

Sweetgreen Case Study

Production Scheduling - Spring 2016

Karen Gao (kcg2113)

Ivy Zheng (iyz2101)

Ivy Pan (ip2257)

Derek Yan (dmy2110)

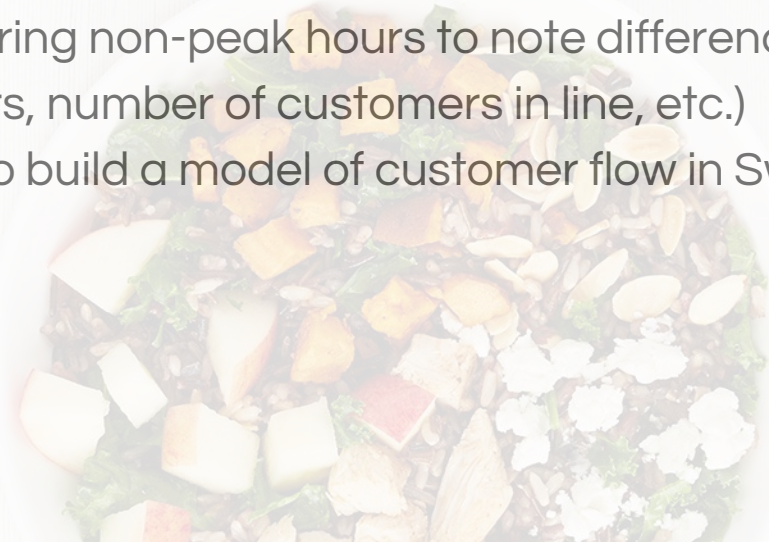
Background

- Sweetgreen is a new but popular salad fast food option near Columbia
- After noticing seemingly long wait times, we decided to do a case study on the salad production process at Sweetgreen



Data Gathering

- Focused on collecting data during dinner rush hour (starting at 6:00 PM)
- Observed queuing time from entering the store until the actual salad ordering process, the length of the salad-making process, the length of the checkout process, interarrival times of customers, average queue length
- Also observed during non-peak hours to note differences and patterns (number of servers, number of customers in line, etc.)
- Aim to use data to build a model of customer flow in Sweetgreen



Objective

- From our observations, we noted that when the line grew to around 15, potential customers would be discouraged by the long queue and leave.
- This positive abandon rate meant Sweetgreen was losing out on potential customers, thus potential profits.
- At times, we observed wait times up to **20** min, generally unacceptable for a restaurant marketed as fast food
- We want to adjust factors in our Sweetgreen model to try to increase efficiency and decrease the wait times of the customers



Current Model

- Arrays:

- `arrival[]` = Uses exp-distributed interarrival times using a rate calculated from observed data, which are first converted to seconds.

```
interarrival = exprnd(meanA, 90, 1);
```

```
interarrival(interarrival < 1) = 0;
```

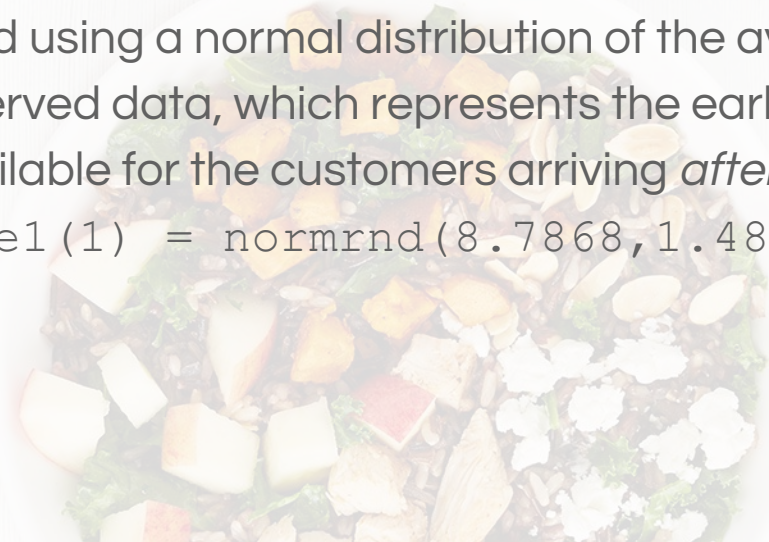
- We accounted for burst (multiple customers coming in at once) by setting the interarrival time = 0 for differences of less than 1 second.
- The arrival times are first generated in seconds, then cut off at the hour mark and converted back into minutes.

```
arrivalTimes = arrivalTimes(arrivalTimes < 3600);
```

```
arrival = arrivalTimes/60;
```

