STA240 Final Project

Anthony Zhao, Abby Li, William Yan

Introduction

In 2024 alone, the restaurant industry was predicted to reach $1 trillion in sales; however, with rising costs and increased competition, owners must find innovative ways to keep up profit (National Restaurant Association, 2024). With increased technology, many restaurants are turning to data to provide insights into operations. Therefore, in this project, we will be analyzing performance metrics in three different scenarios to inform our decision on profit maximization. The goal for this project is three-fold: to utilize probabilistic modeling to optimize restaurant operations, to analyze customer behavior to maximize consumer satisfaction, and to provide business recommendations by considering key performance metrics such as waiting times, the number of chefs, and more. In the following scenarios, we assume the restaurant operates between the hours of 10am - 10pm.

Scenario 1

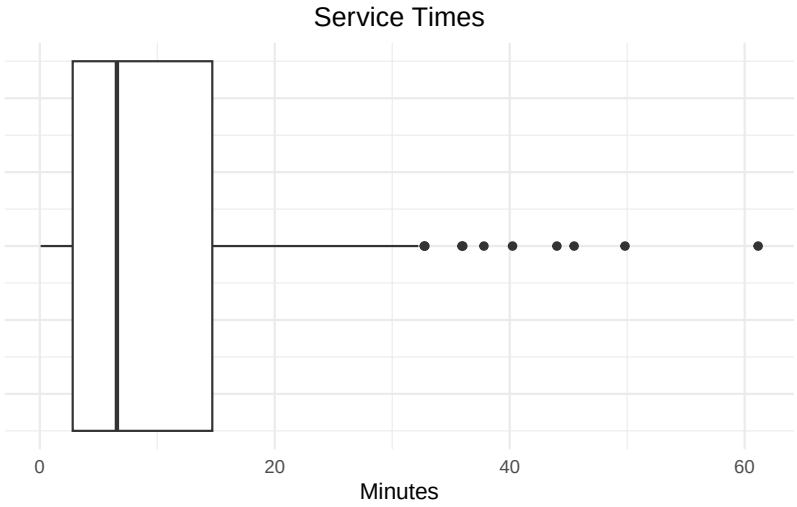
The first scenario considers a setting with 1 table and 1 chef. Customer arrival will be modeled to $W_k \sim Pois(\lambda)$ with $\lambda = 5$ and customer service time will be modeled by $S_k \sim Exp(\lambda)$ where $\lambda = 6$.

- T_k: Arrival time of the kth customer
- W_k: Time between the $k-1$th arrival and the kth arrival

$$W_k = T_k - T_{k-1}.$$

Arrival Times

In this simulation, on average, the number of customers arriving is 5 per hour, so around every 12 minutes. The number of customers that will be arriving within the operating hours is 240, with the first customer arriving 3 minutes after opening at 10:15 AM and the last customer arriving 21 minutes before closing at 8:54 PM.




```
         Min       Q1   Median       Q3      Max
1 0.07139252 2.798893 6.563032 14.6846 61.14708
```

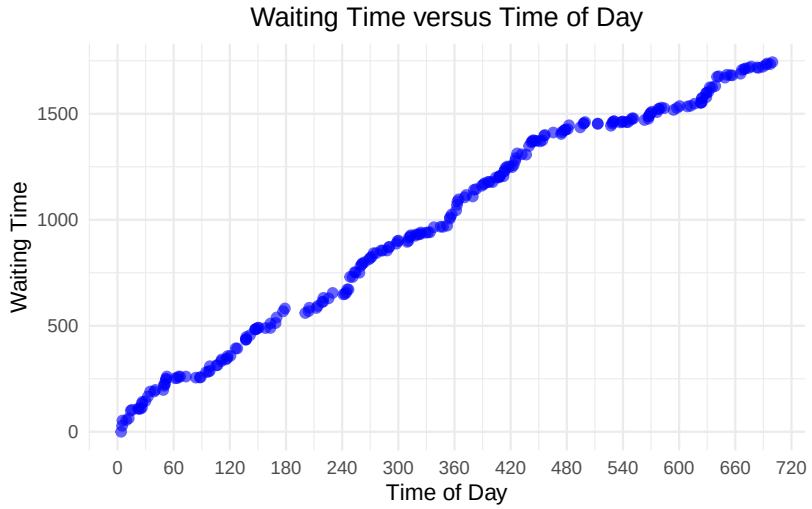
The median service time is 7 minutes, with the shortest service time being 0 and the longest service time being 61, which appears to be an outlier. Overall, the data is skewed right, consistent with an exponential distribution, and this indicates that service times tend to be quick.


```
  Min       Q1   Median       Q3      Max      SD
1   0 512.3454 1018.529 1459.293 1743.515 514.062
```

The median waiting time is 1019 minutes, with the lowest waiting time being 0 and the highest waiting time being 1744. Overall, waiting times tends to be fairly symmetric with a slight positive skew and there are no visible outliers. Wait times are spread out with times being, on average, 514 minutes from the mean of 983 minutes.

