Aalto Univeristy

SCI-C0200 - Physics and Mathematics Studio

Computer Exercise Final Paper: Problem 12. Parking Lot Simulation

Johan Lindell, 708535, johan.lindell@aalto.fi

Returned: 01.04.2021

Contents

1 Introduction

Parking space availability and usage is an important aspect of building new and space-efficient parking lots. This paper focuses on simulating and analyzing arriving, departing and the total number of cars parked in a parking lot. More specifically we are simulating the parking lot at Aalto University. We will do this by implementing a Monte-Carlo mathematical model with Matlab.

The amount of cars arriving every hour is a normal distributed random variable with known mean and variance for every hour of the day. The departing cars is a random variable dependent on the earlier arrived cars. Thus we will first simulate the arriving cars on an hourly basis and after that compute the departing cars and total cars on the parking lot. The parking lot has a maximum capacity which is the limit for the total amount of cars parked. We will do this for two different scenarios A and B. In simulation A the parameters are the same for each day and in simulation B they are affected by a parking ticket inspector. The arrival of a ticket inspector will lower the mean and change the standard deviation of arriving cars as illegally parked student avoid using the lot.

From these simulations we can get important statistics about the cars in the parking lot. We can also analyze how a disturbance of a parking ticket inspector will change these statistics. Later we will also study the sensitivity of the outcome. All distributions and equations used in this paper are known and given in the exercise assignment [1].

2 Mathematical model

As our mathematical model we will implement two Monte-Carlo simulations for the different scenarios (with/without a ticket inspector).

The amount of arriving cars every hour of the day is a $(\lambda_i, \sigma^2_{\lambda_i})$ - normally distributed random variable:

Table 1: Arriving car parameters.

time	01	02	03	04	05	06	07	08	09	10	11	12
λ_i	3	3	3	3	3	3	10	20	60	40	30	20
σ_i	3	3	3	2	2	2	3	5	7	10	10	10
time	13	14	15	16	17	18	19	20	21	22	23	24
λ_i	40	20	30	14	13	15	15	10	8	3	3	3
σ_i	7	5	3	2	2	3	3	4	4	2	2	2

Table 1 shows the parameters for the arriving cars where λ_i is the mean and σ_i is the standard deviation at time i for the arriving cars.

The amount of cars leaving μ_t at time t is dependent upon the amount of arrived cars before t, this is expressed by the following equation:

$$\mu_t = (\sum_{i=1}^{k} p(i) l(t+i-k)) + e(t) \tag{1}$$

where k is a delay parameter, $l(i+t-k)$ is the amount of cars arriving at time $t+i-k$, $e(t)$ is a $(0, \sigma^2_\mu)$ - normally distributed error and $p(i)$ is a weighing factor where $p(i) = (0.7)^i / norm$ where $norm = \sum_{j=1}^{k} p(j)$.

For simulation B the arrival day of the ticket inspector is a $U(a, b)$ uniformly distributed random variable, where a is the first day of the month and b is the last day of the month. The arrival of the inspector will alter the parameters of the distribution shown in table 1 of the arriving cars for the next two days. In this simulation the altered values are $\lambda_i = 3, i = 1...24$ and $\sigma_{\lambda_i} = 2$.

3 Implementation

3.1 Simulation A

The implementation of this simulation is done with Matlab. Both simulations are implemented by iterating over the chosen 10000 days. For each day we start by calculating the arriving cars on an hourly bases. This is done by iterating over `max(0,normrnd(mean, std))` 24 times, which picks a random number from a normal distribution with given parameters with minimum value of 0 since negative amount of cars cannot arrive. These values are saved to an vector `arr`

After the arriving cars are calculated, the departing cars are calculated which is also done by iterating over 24 hours. The iteration is done over equation 1 and saved to the vector `dep`. The weighing factor `p(i)` is defined as a separate function with inputs `i` and `k`. It is also checked that the departing number of cars cannot be negative.

