Revised Version of Homework 2

CS626 Data Analysis and Simulation

Name: Sidi Chang

Due: 12:30 p.m., Tuesday, 5/31

1 Question 1

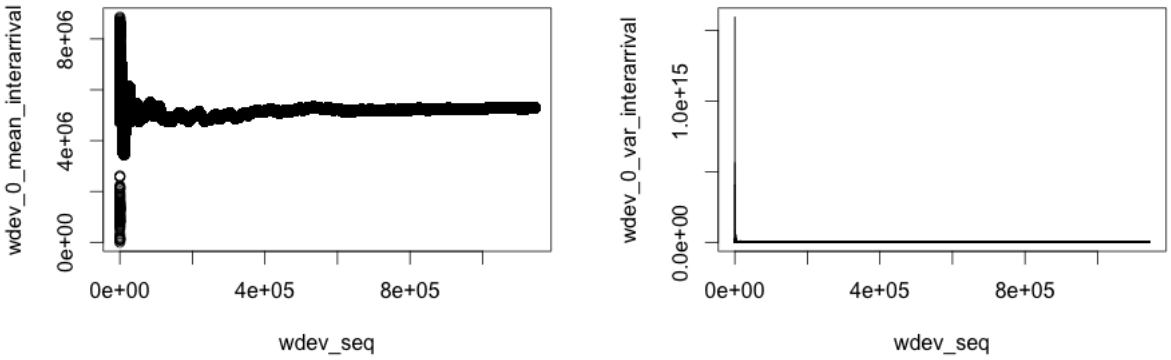
1.1 The first trace: $wdev_0.csv$

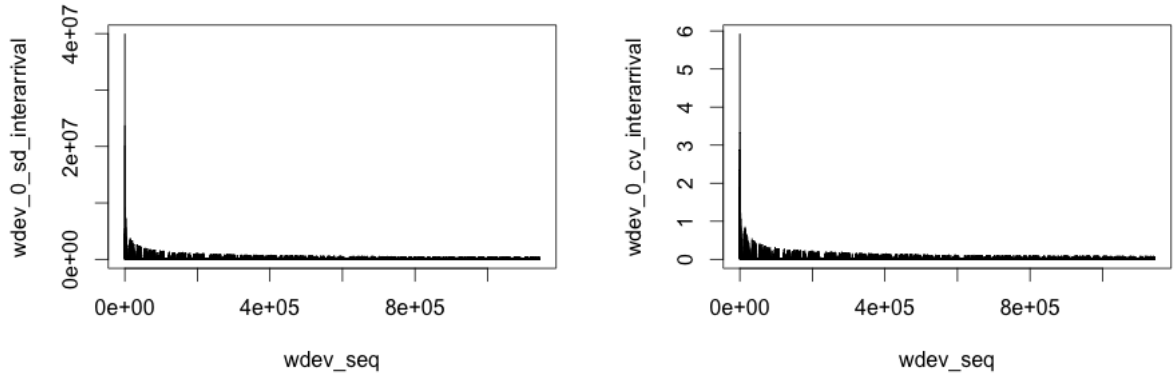
(a). [**inter arrival**] If I follow the equation from the slides shown $r_i = a_i - a_{i-1}$, I will start from the first number as 0 since 1.281664e+17 is too big. Then using R to do the calculation, we can find that:

Table 1: metrics for $wdev_0$

Mean	Var	SD	C.V.
5290159	5.767035e+14	24014653	4.539496

The transient of the mean is below:

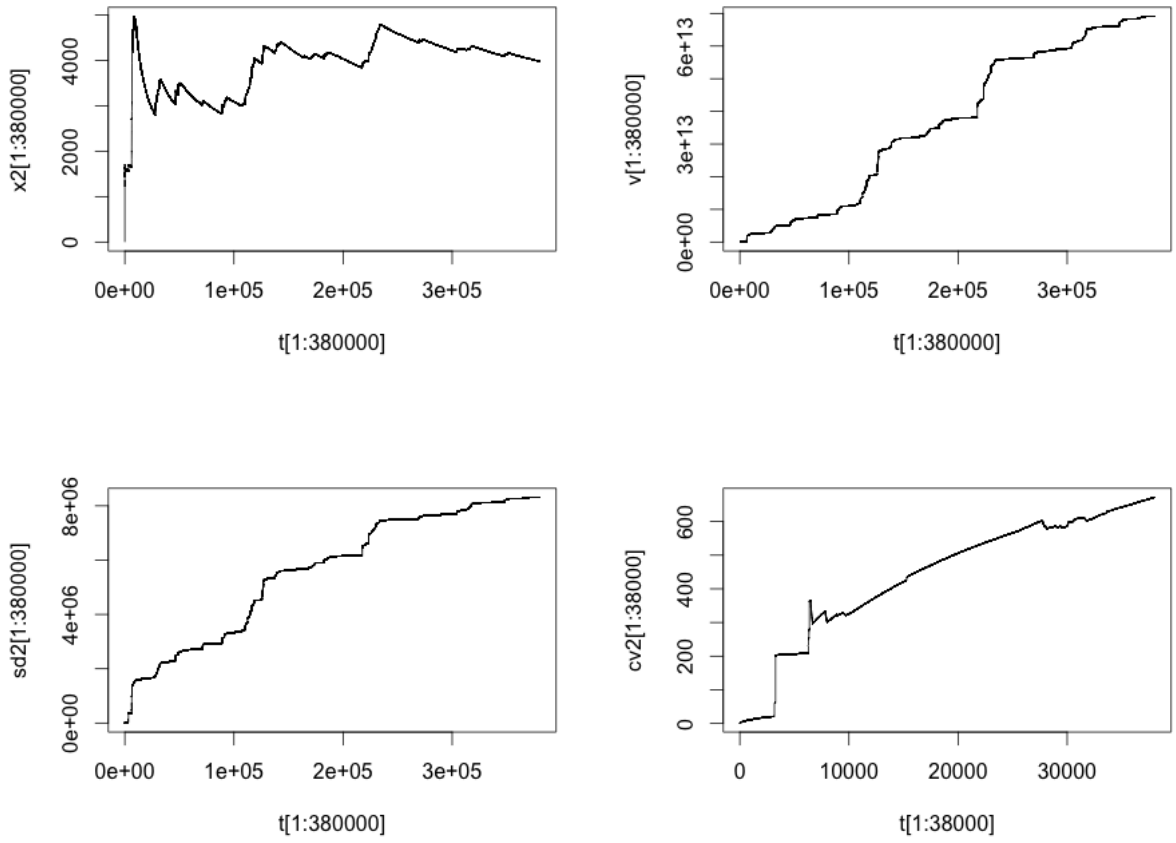




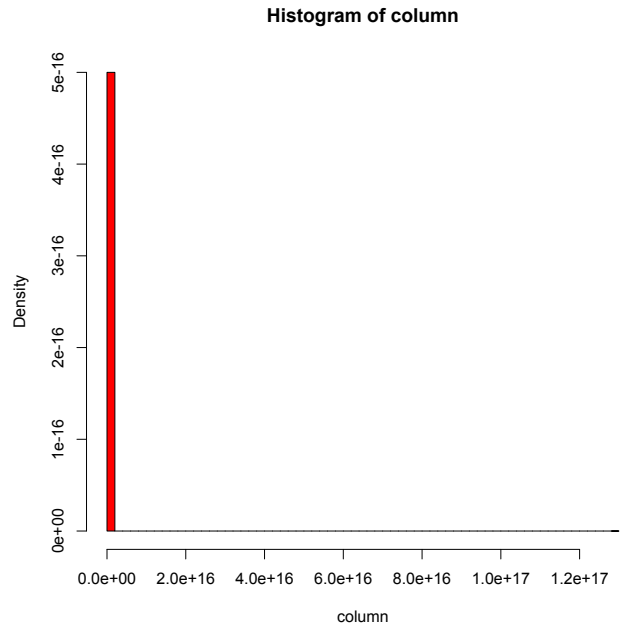
[**service time**]