	Customer	Service_Length	Service_Start	Service_End	Waiting_Time	Time
1	1	29.3442085	3.87011	33.21432	0.00000	10:04
2	2	25.2260151	33.21432	58.44033	28.21476	10:05
3	3	5.2628989	58.44033	63.70323	53.13574	10:05
4	4	11.8316484	63.70323	75.53488	54.26691	10:09
5	5	37.8071290	75.53488	113.34201	63.52481	10:12
6	6	5.7064764	113.34201	119.04849	99.19646	10:14
7	7	11.9278687	119.04849	130.97635	103.62019	10:15
8	8	0.2188514	130.97635	131.19521	109.02745	10:22
9	9	1.8062338	131.19521	133.00144	108.54252	10:23
10	10	6.5821241	133.00144	139.58356	108.43161	10:25

	Arrival_time
1	3.870110
2	4.999560
3	5.304592
4	9.436318
5	12.010067
6	14.145550
7	15.428292
8	21.948904
9	22.652688
10	24.569826



Overall, waiting times appear to be highest around noon or lunch when customer satisfication is lowest. Waiting times decrease steadily 3 pm onwards until closing, suggesting a less busy dinner time where the restaurant has more down time. In our simulation, the maximum service time, 73 minutes, occured an hour before the rush hour, and as a result, waiting times increased significantly after this customer. Although overall service times are fairly quick, the restaurant should do their best to ensure service times are just as fast or even faster during the busy hours.

Scenario 2

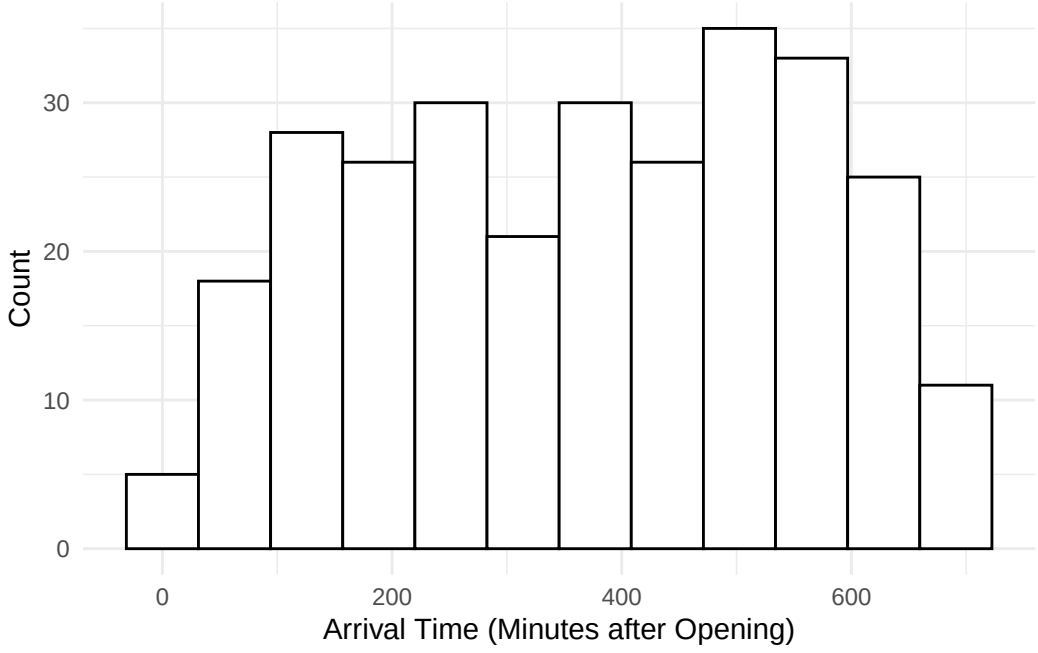
Assumptions:

1. 5 dining tables and L chefs with operating hours 10am - 10pm. We choose here that $L = 2$
2. each table only seats one customer
3. service time modeled by an exponential distribution with rate $S = 3L$, so that the more chefs there are, the faster the service times become
4. 24 customers arrive every hour

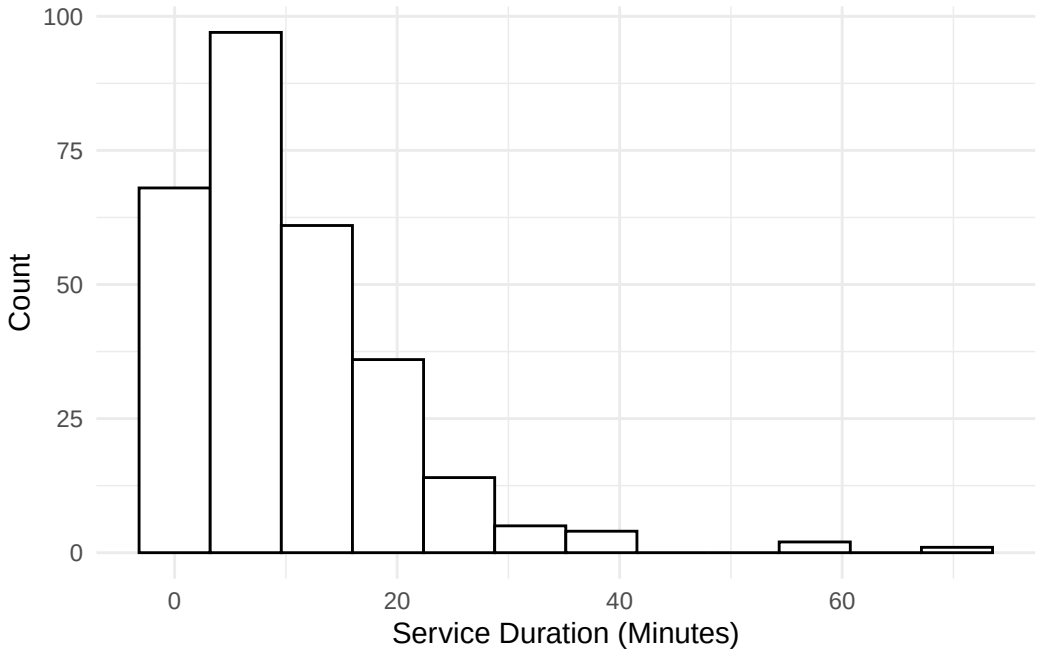
Waiting Times

To model waiting times, we iterate through the day minute by minute.

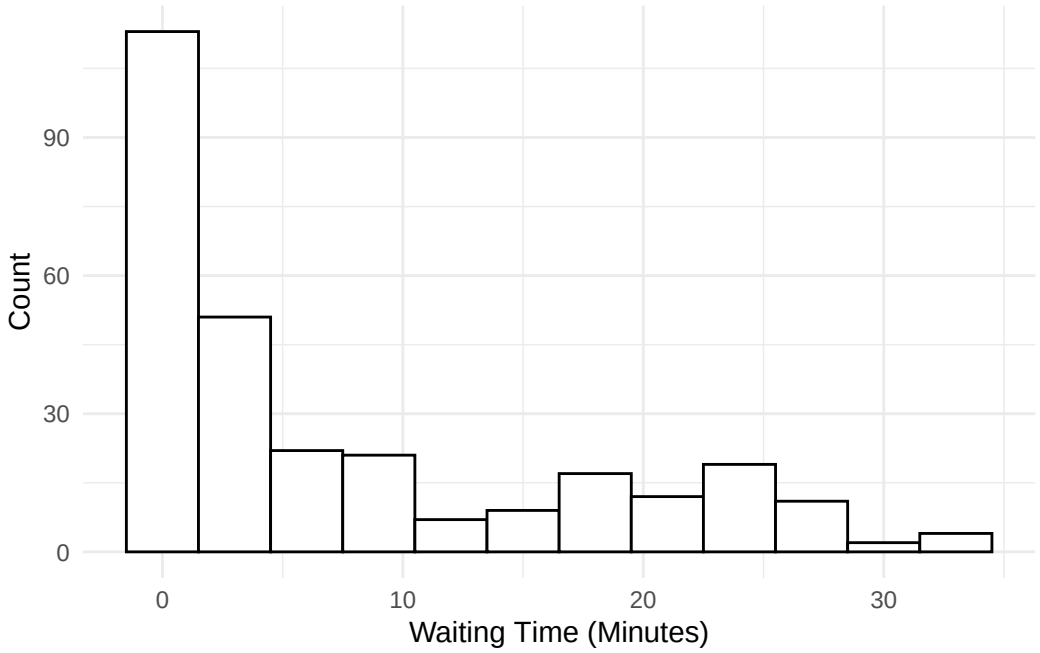
```
scen2_sim_results_by_customer |>
  ggplot(aes(x = arrival_time)) +
  geom_histogram(bins = 12, color = "black", fill = "white") +
  labs(
    x = "Arrival Time (Minutes after Opening)",
    y = "Count"
  ) +
  theme_minimal()
```



