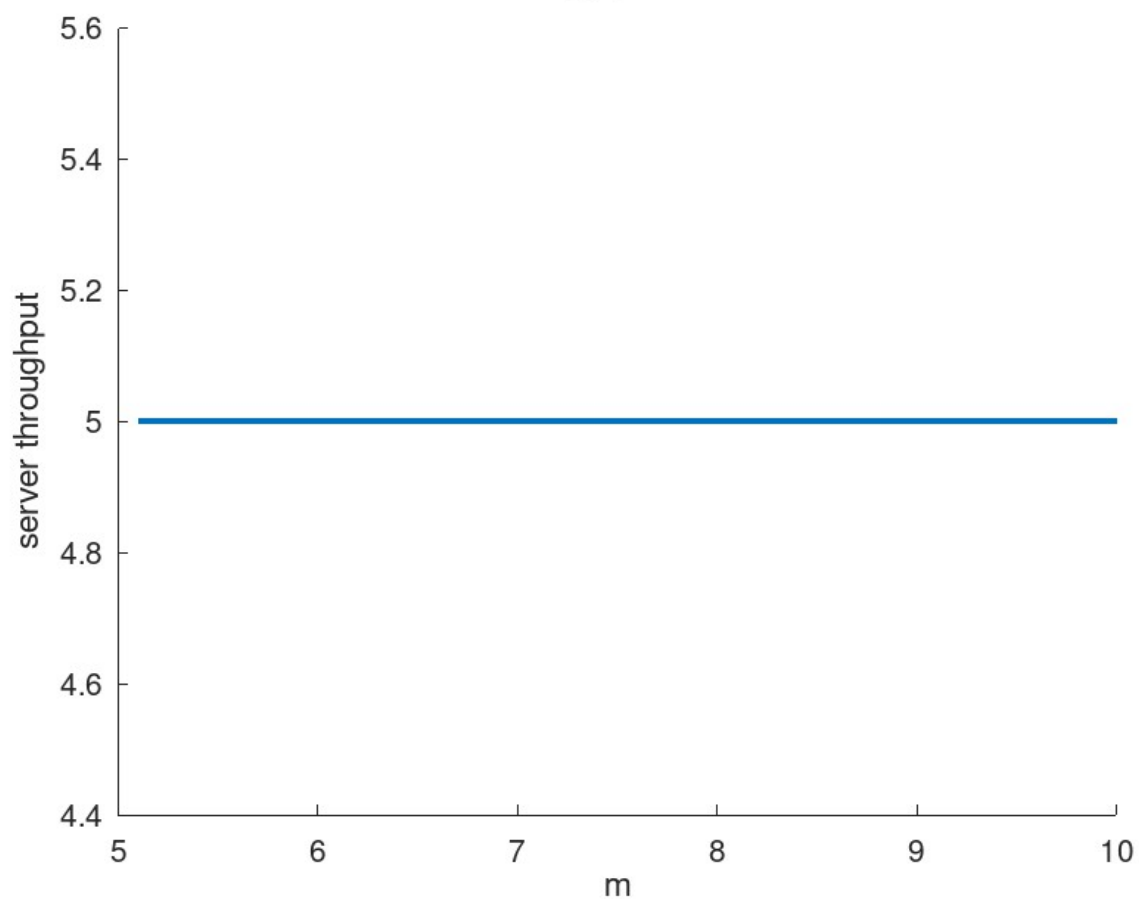
β) (Το $m = \mu$)



average number of requests based on m



server throughput based on m



γ) Παρατηρούμε πως ο μέσος χρόνος εξισορροπείται μεταξύ 8 και 10 πελάτες το λεπτό. Για να έχουμε ένα καλύτερο σύστημα και ταυτόχρονα ελάχιστο δυνατό κόστος, χρειάζεται να διαλέξουμε το ελάχιστο όριο που εξισορροπείται η κατάσταση. Έτσι κοντά στους 8 πελάτες το λεπτό θα ήταν το πιο κατάλληλο.