The total number of cars are calculated after this which is iterated over 23 hours. The iteration is done over the following function:

$$\texttt{tot(i + 1) = tot(i) + arr(i) - dep(i)} \tag{2}$$

where `i` is the time of day (hours), `tot` is the number of cars parked, `arr` are the arriving cars and `dep` are the departing cars. After equation 2 is executed a if statement checks if `tot(i + 1)` exceeds the total amount of parking spots. If this is exceeded, the value of `tot(i+1)` is changed to the parking spot limit and the arriving cars are changed to `arr(i) = tot(i+1) - tot(i)`. This way the upper boundary of the number of parked cars is kept and the arriving cars are changed to match this.

In the next step the arrays `tot`, `arr` and `dep` are appended to matrices which contain all the 10000 days of simulations, these matrices will be of the shape 24x10000 where the columns are hours of the day and rows are the individual days.

After every day `tot(1)` $=$ `tot(24)` is initiated. This way the total number of cars in one day is carried over to the next.

3.2 Simulation B

Simulation B is done the same way as simulation A except for the arriving cars which is altered if a ticket inspector arrives. After the number of cars parked is calculated we check if a ticket inspector has arrived that day. The randomness of the ticket inspector arriving is implemented with `1 == unidrnd(30,1,1)` this statement is true with the probability of $1/30$. This simulates the $U(1, 30)$ uniformly distributed random variable of the ticket inspector arriving. If the statement is true it changes the value of a variable `ll` to 2. When calculating the arriving cars the value of `ll` is checked and if it is greater than 0 it changes the mean value and standard deviation in the `normrnd` function. Every day `ll` is decreased by 1 which means after 2 days it will no longer have an effect on arriving cars.

4 Outcomes and sensitivity analysis

4.1 Outcomes

I have chosen to visualize the outcomes as histograms where the x-axis is the time in hours and the y-axis is the amount of cars. The first histogram shows the mean values for the amount of cars for each hour of a day over the 10000 days. The second histogram shows the first 30 days of the simulation. I have chosen to show the first 30 days as this shows the effect that the parking ticket inspector will have. I have also calculated the mean, median, standard deviation over all the values for arriving, departing and parked cars. I have also chosen to analyze the differences between simulation A an B.

4.1.1 Simulation A

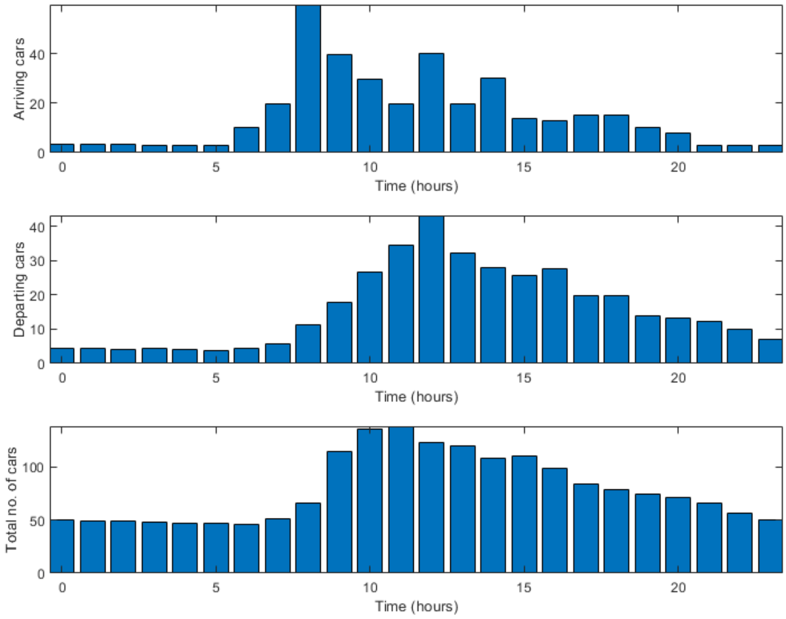

Figure 1: Mean values for each hour of the day.

The numbers in figure 1 are as expected given the parameters for the simulation. To get these values I have taken the mean values of the columns in the 24x10000 matrices that contain all the values of the simulations. The total number of cars stay well within the limits of the parking lot, although individual values do go up to 200.

Table 2: Statistics for simulation A.