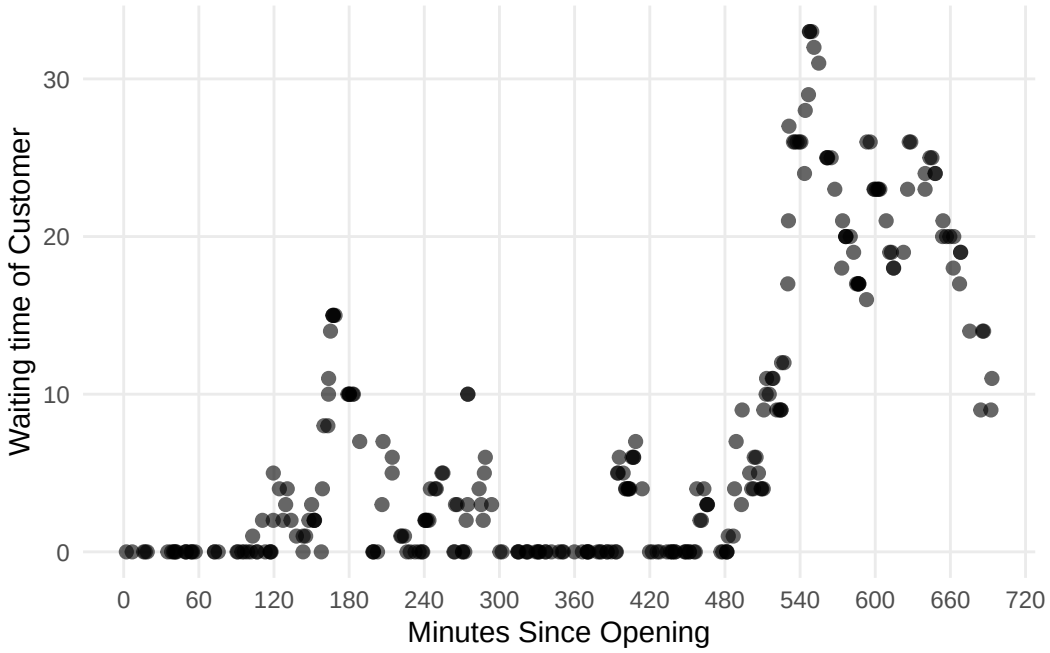
```r
scen2_sim_results_by_customer |>
  ggplot(aes(x = service_length)) +
  geom_histogram(bins = 12, color = "black", fill = "white") +
  labs(
    x = "Service Duration (Minutes)",
    y = "Count"
  ) +
  theme_minimal()
```