δ) Γνωρίζουμε πως $\gamma = \lambda(1 - P(\text{χασουμε πελάτη}))$. Σε ένα M/M/1 σύστημα, η ουρά έχει άπειρη χωρητικότητα οπότε $P(\text{χασουμε πελάτη}) = 0$ και έτσι το throughput είναι σταθερό και ίσο με λ

Κώδικας

```
clc;
clear all;
close all;
pkg load statistics
pkg load queueing

lamda = 5;
utilization = [0,500];
server_response_time = [0,500];
average_number_of_requests = [0,500];
server_throughput = [0,500];

m = [5.1:0.01:10];

for i=1:columns(m)
    [utilization(i),server_response_time(i),average_number_of_requests(i),server_throughput(i)] =
    qmmm1(lamda, m(i));
endfor
figure(1);
hold on;
title("Utilization based on m");
plot(m,utilization,"linewidth", 2.2);
xlabel("m");
ylabel("utilization");
hold off;

figure(2) ;
hold on;
title("server response time based on m");
plot(m,server_response_time,"linewidth", 2.2);
xlabel("m");
ylabel("server response time");
hold off;

figure(3) ;
hold on;
title("average number of requests based on m");
plot(m,average_number_of_requests,"linewidth", 2.2);
xlabel("m");
ylabel("average number of requests");
hold off;
figure(4);
hold on;
title("server throughput based on m");
plot(m,server_throughput,"linewidth", 2.2);
xlabel("m");
ylabel("server throughput");
hold off;
```

Διαδικασία γεννήσεων θανάτων (birth-death process) L εφαρμογή σε σύστημα M/M/1/K

α)

$$\text{Έχουμε πως } \lambda_i = \frac{\lambda}{(i+1)} \text{ και } \mu_i = \mu, i=0,1,2,3$$

$$\lambda_0 = P_0 = \mu_1 P_1 \Rightarrow \mu_{k-1} P_{k-1} = \mu_k P_k, k=1,2,3,4$$

$$P_1 + P_2 + P_3 + P_4 = 1$$

$$P_k = \frac{\lambda^k}{k! \mu^k} P_0, k=0,1,2,3,4 \Rightarrow P_k = \frac{\rho^k}{k!} P_0, k=0,1,2,3,4$$

$$\sum P_k = 1 \Rightarrow P_0 \left(1 + \rho + \frac{\rho^2}{2} + \frac{\rho^3}{6} + \frac{\rho^4}{24} \right) = 1 \Rightarrow$$