Statistic	Arriving cars	Departing cars	Tot. no. cars
Mean	15.41	15.79	75.10
Std.	15.46	12.56	47.59
Median	11.45	13.30	73.16
Max	86	66	200

Table 2 shows the means, standard deviations and maximum values for all the values in the simulation. I have done this by flattening the 24x10000 matrices and calculated the mean, max values and standard deviation of these vectors. I have not shown the minimum values as these are all zero.

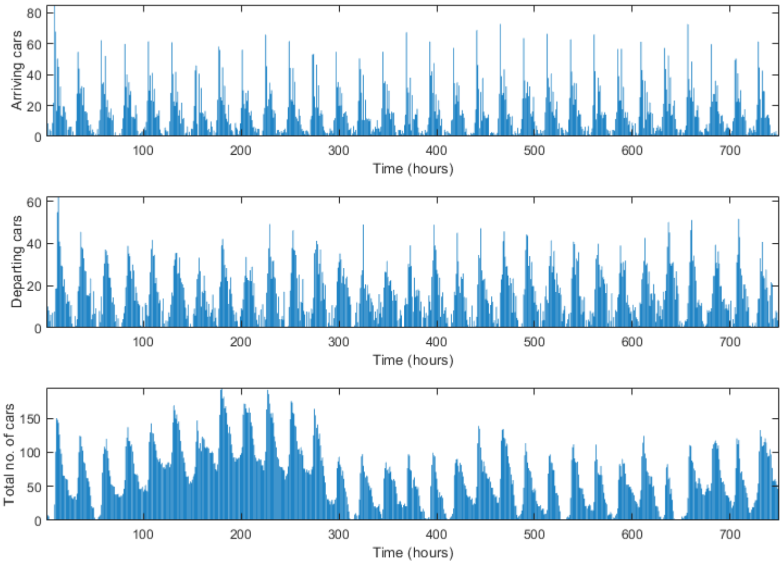

Figure 2: Amount of cars for the first 30 days.

Figure 2 shows the first 30 days of the simulation and gives a good idea of how the amount of cars behave.

4.1.2 Simulation B

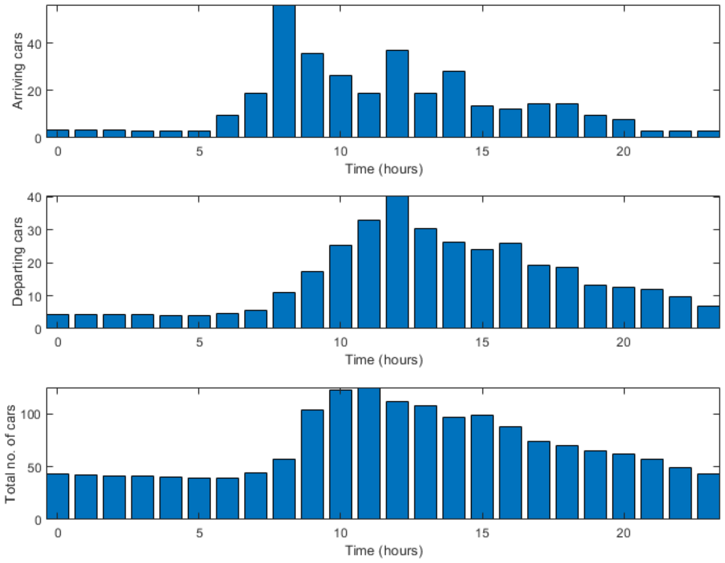

Figure 3: Mean values for each hour of the day.

Table 3: Statistics for simulation B.

Statistic	Arriving cars	Departing cars	Tot. no. cars
Mean	14.56	14.99	69.55
Std.	15.14	12.53	47.88
Median	9.98	12.10	66.36
Max	86	66	200

The histograms in figure 3 looks very familiar to figure 1 in simulation A. However when we compare tables 2 and 3 we see that the means are noticeably lower in simulation B. Notable here is that even though the mean values are lower the maximum values are the same. Table 3 values are calculated with the same method as simulation A statistics.