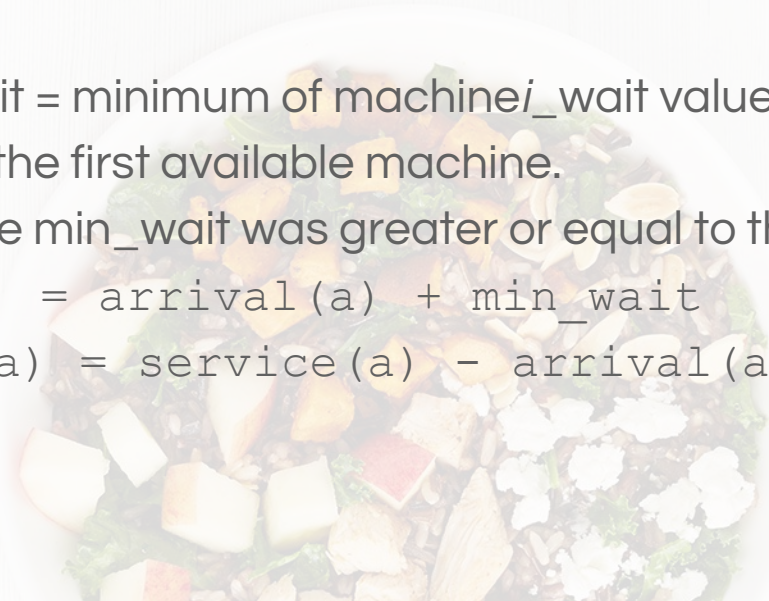

Current Model

- Arrays, continued:
 - `service[]` = time at which each customer begins service.
 - `machine1[], machine2[], ...` = time at which machine i completed service for a customer that was assigned to it.
 - Initialized using a normal distribution of the average wait time from our observed data, which represents the earliest time each machine was available for the customers arriving *after* $t = 0$.
`machine1(1) = normrnd(8.7868, 1.4806); etc.`



Current Model

- Calculations:
 - To find which machine would be the first available to each customer a
 $machine1_wait = machine1(end) - arrival(a)$
etc.
 - Find $min_wait = \text{minimum of } machine_i_wait \text{ values}$, and assign the customer to the first available machine.
 - As long as the min_wait was greater or equal to than 0,
 $service(a) = arrival(a) + min_wait$
 - Thus, $wait(a) = service(a) - arrival(a)$



Current Model

- Calculations, continued:
 - The processing time for each machine is randomly generated using a normal distribution based off our observed data.
 - For whichever machine customer a was assigned to, we would add the order's completion time to the end of the machine's array.

```
processing = normrnd(meanM, sqrt(varM));  
  
if min_wait == machine1_wait  
    machine1(end+1) = service(a) + processing;  
elseif min_wait == machine2_wait  
    machine2(end+1) = service(a) + processing;  
etc
```