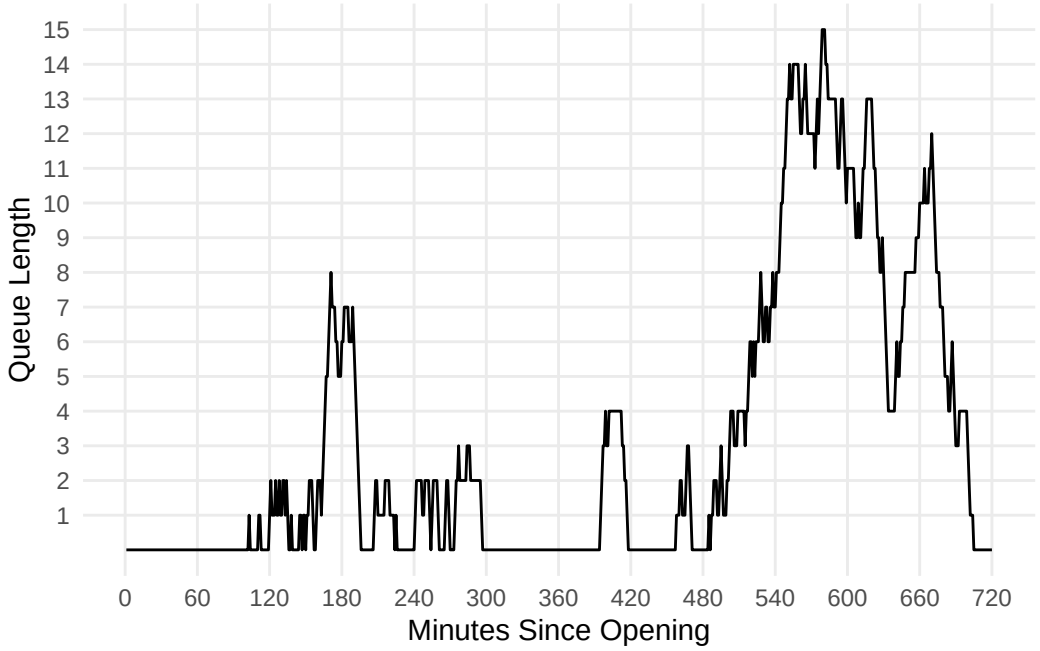

```r
scen2_sim_results_by_customer |>
  ggplot(aes(x = waiting_time)) +
  geom_histogram(bins = 12, color = "black", fill = "white") +
  labs(
    x = "Waiting Time (Minutes)",
    y = "Count"
  ) +
  theme_minimal()
```



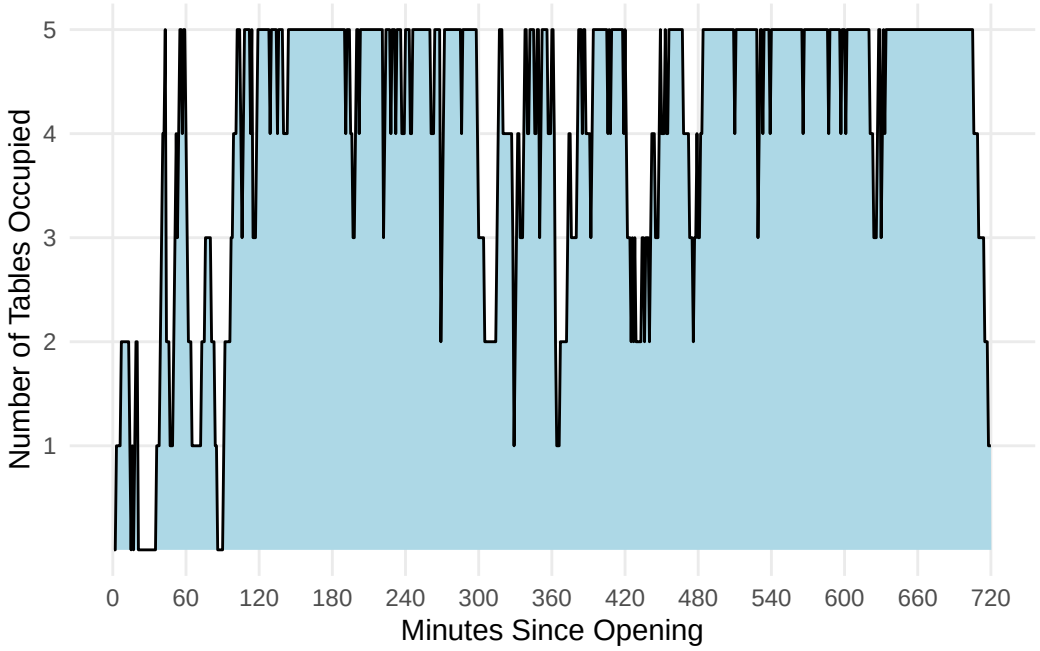

```
scen2_sim_results_by_customer |>
  ggplot(aes(x = arrival_time, y = waiting_time)) +
  geom_point(size = 2, alpha = 0.6) +
  scale_x_continuous(breaks = seq(0, as.numeric(total_time_s2), by = 60)) +
  labs(
    x = "Minutes Since Opening",
    y = "Waiting time of Customer"
  ) +
  theme_minimal() +
  theme(panel.grid.minor = element_blank())
```




```
scen2_sim_results_by_minute |>
  ggplot(aes(x = minutes_since_opening, y = queue_size)) +
  geom_line() +
  scale_y_continuous(breaks = seq(1, max(sim_s2$queue_size_history), by = 1)) +
  scale_x_continuous(breaks = seq(0, as.numeric(total_time_s2), by = 60)) +
  labs(
    x = "Minutes Since Opening",
    y = "Queue Length"
  ) +
  theme_minimal() +
  theme(panel.grid.minor = element_blank())
```




```
# OCCUPIED TABLES
scen2_sim_results_by_minute |>
  ggplot(aes(x = minutes_since_opening, y = occupied_tables)) +
  geom_area(fill = "lightblue") +
  geom_line() +
  scale_y_continuous(breaks = seq(1, num_tables_s2, by = 1)) +
  scale_x_continuous(breaks = seq(0, as.numeric(total_time_s2), by = 60)) +
  labs(
    x = "Minutes Since Opening",
    y = "Number of Tables Occupied"
  ) +
  theme_minimal() +
  theme(panel.grid.minor = element_blank())
```