Mean	Var	SD	C.V.
3649.102	295778230	17198.2	4.712995

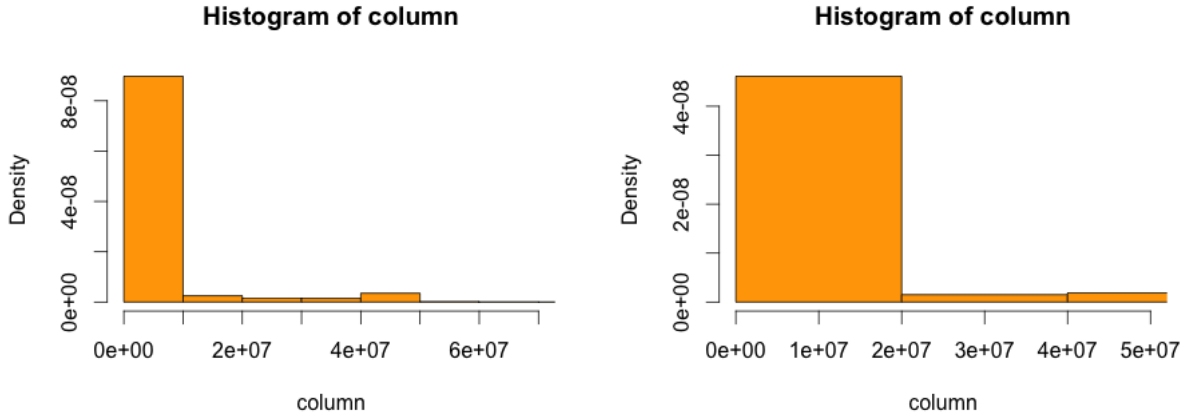
The graph is as blow:



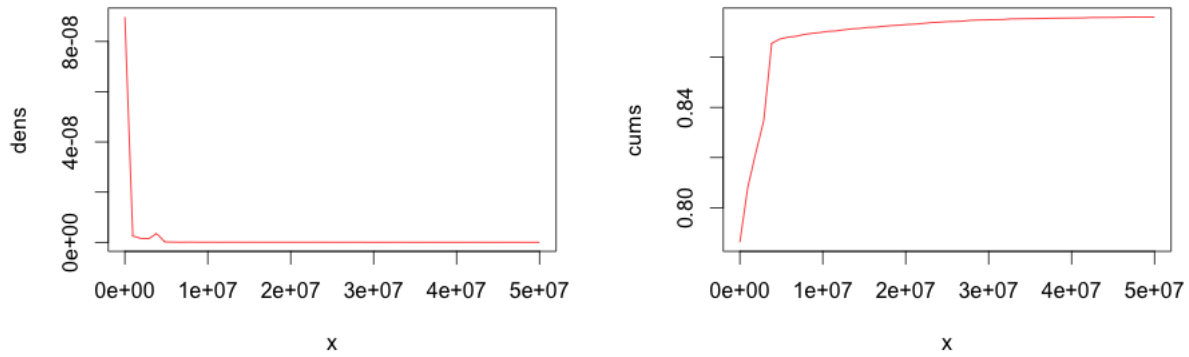
(b). First, I draw the histograms of the whole graph and it is blow using bins 60:



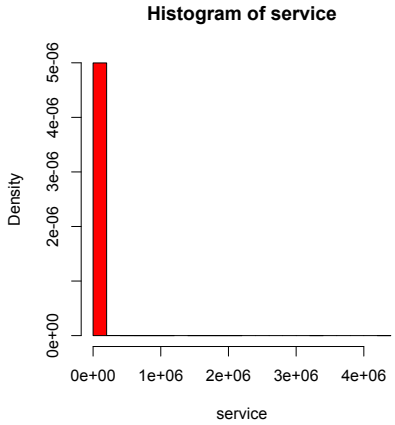
We can see the graph is so ugly and there is no meaning to draw pdf and cdf for this graph, then I find a solution, I get rid of the first big data and also just make the data restricted to (0,50000000) which contains 90% of the numbers. Then I draw the graph with different bins:



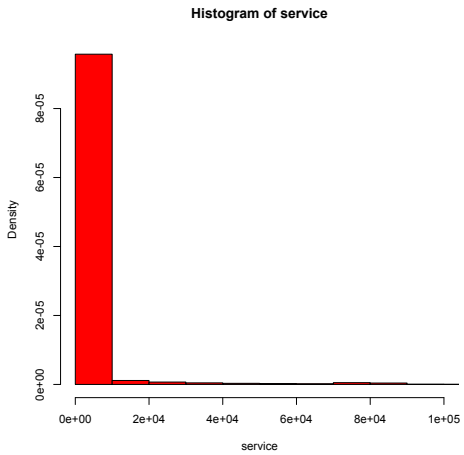
The first bin is 60 and the second bin is 40, we can see 60 is better. Then I draw the pdf and cdf of inter-arrival:



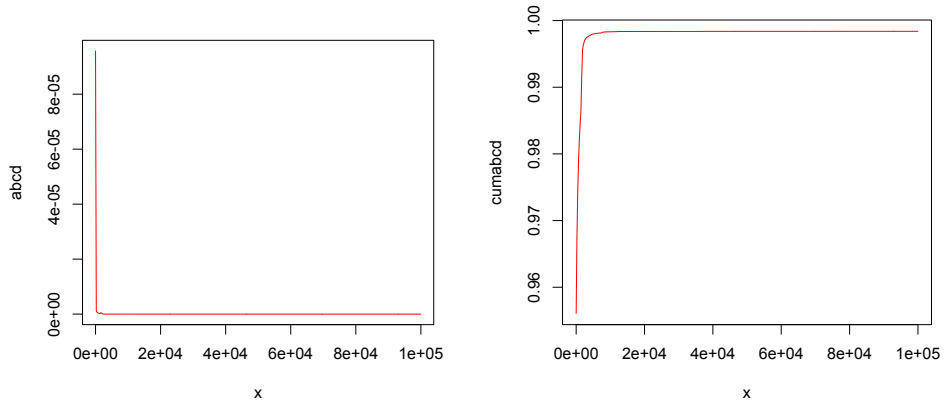
Then I also do something for the service time, first I draw the graph for it to see what will happen:



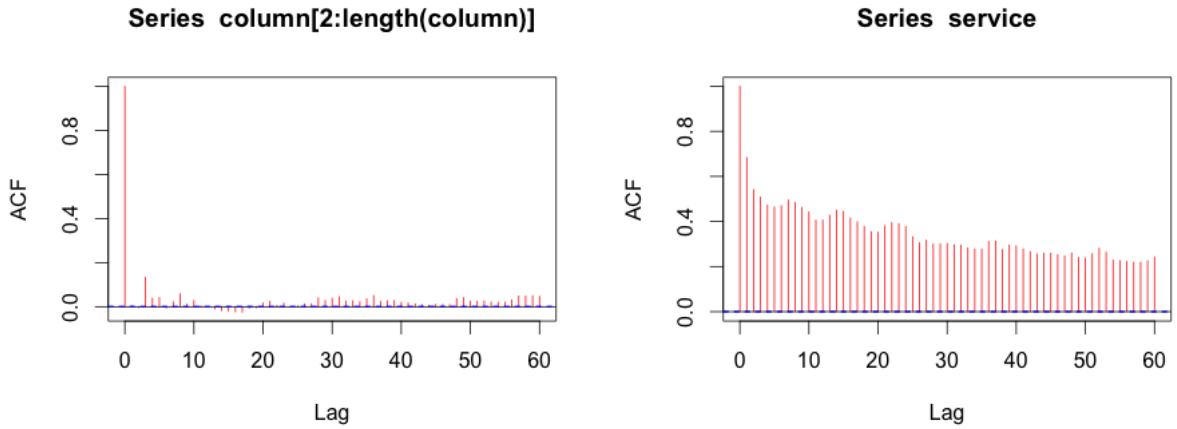
Also we can see the graph is terrible like this. So I also do some cut and only take (0,100000) and bins=330, we can get histogram below:



Then we can get the pdf and cdf of the service time:

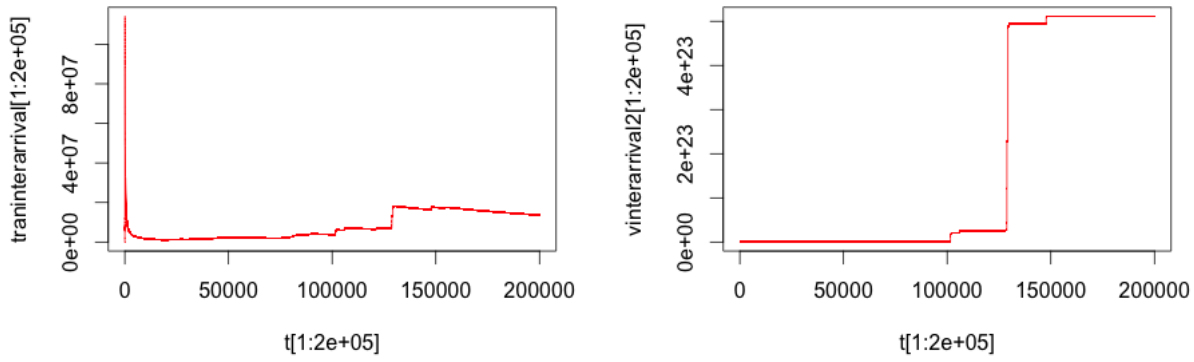


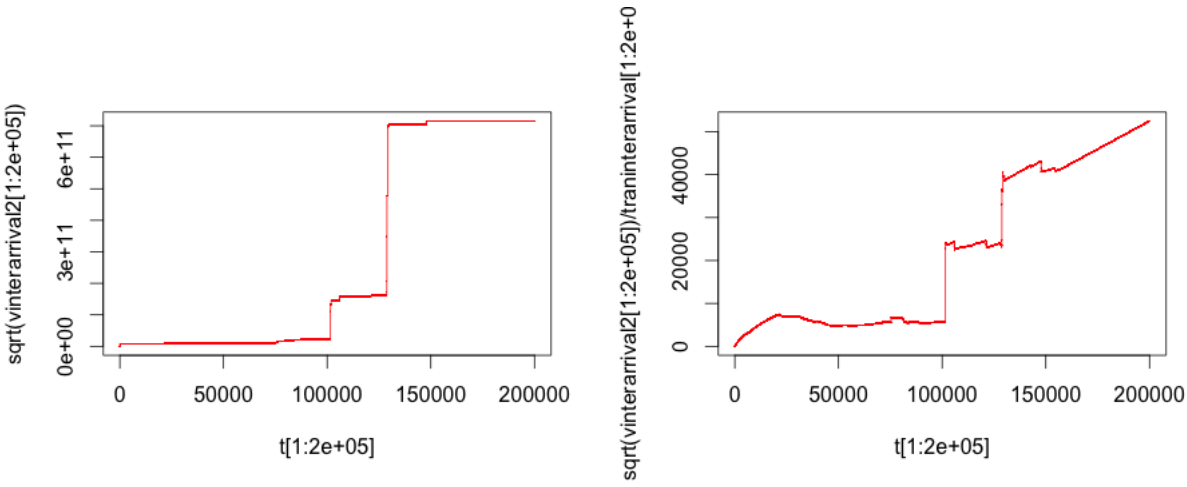
[**Autocorrection function**] There is a function in R called acf:



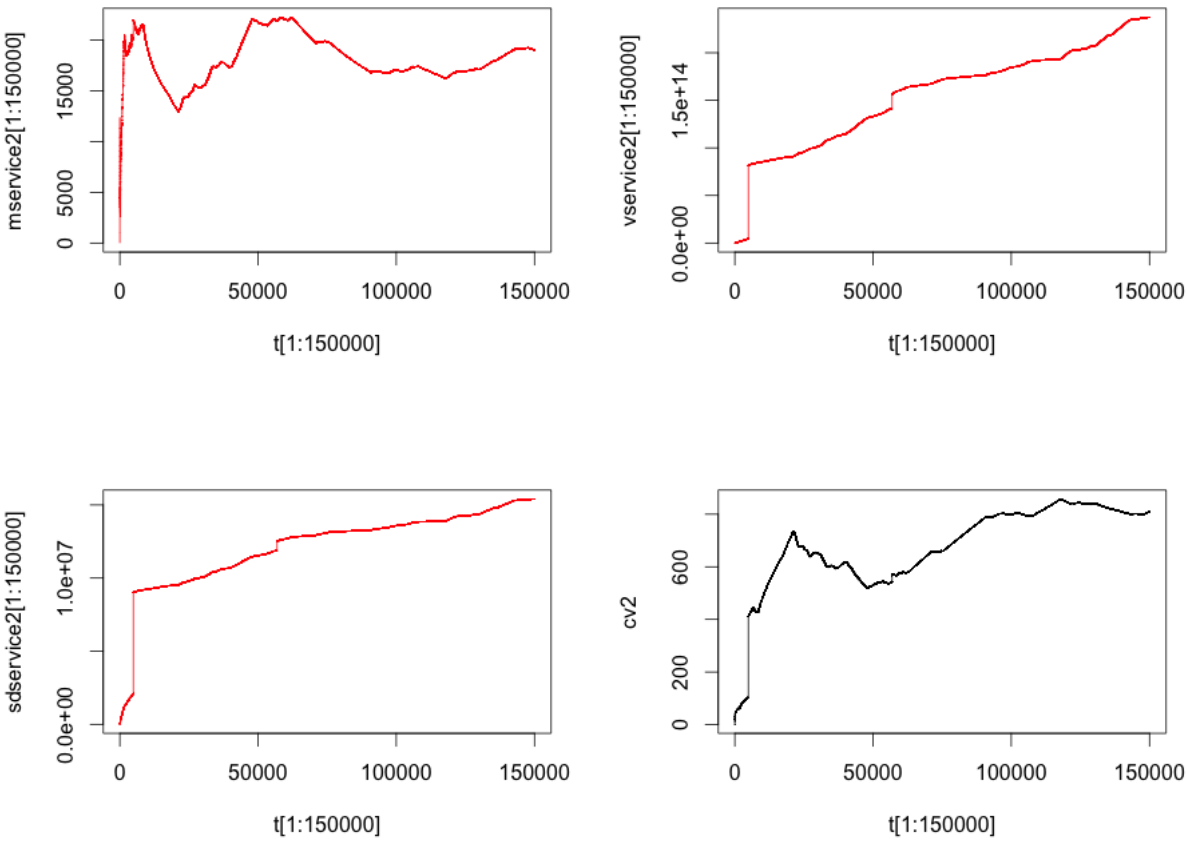
1.2 Trace 2: $Proj_3$

(a) First the mean, var, SD and C.V. As I learned from the first trace, I decided to start with arrival[1], which means when the first arrival comes, I start to record the time.
Inter-arrival Time: Mean is 3664617, Var is 5.9327e+17, SD is 770240222, C.V. is 210.18. Then the transient measure:

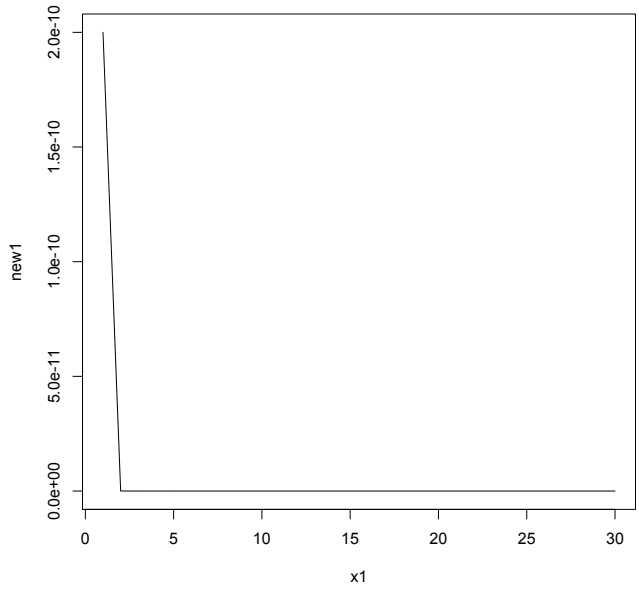


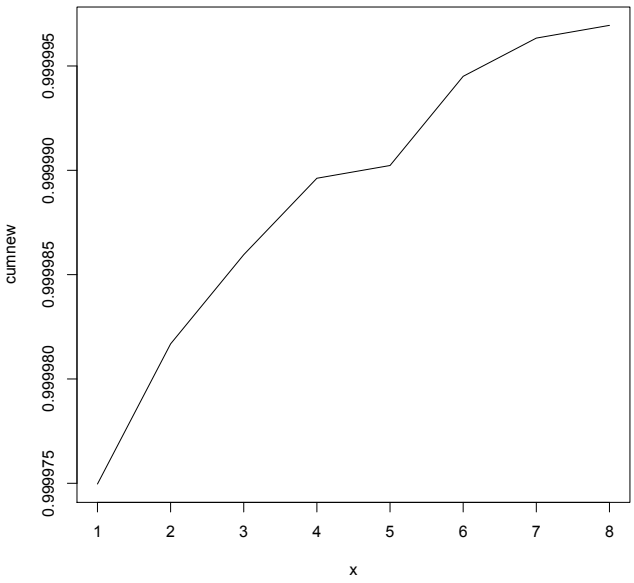


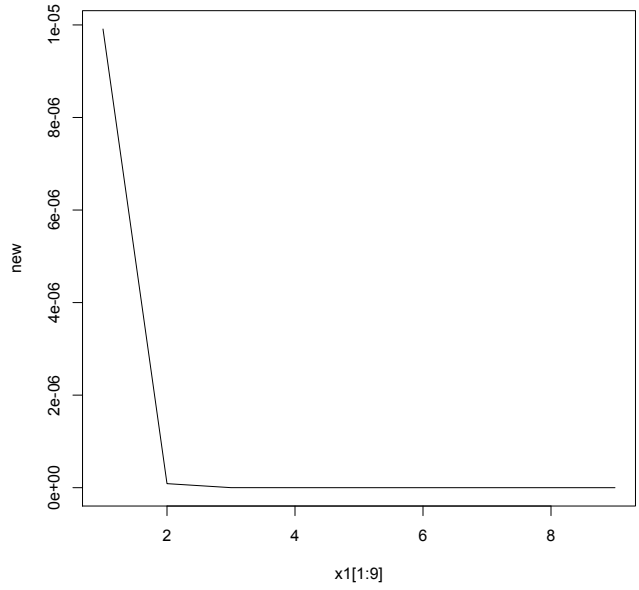
Service time: Mean is 13247, Var is 572198941, SD is 23921, C.V. is 1.8058. The transient measure is as below:

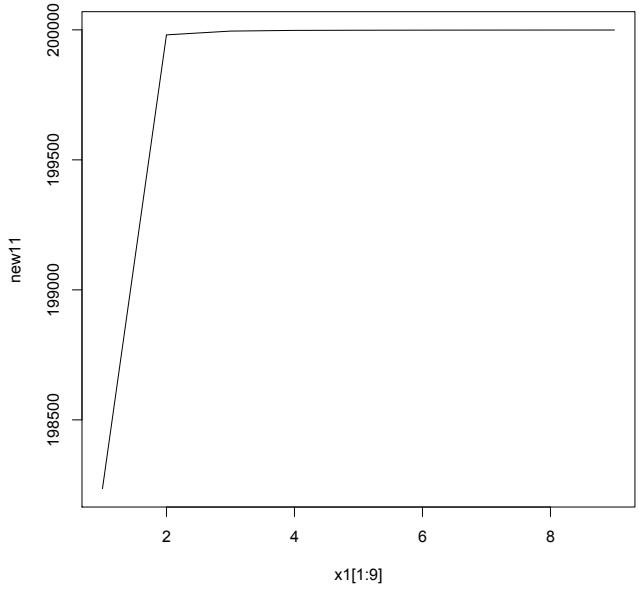


(b) It is as shown below: (Above inter-arrival, below service)

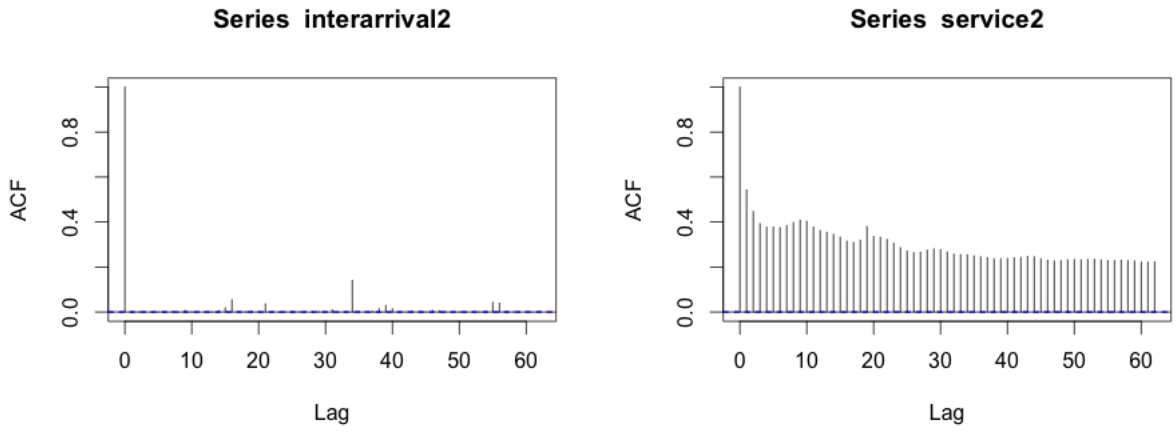

new1
0.0e+00
5.0e-11
1.0e-10
1.5e-10
2.0e-10
0
5
10
15
20
25
30
x1


cumnew
0.999975
0.999980
0.999985
0.999990
0.999995
1
2
3
4
5
6
7
8
x


new
0e+00
2e-06
4e-06
6e-06
8e-06
1e-05
2
4
6
8
x1[1:9]


new11
198500
199000
199500
200000
2
4
6
8
x1[1:9]

(c) As below:



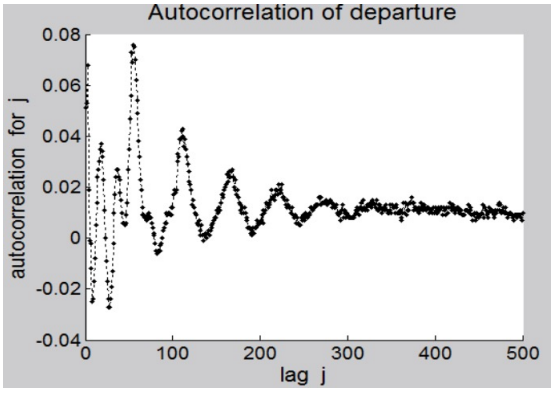
2 Question 2

2.1 Several metrics

Firstly, I have to make the program starts at 0 cause the first number is too big, then retrieve the first 50,000 data sets out to do this question(Increase the efficiency of computing). I use the program ssq1.c and acs.c to get the result.