Current Model

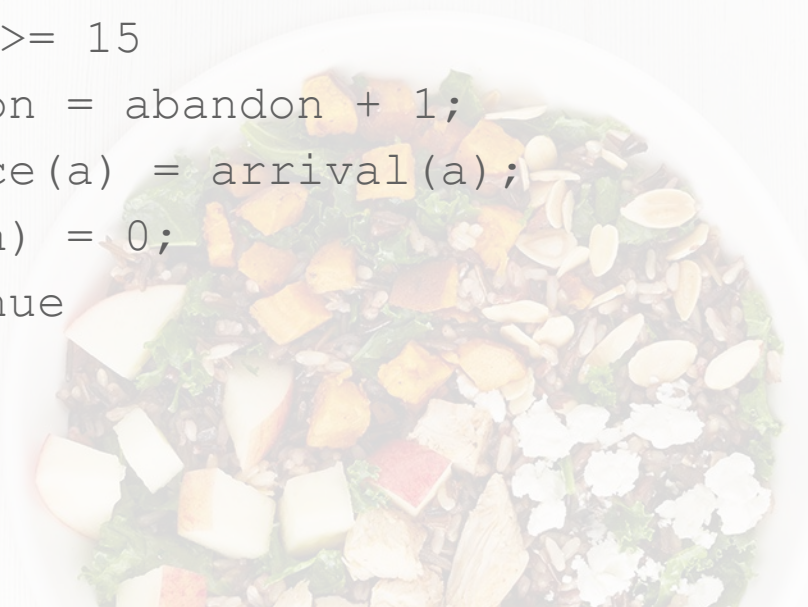
- Abandon Rate
 - To account for customers who abandon the line (let's say when the queue is greater than or equal to 15), we first must calculate how many are in the queue when a customer first arrives:
$$\text{queue} = 12 + a - \text{numel}(\text{service}(\text{service} < \text{arrival}(a))) - \text{abandon}$$
 - The number of customers we begin with (e.g. 12) is arbitrary and does not affect the overall average wait time of 1000 iterations.



Current Model

- Abandon Rate, continued:
 - Through observation, we found that 15 was the queue length that customers began to abandon.

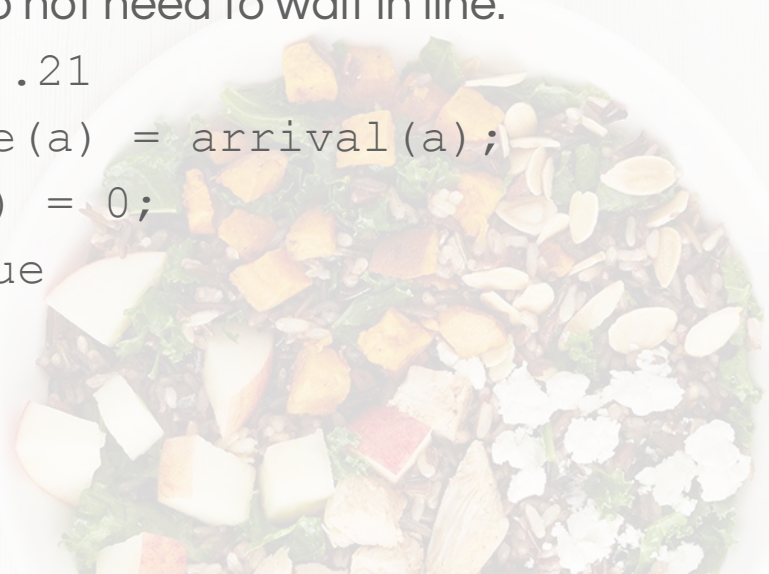
```
if queue >= 15
  abandon = abandon + 1;
  service(a) = arrival(a);
  wait(a) = 0;
  continue
end
```

A bowl of fresh salad with various fruits, nuts, and greens. The salad includes sliced apples, almonds, and other ingredients. The bowl is white and sits on a light-colored surface.

Current Model

- Pre-Order Rate
 - Sweetgreen's public data suggests that 21% of its orders are made online and available for pick-up at the front of the counter, thus these customers do not need to wait in line.

```
if rand < .21
    service(a) = arrival(a);
    wait(a) = 0;
    continue
end
```