Restaurant Profits

Assumptions:

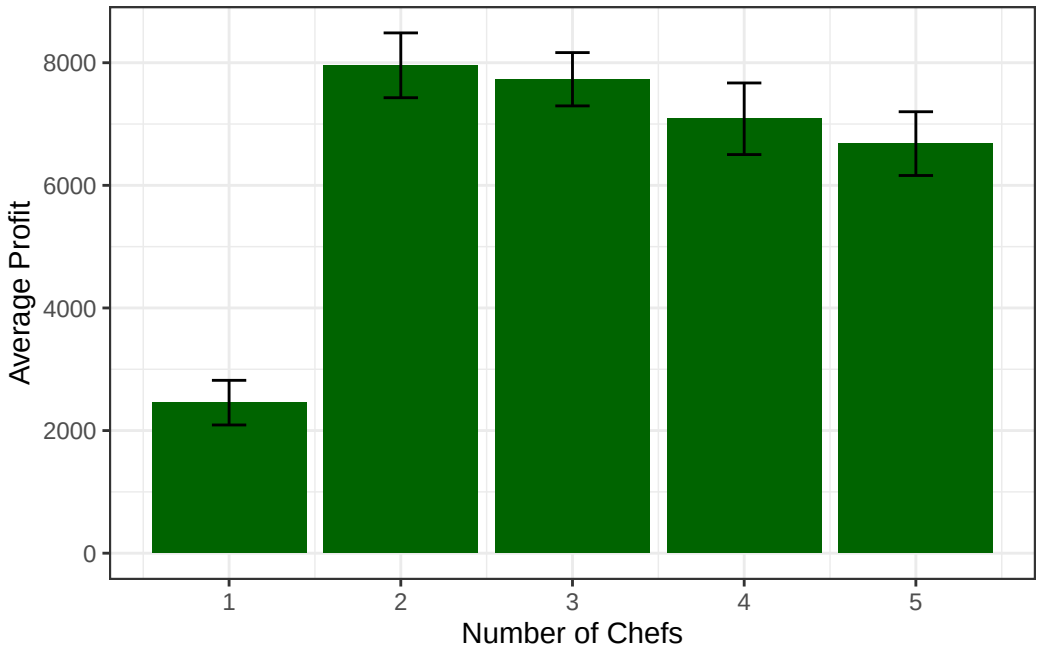
1. each customer spends $50 per meal (customers who are still in the queue when the restaurant closes won't pay)
2. each chef earns a wage of $40 per hour (paid for the entire duration of the restaurant's operating hours)
3. Each table cost $1000 per day (extra service cost, rent, etc.)
4. For customers who waited more than 30 minutes, they earn the restaurant half the amount of customers who didn't.