Table 2: metrics for $wdev_0$

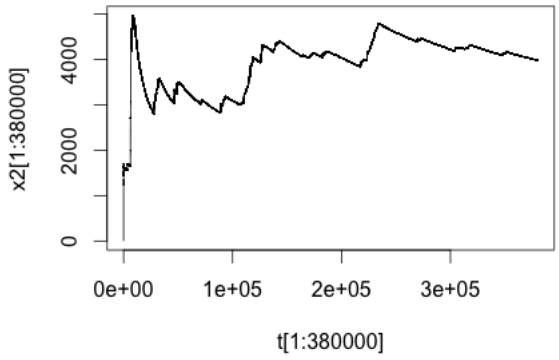
Mean delay time	system utilization	Average waiting queue length
69209.26	0.0058	0.002424



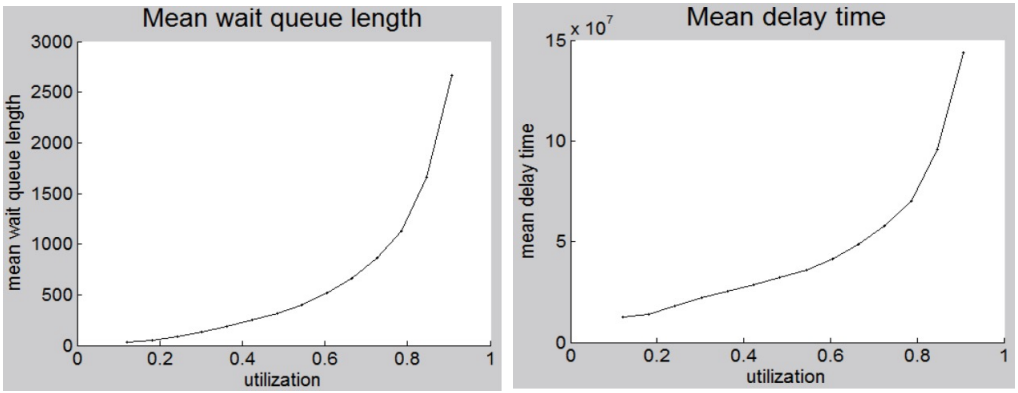
(b) I found a code on the website from Richmond University, he made the C code into R. Running his R program, I got the data below:
The original service time is as below, I just use the data from 1:380000 and got the graph:

Table 3: metrics for $wdev_0$ with exponential

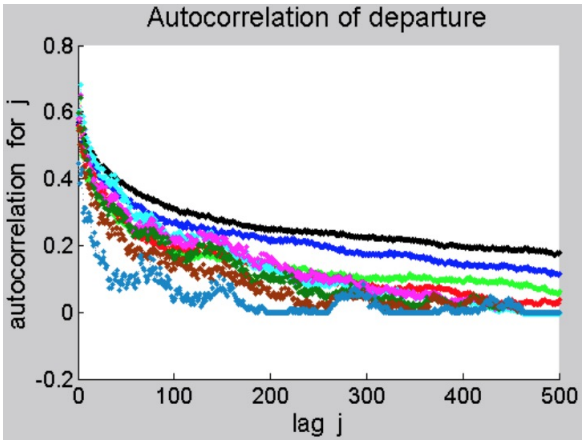
Mean delay time	system utilization	Average waiting queue length
76491.77	0.0058	0.0032



(c) I will choose the data from 100 to 100000, I will firstly show the data with trace data, then show the graph with exponential service times in the last question. So the graph is below:



Then in R, with the command hold on, we can plot several lines in one graph together, then I plot the line for utility 0.1-0.9 which is shown below:



Comment:

1. I found that using exponential service time, we can find that the system utility in the beginning has higher system usage, then it just goes down and in the end to a stable value. (This is the end of the first trace)

2. Using exponential service time and arrival time, obviously the delay time is smaller than the original track.

3. From the autocorrelation of departure figure, we can find out that when j becomes larger, the autocorrelation becomes smaller but always bigger than 0.