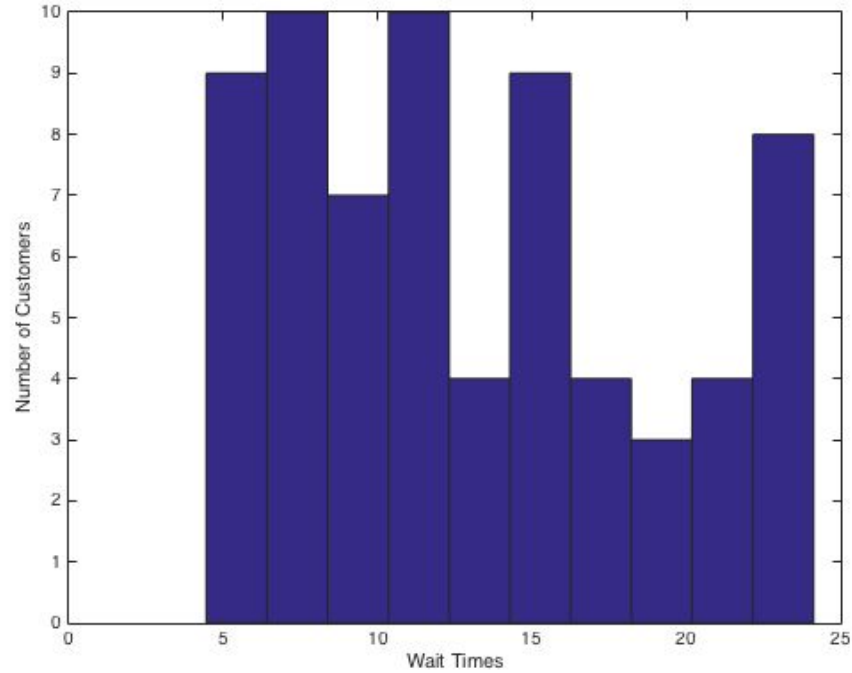
A bowl of salad with various fruits, nuts, and greens. The salad includes sliced apples, almonds, and other ingredients. The bowl is white and the salad is colorful.

Issues with Current Setup

- We tested our model by running through 1000 iterations.
 - We found that the number of lost customers was approximately **19.33 per hour** for an abandon rate of **18.0%** for the 6pm dinner rush.
 - The average wait time was approximately **13.7 minutes**.
- This is an even greater issue for the lunch rush (12pm - 2pm) which, according to Google's listings of busiest times available for restaurants, is approximately 15 - 25% busier than 6pm.



Current Model

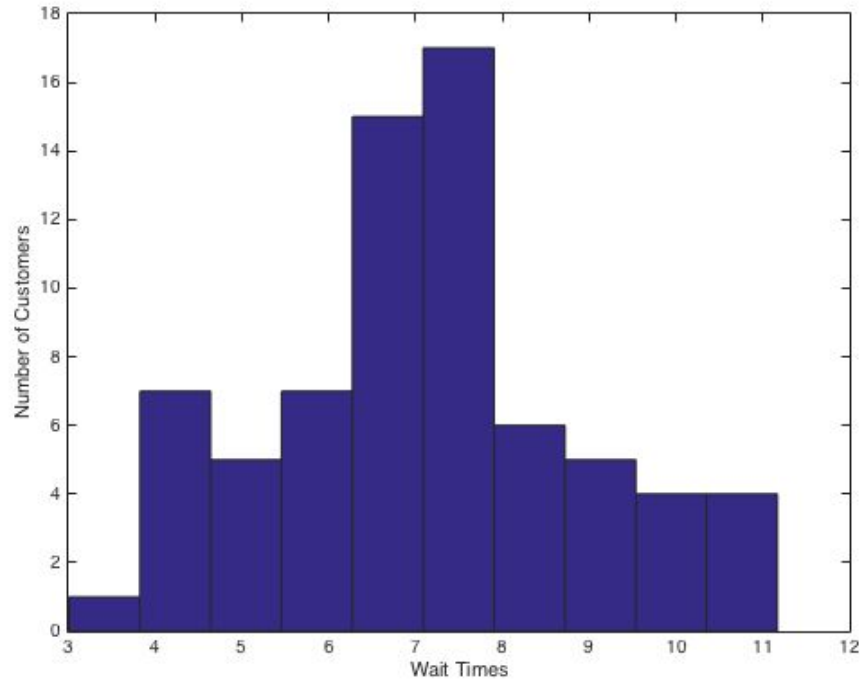


Modified Model - 5 Machines (P5)

- We added another machine so that 5 machines were running in parallel. The initialization and processing rate assumptions are the same.
- We found that the number of lost customers was approximately **13.71 per hour** for an abandon rate of **12.7%** for the 6pm dinner rush. The average wait time was **8.2 minutes**.