Maximizing Profits

With 5 tables, 24 customers arriving per hour, and these dollar amounts, how many chefs should we hire? We will run our simulation 100 times with 1 to 5 chefs on staff, to see which will maximize the expected profit.

	total_customers	profit	num_chefs	num_tables	avg_waiting_time	long_waits
1	279	2245	1	5	1.401290e+02	249
2	288	2095	1	5	1.826007e+02	273
3	288	7960	3	5	4.131944e-01	0
4	288	7960	3	5	1.902778e+00	0
5	288	7480	4	5	5.902778e-02	0
6	288	2945	1	5	1.275799e+02	239
7	255	5350	5	5	3.921569e-03	0

```
8             253   6210         3          5     3.992095e-01          0
  avg_queue_length max_queue_length avg_tables_occupied
1      54.300000000              128            4.937500
2      73.040277778              132            4.913889
3       0.165277778                7            2.677778
4       0.761111111               15            2.906944
5       0.023611111                2            2.256944
6      51.031944444              102            4.904167
7       0.001388889                1            1.365278
8       0.140277778                6            2.411111
```

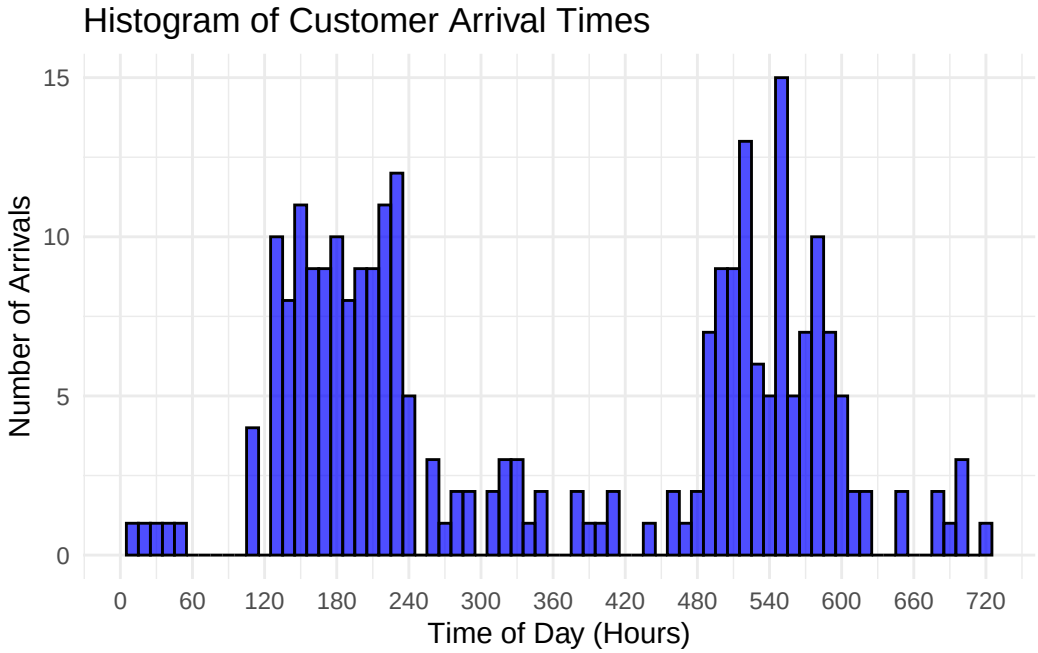


Scenario 3

To make the simulation more realistic, we have a third scenario.

Assumptions: 1. Open at 10am, close at 10pm 2. From 12pm to 2pm and 6pm to 8pm, 60 customers arrive every hour. Otherwise, 6 arrive every hour. 3. Instead of simulating service times with Exp( ) where = 3 times the number of chefs, we do = ln(chefs + 1), so that additional chefs beyond 2 make more of an impact. 4. Each customer will sit for a minimum of 45 minutes. This flat value will be added to the simulated service time, and is unaffected by staffing. 5. In the profit calculation, there is a cost of adding additional tables (which are now variable), which is $40 per table. 6. Chefs still cost $40 per hour to hire, and each customer earns $50.