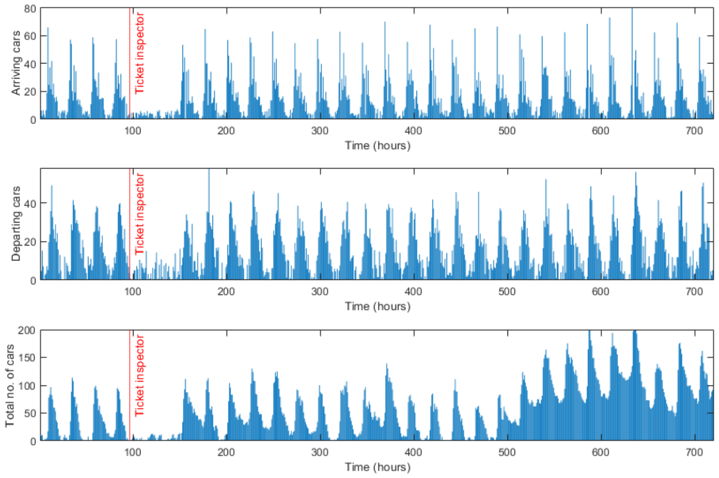

Figure 4: Amount of cars for the first 30 days.

Figure 4 is quite similar to figure 2, here however we can clearly see the effect of the ticket inspector. As the mean and standard deviation are constant and lower than usual the parking spot is empty or has very few cars. This is due to cars not arriving which also has an effect on cars departing. The red line seen in figure 4 shows when the parameters change due to the ticket inspector.

4.1.3 Comparing A and B

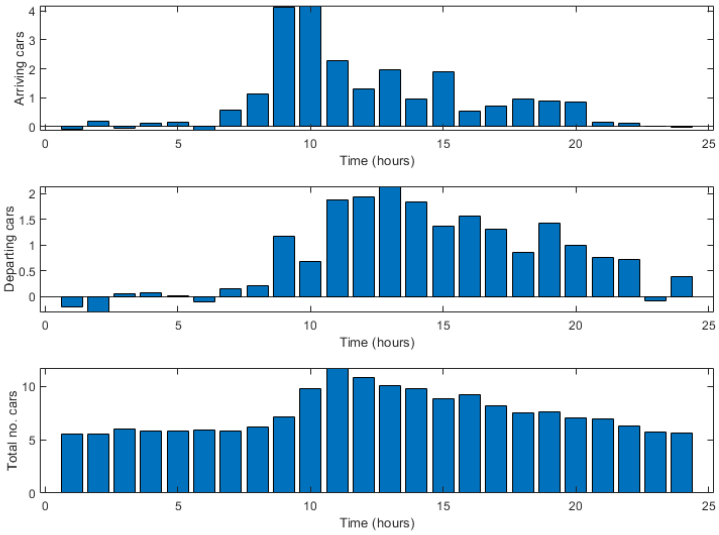

Figure 5: Differences between figure 1 and 3

From figure 5 we can see that the mean value for most of the values is higher in simulation A compared to simulation B which is expected as almost all of the parameter mean values for arriving cars are higher than in simulation B due to the parking ticket inspector.

4.2 Sensitivity analysis

From figure 4 we can clearly see that the simulation gets affected by the changes in the parameters. When the ticket inspector affects the parameters the parking lot is almost empty all the time. This also shows that to get an accurate simulation of reality it is key to have accurate parameter values. This model only takes into account parking ticket inspectors but there are many other factors which would affect the outcomes. This would for instance include: weather and the date (for example is it a week day or weekend).

5 Conclusions and reflections

In reality this model would be too simple to accurately simulate reality. This is due to many assumptions. It is also unclear how accurate the parameters in table 1 are since this is not specified in the exercise assignment [1]. Some numbers however are usable, for instance the parking ticket inspector did not affect the maximum and minimum values for arriving, departing and parked cars. Thus this simulation would have some use in planning a parking spot. The frequency of maximum values however changes as the parameters change. This could lead to inefficiencies in the planning of a parking lot.

However the methods used in this paper are well suited for this problem. Thus if less assumptions where used and the accuracy of the parameters was verified this could have applications in real cases.