Modified Model - 5 Machines (P5)



Modified Model - x2 Pre-Order Rate

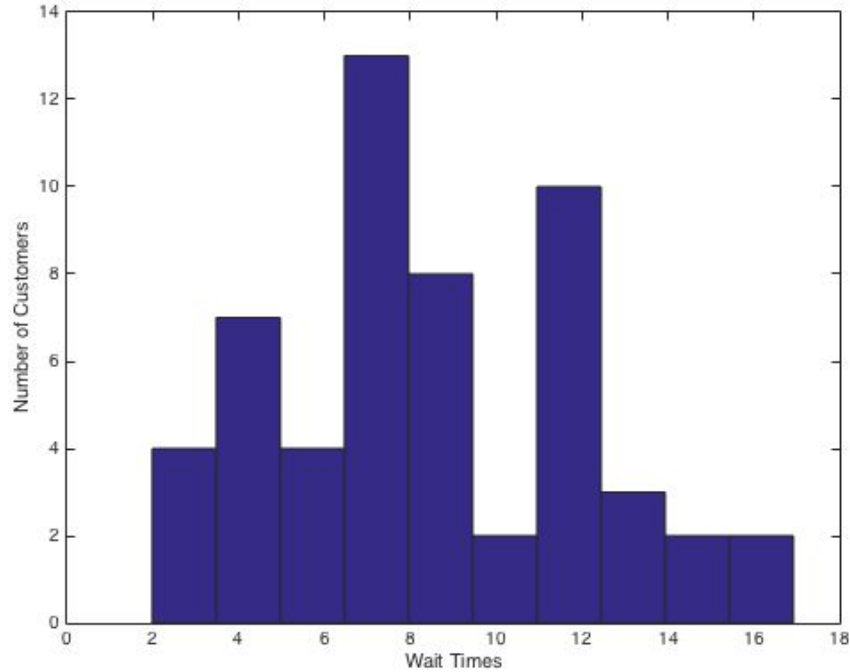
- If Sweetgreen increased its marketing to encourage more people to pre-order, it would reduce the average wait time and queue length, thus decreasing the abandon rate.

```
if rand < .42
    service(a) = arrival(a);
    wait(a) = 0;
    continue
end
```

- The number of lost customers is reduced to **8.87 per hour** for an abandon rate of **8.2%** and average wait time of **10.5 minutes**.