$$P_0 = 0.606635071,$$

$$P_1 = 0.303317536,$$

$$P_2 = 0.0758293839,$$

$$P_3 = 0.0126382306,$$

$$P_4 = 0.00157977883$$

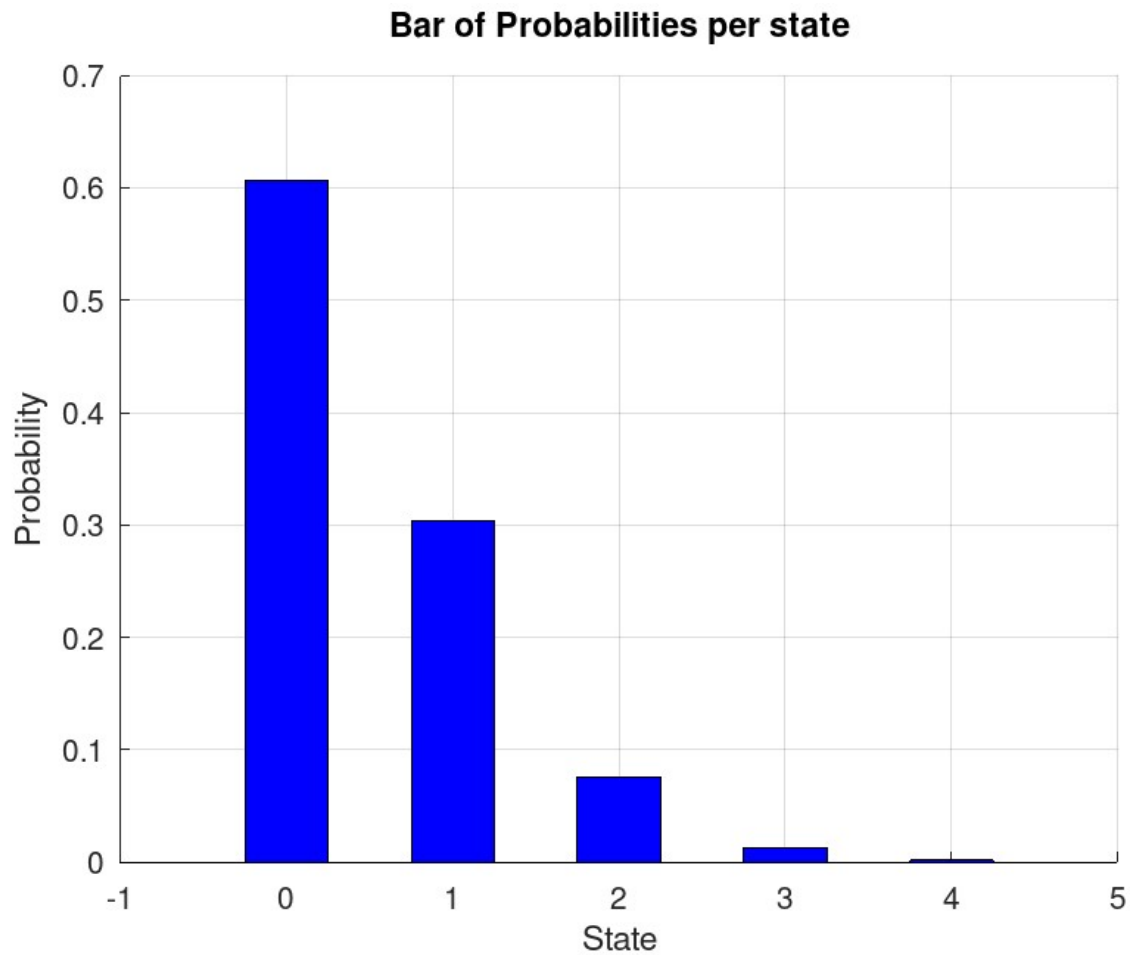
$$P(\text{Blocking}) = P_4 = 0.00157977883$$

β)