6 Source code

6.1 Simulation A

```
%SCI-C0200
%EXERCISE 12. PARKING SPOT SIMULATION
%SIMULATION A
%no. of simulations
sims=10000;

%mean values for arriving cars kl. 00-24
lambda = [3 3 3 3 3 3 10 20 60 40 30 20 40 20 30....
    14 13 15 15 10 8 3 3 3];

%std values for arriving cars kl. 00-24
sigma = [3 3 3 2 2 2 3 5 7 10 10 10 7 5 3....
    2 2 3 3 4 4 2 2 2];
```

```
%initiating vectors

%no. of cars
tot = zeros(1,24);
%arriving cars
arr = zeros(1,24);
%departing cars
dep = zeros(1,24);


%initiating matrices where data will be stored
%rows are data for each simulation
%columns are hours in day
tot_data=zeros(sims,24);
arr_data = zeros(sims,24);
dep_data=zeros(sims,24);


%parameters and starting values

%delay paramteer
k=5;

%error standard deviaiton
sigmamu=5;

%parking lot limit
lim = 200;

%no. of cars at 00:00
tot(1) = 10;

for run=(1:sims)

%arriving cars calculation
for i =(1:24)
   arr(i) = max(0, normrnd(lambda(i), sigma(i)));
end


%departing cars calculation
for t=(1:24)

    %sum for dep term
    summa=0;
    for i=(1:k)
        summa = summa + p(i,k) * arr(max(1,t + i - k));
    end

    %calculating error term
    dep(t) = max(0,summa + normrnd(0,sigmamu));
end


%total no. of cars calculation
for i=(1:23)
    tot(i+1) = min(200,max(0,tot(i) + arr(i) - dep(i)));

    % if parking lot limit is reached
    if tot(i+1) > lim
        tot(i+1) = lim;
        arr(i) = max(0, tot(i+1) - tot(i));
    end

end

%appending calculated data to matrices
dep_data(run,:)=dep;
tot_data(run,:)=tot;
arr_data(run,:)=arr;

%this days last car amount equals next days first car amount
tot(1) = tot(end);

end

%plotting first month of simulations
%flatting arriving, departing and total no. of cars matrices
flat_arr = reshape(arr_data.',1,[]);
flat_dep = reshape(dep_data.',1,[]);
flat_tot = reshape(tot_data.',1,[]);
figure
subplot(3,1,1)
bar((1:25*30),flat_arr(1:750))
hold on
hold off
xlabel("Time (hours)")
ylabel("Arriving cars")
```

```
axis tight
subplot(3,1,2)
bar((1:25*30),flat_dep(1:750))
hold on
hold off
xlabel("Time (hours)")
ylabel("Departing cars")
axis tight
subplot(3,1,3)

bar((1:25*30),flat_tot(1:750))
hold on
hold off
xlabel("Time (hours)")
ylabel("Total no. of cars")
axis tight



%plotting avg. for one day
figure
subplot(3,1,1)
bar((0:23),mean(arr_data))
xlabel("Time (hours)")
ylabel("Arriving cars")
axis tight
subplot(3,1,2)
bar((0:23),mean(dep_data))
xlabel("Time (hours)")
ylabel("Departing cars")
axis tight
subplot(3,1,3)
bar((0:23),mean(tot_data))
xlabel("Time (hours)")
ylabel("Total no. of cars")
axis tight
```