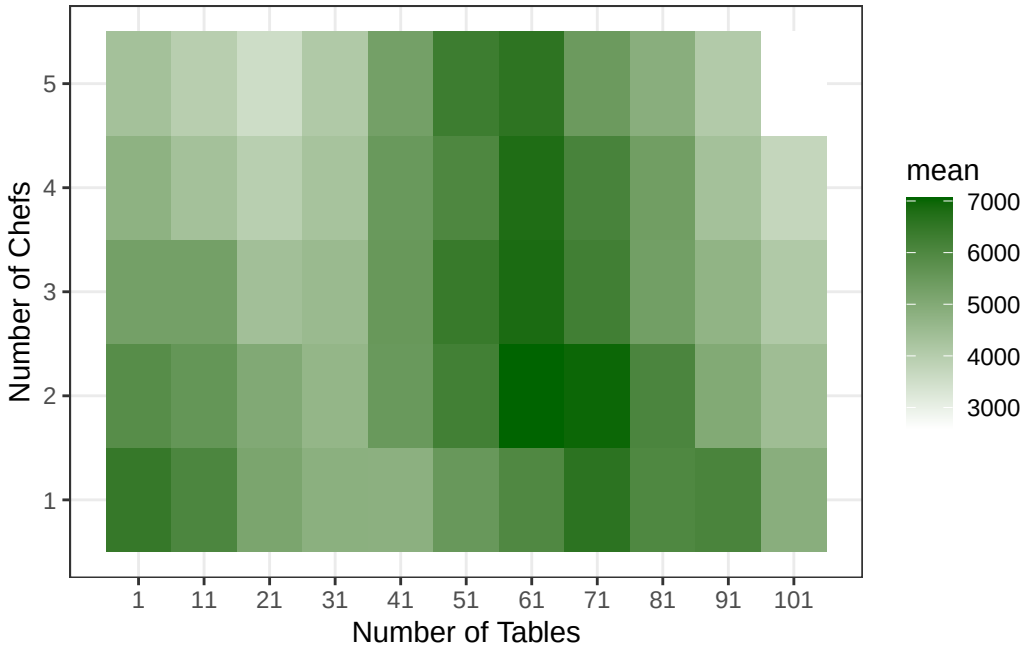
Arrival Times



Maximizing Profits

Under this scenario, how can we maximize profits?

```
`summarise()` has grouped output by 'num_chefs'. You can override using the
`.groups` argument.
```



```
`summarise()` has grouped output by 'num_chefs'. You can override using the
`.groups` argument.

# A tibble: 10 x 4
# Groups:   num_chefs [5]
   num_chefs num_tables mean_profit    sd
       <dbl>      <dbl>       <dbl> <dbl>
 1         2         61       7058.  670.
 2         2         71       6978. 1195.
 3         3         61       6839.  933.
 4         4         61       6784.  895.
 5         1         71       6598.  799.
 6         5         61       6576. 1093.
 7         1          1       6451.  370.
 8         3         51       6410   663.
 9         5         51       6328.  615.
10         3         71       6258.  956.

`summarise()` has grouped output by 'num_chefs'. You can override using the
`.groups` argument.

# A tibble: 3 x 5
# Groups:   num_chefs [2]
  num_chefs num_tables profit mean_waiting_time    sd
      <dbl>      <dbl>  <dbl>             <dbl> <dbl>
1         2         61  7058.              9.31  3.59
2         2         71  6978.              3.90  2.48
3         3         61  6839.              6.36  2.22

`summarise()` has grouped output by 'num_chefs'. You can override using the
`.groups` argument.

# A tibble: 3 x 5
# Groups:   num_chefs [2]
  num_chefs num_tables profit mean_long_waits    sd
      <dbl>      <dbl>  <dbl>           <dbl> <dbl>
1         2         61  7058.            30.9 26.8
2         2         71  6978.             5.5  9.93
3         3         61  6839.            12.4 13.9

`summarise()` has grouped output by 'num_chefs'. You can override using the
`.groups` argument.

# A tibble: 3 x 5
# Groups:   num_chefs [2]
  num_chefs num_tables profit mean_queue_length    sd
```

```
      <dbl>      <dbl>  <dbl>             <dbl> <dbl>
1         2         61  7058.              3.61  1.66
2         2         71  6978.              1.56  1.11
3         3         61  6839.              2.43  1.01
```

```
`summarise()` has grouped output by 'num_chefs'. You can override using the
`.groups` argument.
```

```
# A tibble: 3 x 5
# Groups:   num_chefs [2]
  num_chefs num_tables profit mean_max_queue    sd
      <dbl>      <dbl>  <dbl>          <dbl> <dbl>
1         2         61  7058.           27.2  6.34
2         2         71  6978.           17.0  5.80
3         3         61  6839.           22.7  5.03
```

```
`summarise()` has grouped output by 'num_chefs'. You can override using the
`.groups` argument.
```

```
# A tibble: 3 x 5
# Groups:   num_chefs [2]
  num_chefs num_tables profit mean_occupied    sd
      <dbl>      <dbl>  <dbl>         <dbl> <dbl>
1         2         61  7058.          34.9  2.81
2         2         71  6978.          36.4  3.52
3         3         61  6839.          31.3  2.38
```