i) Η μήτρα μεταβάσεων είναι:

-5.0000	5.0000	0	0	0
10.0000	-12.5000	2.5000	0	0
0	10.0000	-11.6667	1.6667	0
0	0	10.0000	-11.2500	1.2500
0	0	0	10.0000	-10.0000

ii)



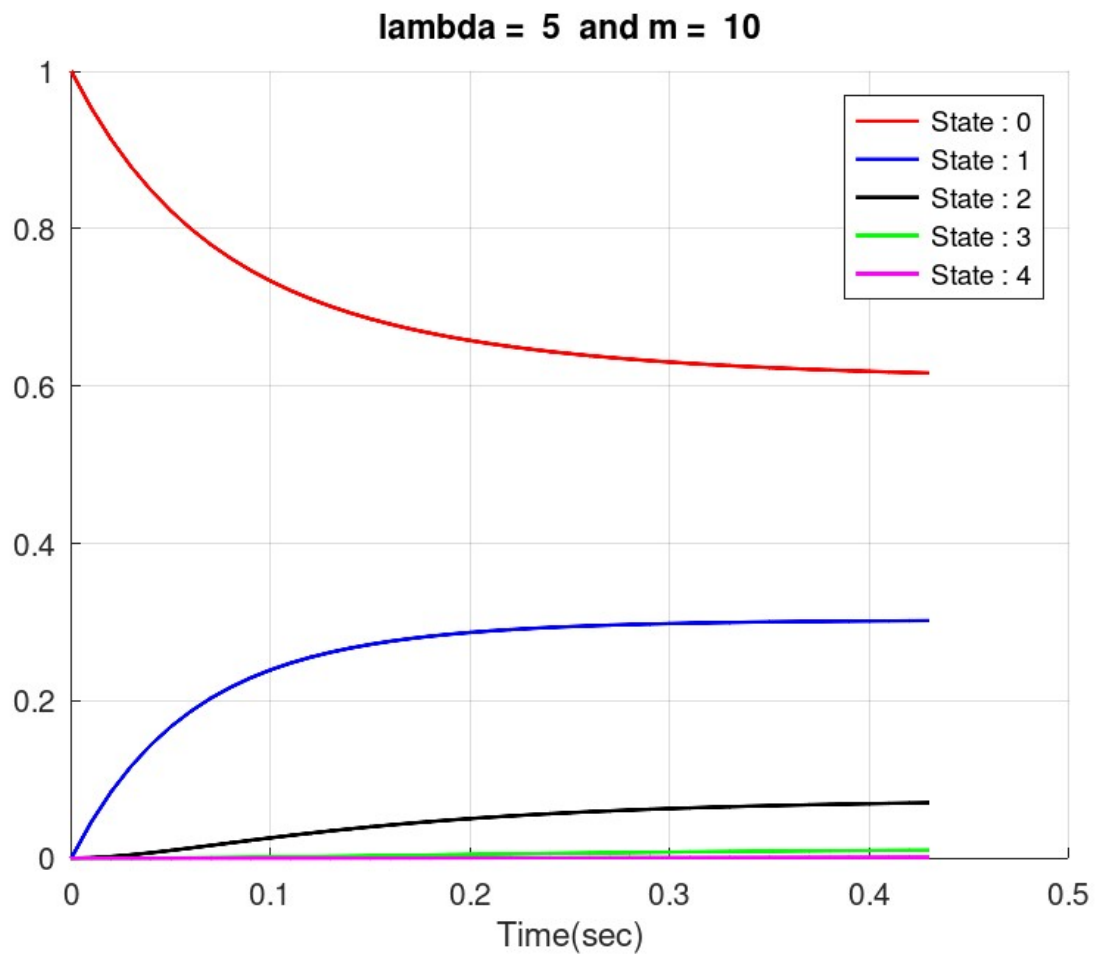
iii)