Modified Model - x2 Pre-Order Rate

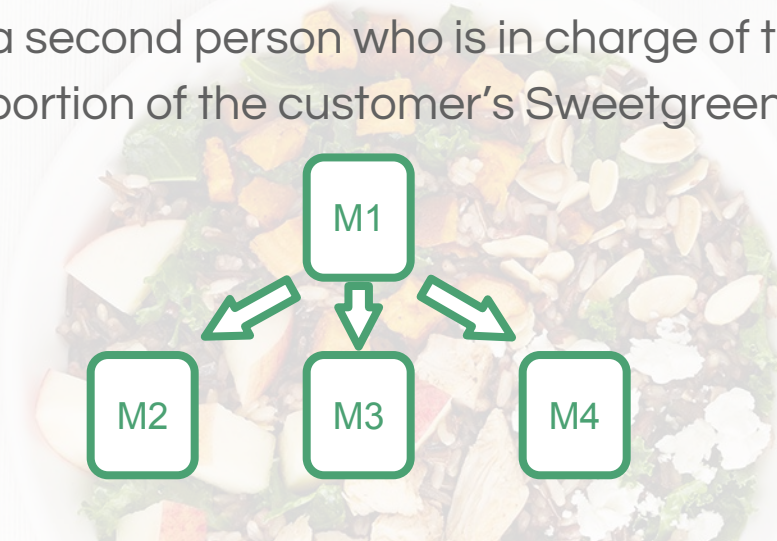


Modified Model - x2 Pre-Order Rate

- The issue of this model is that it does not account for the decrease in efficiency for workers, who may have to stop more often to create the pre-ordered salads.
 - Sweetgreen can alleviate this by hiring an additional employee who makes only pre-ordered salads, but this would incur extra salary costs.
 - Consider the case that Sweetgreen doesn't hire an additional worker, but instead allows each machine's processing time to increase by 20%.
 - The number of lost customers is increased to **12.11 per hour** for an abandon rate of **11.2%** and average wait time of **10.5 minutes**.
 - This is still much better than the original system (abandon rate of 18%).

Modified Model - Flow Shop (Partial Assembly Line)

- Sweetgreen's business model emphasizes customer interaction thus the complete assembly line model, e.g. Chipotle, would compromise their intent.
- However, Sweetgreen already occasionally uses a ***partial assembly line system***, where one person is in charge of base ingredients only before handing it off to a second person who is in charge of the toppings and acts as the one-on-one portion of the customer's Sweetgreen experience.



Modified Model - Flow Shop (Partial Assembly Line)

- Assume that the base ingredients portion has limited customer interaction thus only requires 20% of the total processing time.

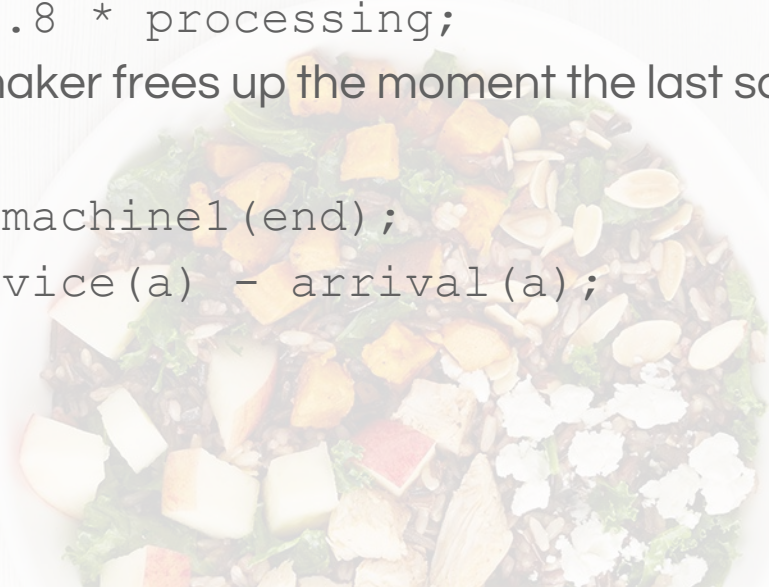
```
baseprocess = .2 * processing;
```

```
topprocess = .8 * processing;
```

- The salad base maker frees up the moment the last salad base was complete.

```
service(a) = machine1(end);
```

```
wait(a) = service(a) - arrival(a);
```



Modified Model - Flow Shop (Partial Assembly Line)

- After the salad base making is complete—

```
machine1(end+1) = service(a) + baseprocess;
```

- Pass onto the first available machine for toppings.

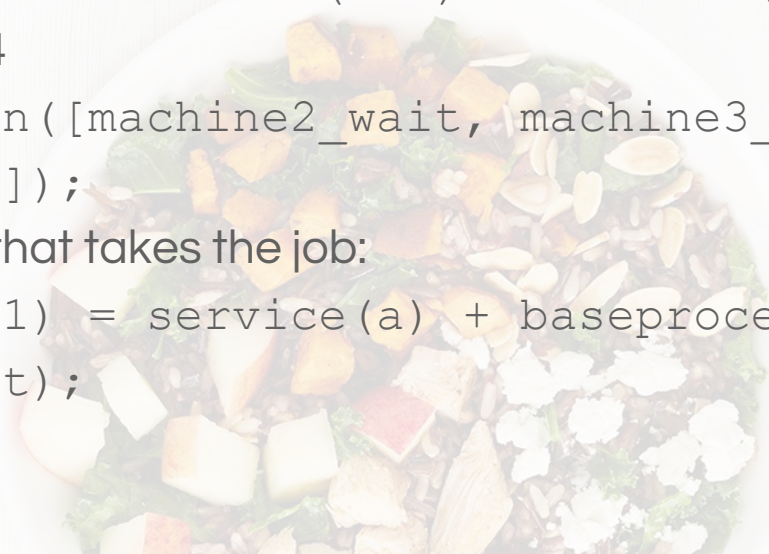
```
machine2_wait = machine2(end) - service(a) - baseprocess;
```