6.2 Simulation B

```
%SCI-C0200
%EXERCISE 12. PARKING SPOT SIMULATION
%SIMULATION B
%no. of simulations
sims=10000;
```

```
%mean values for arriving cars kl. 00-24
lambda = [3 3 3 3 3 3 10 20 60 40 30 20 40 20 30....
    14 13 15 15 10 8 3 3 3];

%std values for arriving cars kl. 00-24
sigma = [3 3 3 2 2 2 3 5 7 10 10 10 7 5 3....
    2 2 3 3 4 4 2 2 2];


%initiating vectors

%no. of cars
tot = zeros(1,24);
%arriving cars
arr = zeros(1,24);
%departing cars
dep = zeros(1,24);


%initiating matrices where data will be stored
%rows are data for each simulation
%columns are hours in day
tot_data=zeros(sims,24);
arr_data = zeros(sims,24);
dep_data=zeros(sims,24);


%parameters and starting values

%delay parameter
k=5;
%error standard deviation
sigmamu=5;

%parking lot limit
lim = 200;

%days left that are affected by ticket inspector
ll = 0;
%array containing days when ticket inspector arrived
ll_paivat = [];
%no. of cars 00:00
tot(1) = 10;
```

```
for run=(1:sims)

%arriving cars calculation
for i =(1:24)
   if ll == 0
       %if ticket inspector has no effect
       arr(i) = max(0, normrnd(lambda(i), sigma(i)));
   else
       %if days after ticket inspector is less than 2
       arr(i) = max(0, normrnd(3, 2));
   end

end

%departing cars calculation
for t=(1:24)

     %sum for dep term
    summa=0;
    for i=(1:k)
        summa = summa + p(i,k) * arr(max(1,t + i - k));
    end

    %calculating error term
    dep(t) = max(0,summa + normrnd(0,sigmamu));
end


%total no. of cars calculation
for i=(1:23)
    tot(i+1) = max(0,tot(i) + arr(i) - dep(i));

    % if parking lot limit is reached
    if tot(i+1) > lim
        tot(i+1) = lim;
        arr(i) = max(0, tot(i+1) - tot(i));
    end
end

%appending calculated data to matrices
ll = max(0,ll - 1);
dep_data(run,:)=dep;
tot_data(run,:)=tot;
```

```
arr_data(run,:)=arr;

%Ticket inspector arrives this day with prbability of 1/30
if 1 == unidrnd(30,1,1)
    %saves which hour the ticket inspector arrives on
    ll_paivat(end + 1) = run * 24;
    ll = 2;
end
%this days last car amount equals next days first car amount
tot(1) = tot(end);
end

%plotting first month of simulations
%flatting arriving, departing and total no. of cars matrices
ll_paivat = ll_paivat(ll_paivat(1,:) <= 30*24);
flat_arr = reshape(arr_data.',1,[]);
flat_dep = reshape(dep_data.',1,[]);
flat_tot = reshape(tot_data.',1,[]);
figure
subplot(3,1,1)
bar((1:24*30),flat_arr(1:720))
hold on
%plotting x-lines when ticket inspector starts affecting data
for i=1:length(ll_paivat)
    xline(ll_paivat(i), '-r','Ticket inspector')
end
hold off
xlabel("Time (hours)")
ylabel("Arriving cars")
axis tight
subplot(3,1,2)
bar((1:24*30),flat_dep(1:720))
hold on
%plotting x-lines when ticket inspector starts affecting data
for i=1:length(ll_paivat)
    xline(ll_paivat(i), '-r','Ticket inspector')
end
hold off
xlabel("Time (hours)")
ylabel("Departing cars")
axis tight
subplot(3,1,3)

bar((1:24*30),flat_tot(1:720))
```

```
hold on
%plotting x-lines when ticket inspector starts affecting data
for i=1:length(ll_paivat)
    xline(ll_paivat(i), '-r','Ticket inspector')
end
hold off
xlabel("Time (hours)")
ylabel("Total no. of cars")
axis tight


%same plot as in part A
figure
subplot(3,1,1)
bar((0:23),mean(arr_data))
xlabel("Time (hours)")
ylabel("Arriving cars")
axis tight
subplot(3,1,2)
bar((0:23),mean(dep_data))
xlabel("Time (hours)")
ylabel("Departing cars")
axis tight
subplot(3,1,3)
bar((0:23),mean(tot_data))
xlabel("Time (hours)")
ylabel("Total no. of cars")
axis tight
```

6.3 P-norm function

```
function out = p(i,k)

%p norm calculation
p_norm=0;
for j=(1:k)
    p_norm = p_norm + 0.7^j;
end

%p calculation
out = 0.7^i / p_norm;
end
```

References

[1] *SCI-C0200 Tietokoneharjoitustyö, Tehtävä 12. Pysäköintipaikan simulointi.* URL: `https://mycourses.aalto.fi/pluginfile.php/1402336/mod_folder/content/0/Teht%5C%C3%5C%A4v%5C%C3%5C%A412.pdf`. (accessed: 09.05.2021).