$P = 6.0664e-01 \quad 3.0332e-01 \quad 7.5829e-02 \quad 1.2638e-02 \quad 1.5798e-03$

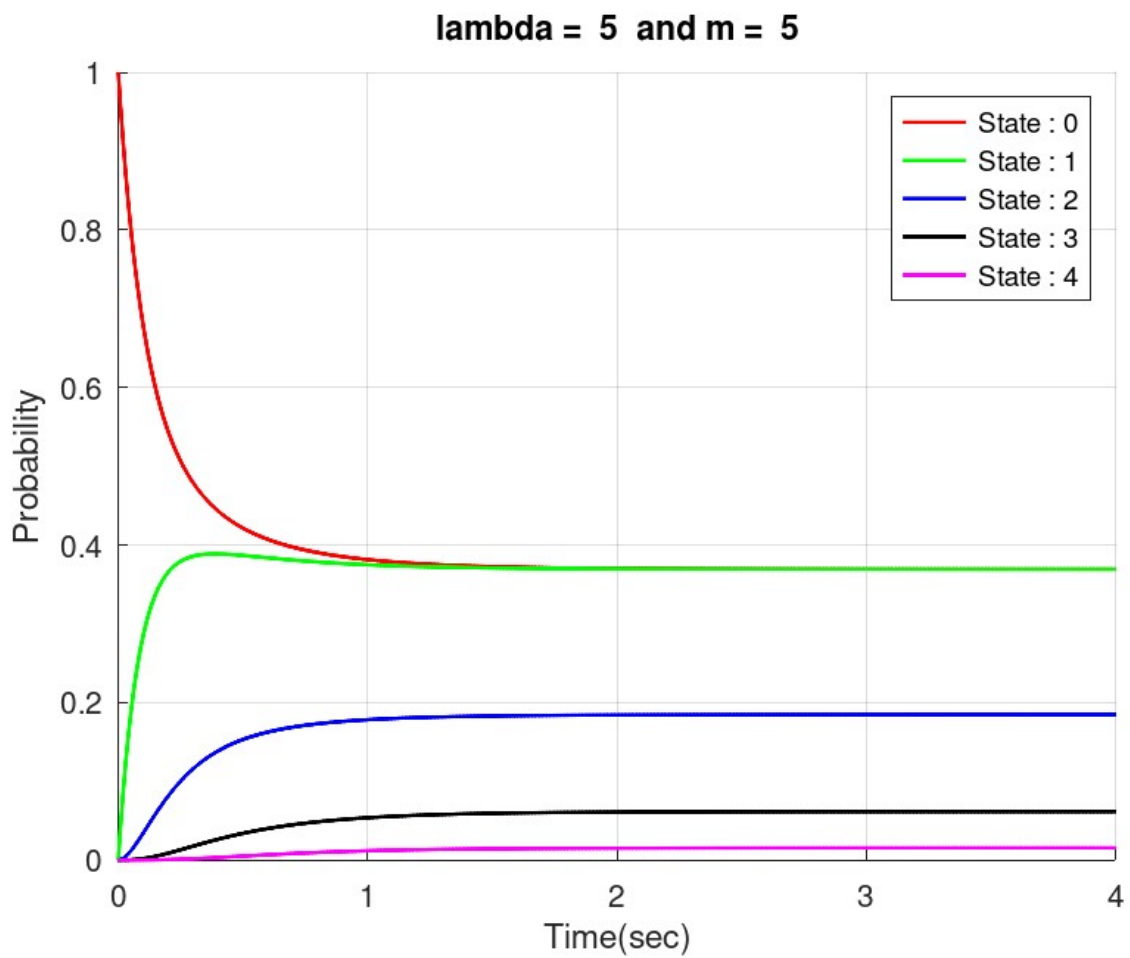
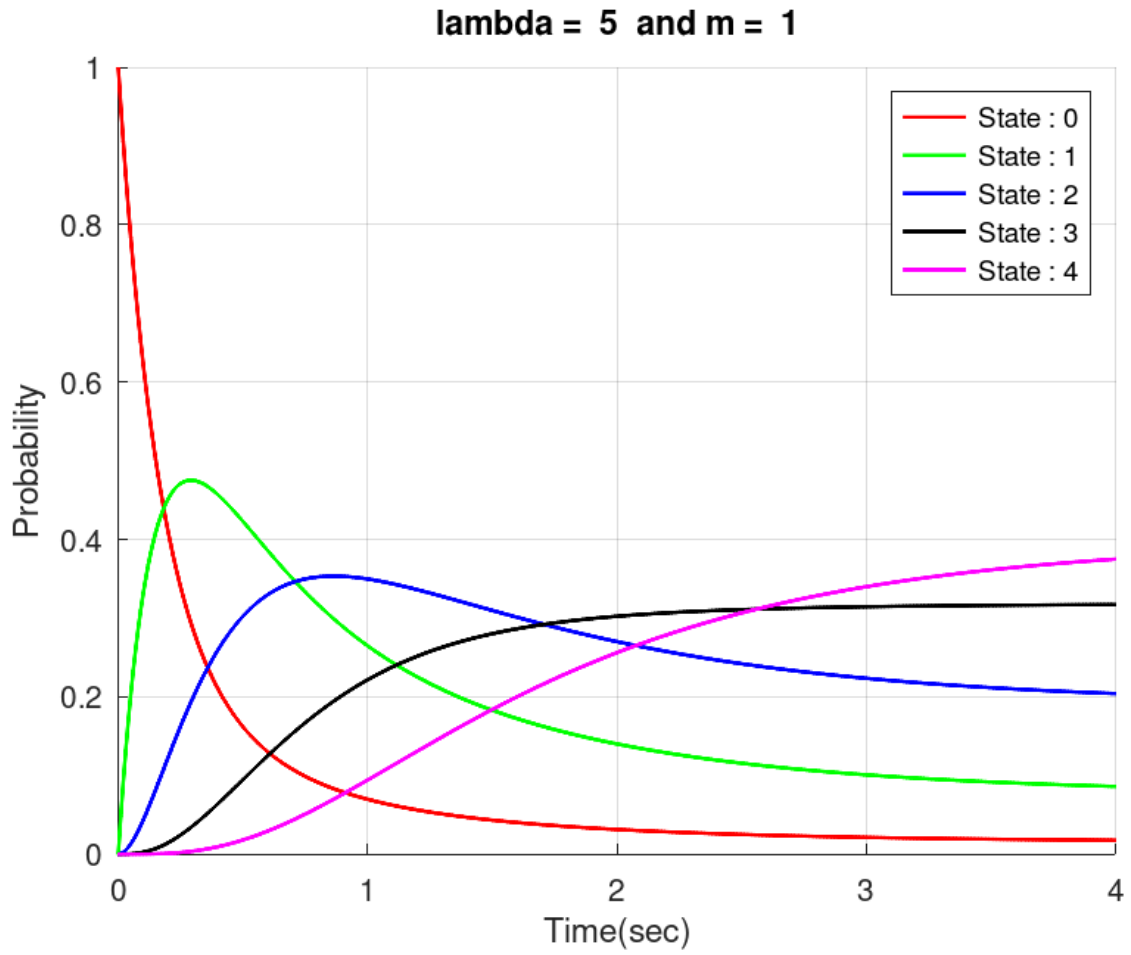
Average Number of customers in the system : $\sum k P_k = 0.4992$