for machines 2 - 4

```
min_wait = min([machine2_wait, machine3_wait,  
machine4_wait]);
```

- For the machine that takes the job:

```
machine2(end+1) = service(a) + baseprocess + topprocess +  
max(0, min_wait);
```

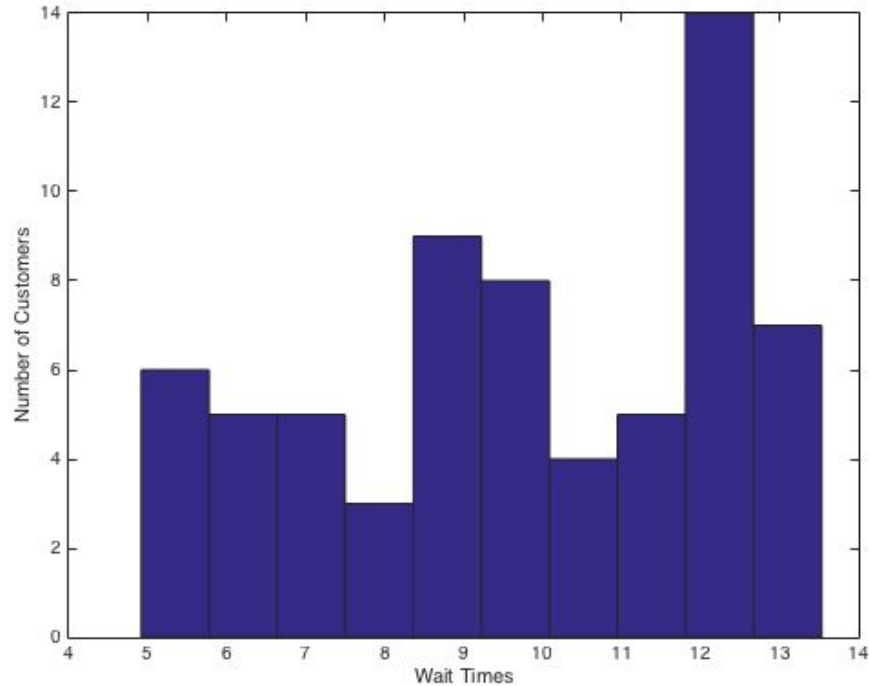


Modified Model - Flow Shop (Partial Assembly Line)

- The number of lost customers is reduced to **15.14 per hour** for an abandon rate of **14.1%** and average wait time of **9.5 minutes**.
- Unlike the other solutions, this model doesn't require any additional cost of hiring another employee for the dinner rush or marketing in order to encourage more customers to use the pre-ordering system.



Modified Model - Flow Shop (Partial Assembly Line)



Comparison

	Original Model (P4)	Additional Worker (P5)	Double Pre-order (P4)	Flow Shop (20/80)
Abandon per Hour	19.33	13.71	12.11	15.14
Abandon Rate	18.0%	12.9%	11.2%	14.1%
Avg Wait Time (minutes)	13.7	8.2	10.5	9.5

Conclusions

- As a convenience food option, Sweetgreen will find it hard to remain a permanent part of the Columbia community with its current setup, which has inefficiencies that result in long queues and wait times
- We suggest increasing pre-orders as the best alternative to increasing Sweetgreen's efficiency
 - Technology use very fitting with the environment (college campus)
 - Advertising cost relatively small and no need to retrain workers
 - Pre-orders also increase convenience to the customer, making Sweetgreen an even more attractive choice