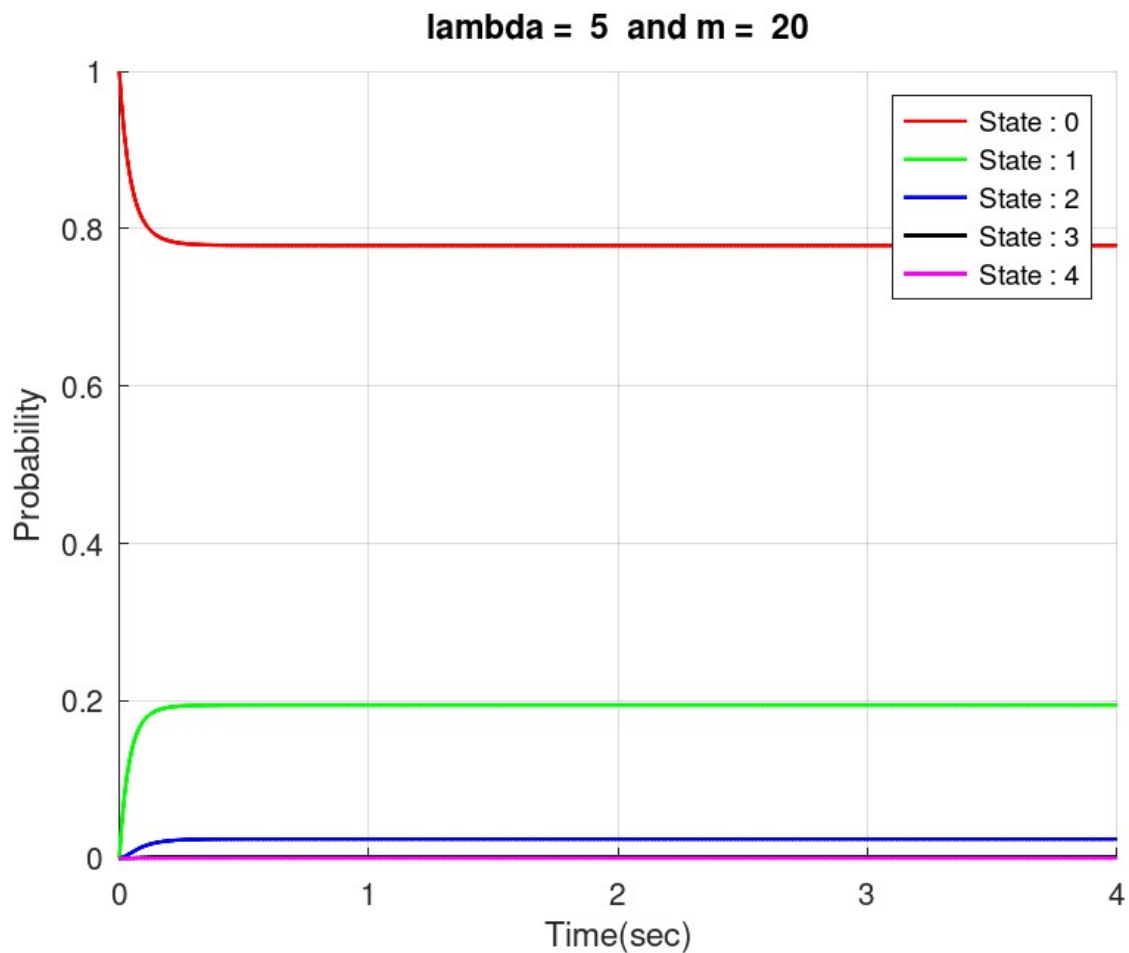
iv) $P_4 = P(\text{Blocking}) = 1.5798\text{e-}03$

v)



vi)





Όσο ο όρος ρ απομακρύνεται από το 1 τόσο ο χρόνος σύγκλισης των πιθανοτήτων μειώνεται καθώς και οι πιθανότητες για τις αρχικές καταστάσεις είναι πολύ μεγαλύτερες από τις επόμενες.

Κωδικας

```

clc;
clear all;
close all;

pkg load statistics
pkg load queueing

lambda = 5;
m = 10;
states = [0, 1, 2, 3, 4]; % system with capacity 4 states
% the initial state of the system. The system is initially empty.
initial_state = [1, 0, 0, 0, 0];

% define the birth and death rates between the states of the system.
births_B = [lambda, lambda/2, lambda/3, lambda/4];
deaths_D = [m, m, m, m];

% get the transition matrix of the birth-death process
transition_matrix = ctmcdb(births_B, deaths_D);
display (transition_matrix)

# (ii)
% get the ergodic probabilities of the system

```

```

P = ctmc(transition_matrix);
display (P);
figure(1);
hold on;
title("Bar of Probabilities per state")
xlabel("State")
ylabel("Probability")
bar(states, P, "b", 0.5);
grid on;
hold off;
# (iii)
display( " Average Number of customers in the system : ")
display( sum(P.*[0,1,2,3,4]))

# (iv)
display( " Probability of blocking a customer :")
display( P(5) )

%P[Blocking]

P_Blocking = P(5);
display(P_Blocking)

index = 0;
for T = 0 : 0.01 : 50
    index = index + 1;
    Po = ctmc(transition_matrix, T, initial_state);
    Prob0(index) = Po(1);
    Prob1(index) = Po(2);
    Prob2(index) = Po(3);
    Prob3(index) = Po(4);
    Prob4(index) = Po(5);
    if Po - P < 0.01
        break;
    endif
endfor

T = 0 : 0.01 : T;
figure(2);
title(strjoin({"lambda = ",num2str(lambda)," and m = ",num2str(m)}))
xlabel("Time(sec)")
hold on;
plot(T, Prob0, "r", "linewidth", 1.5);
plot(T, Prob1, "b", "linewidth", 1.5);
plot(T, Prob2, "k", "linewidth", 1.5);
plot(T, Prob3, "g", "linewidth", 1.5);
plot(T, Prob4, "m", "linewidth", 1.5);
legend("State : 0","State : 1","State : 2","State : 3","State : 4");
grid on;
hold on;

m = [1,5,20];
for i=1:columns(m)
    deaths_D = [m(i), m(i), m(i), m(i)];
    transition_matrix = ctmcdb(births_B, deaths_D);
    index = 0;

```

```

for T = 0 : 0.01 : 4
    index = index + 1;
    P0 = ctmc(transition_matrix, T, initial_state);
    Prob0(index) = P0(1);
    Prob1(index) = P0(2);
    Prob2(index) = P0(3);
    Prob3(index) = P0(4);
    Prob4(index) = P0(5);
    if P0 - P < 0.01
        break;
    endif
endfor

```

```

T = 0 : 0.01 : T;
figure(i+2);
title(strjoin({"lambda = ",num2str(lambda)," and m = ",num2str(m(i))}))
xlabel("Time(sec)")
ylabel("Probability")
hold on;
plot(T, Prob0, "r", "linewidth", 1.5);
plot(T, Prob1, "g", "linewidth", 1.5);
plot(T, Prob2, "b", "linewidth", 1.5);
plot(T, Prob3, "k", "linewidth", 1.5);
plot(T, Prob4, "m", "linewidth", 1.5);
legend("State : 0","State : 1","State : 2","State : 3","State : 4");
grid on;
hold off;
endfor

```