Review of HW 2

Clean data first!

1. From 7AM-11:05PM, draw an inventory build-up diagram. For example, draw how many customers are present in the store at every minute. Calculate the average inventory level, i.e., on average how many customers were in the store.
2. Plot how many customers entered the store in every 15 minutes from 7AM-11:05PM. Then, calculate how many customers entered the store per hour on average.
3. Take the average of trip_duration_mins, considering the group size. Use your answers from 1 and 2, and apply the Little's Law to calculate the average flow time. Confirm that the average flow time that you calculate using the Little's Law is consistent with the average trip_duration_mins.

```r
rogers <- read.csv("../data/Rogers_022824.csv")

# clean data
rogers <- rogers |> distinct(purchase_datetime, transaction_id, session_id, trip_duration_mins, group_size
summary(rogers)
```

```
##  purchase_datetime  transaction_id      session_id        trip_duration_mins
##  Length:4646        Length:4646        Length:4646        Min.   : 0.1833
##  Class :character   Class :character   Class :character   1st Qu.: 1.1833
##  Mode  :character   Mode  :character   Mode  :character   Median : 2.0166
##                                                           Mean   : 2.7562
##                                                           3rd Qu.: 3.5500
##                                                           Max.   :18.1166
##                                                           NA's   :1
##    group_size
##  Min.   :1.000
##  1st Qu.:1.000
##  Median :1.000
##  Mean   :1.112
##  3rd Qu.:1.000
##  Max.   :4.000
##  NA's   :1
```


```r
rogers <- rogers |> filter(!is.na(trip_duration_mins))

dim(rogers)
```

```
## [1] 4645    5
```

Unif of obs / Session ID
Datatime

```
rogers <- rogers |>
  mutate(purchase_datetime = as.POSIXct(purchase_datetime, format = "%m/%d/%Y %H:%M"))

# collapsing into id level
rogers <- rogers |> distinct(purchase_datetime, trip_duration_mins, group_size) |> as.data.table()
dim(rogers)
```

exit time

Session ID X

```
## [1] 2169    3
```

```
# add variables
rogers[, entry_datetime := purchase_datetime - as.difftime(trip_duration_mins, units = "mins")]
summary(rogers)
```

```
##  purchase_datetime                trip_duration_mins   group_size
##  Min.   :2024-02-28 07:03:00.0   Min.   : 0.1833    Min.   :1.000
##  1st Qu.:2024-02-28 12:39:00.0   1st Qu.: 0.9166    1st Qu.:1.000
##  Median :2024-02-28 16:07:00.0   Median : 1.4833    Median :1.000
##  Mean   :2024-02-28 16:03:33.5   Mean   : 2.0459    Mean   :1.088
##  3rd Qu.:2024-02-28 19:47:00.0   3rd Qu.: 2.5500    3rd Qu.:1.000
##  Max.   :2024-02-28 23:01:00.0   Max.   :18.1166    Max.   :4.000
##  entry_datetime
##  Min.   :2024-02-28 07:01:56.00
##  1st Qu.:2024-02-28 12:37:48.00
##  Median :2024-02-28 16:05:01.00
##  Mean   :2024-02-28 16:01:30.75
##  3rd Qu.:2024-02-28 19:45:56.00
##  Max.   :2024-02-28 23:00:39.00
```

Inventory build up diagram → where approximation error

min

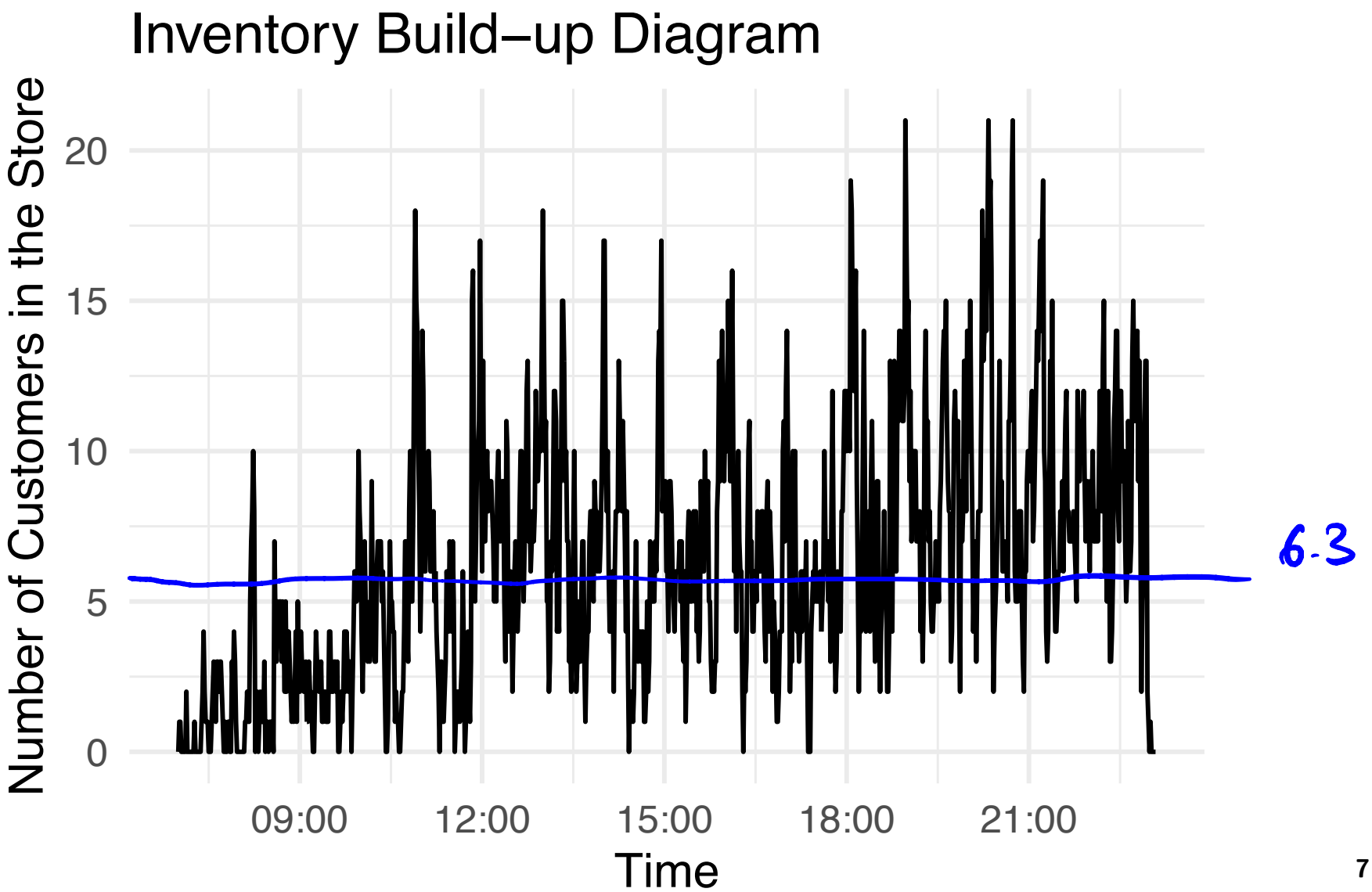
```r
# Create a sequence of minutes from 7AM to 11:05PM
time_seq <- seq(as.POSIXct("2024-02-28 07:00:00"), as.POSIXct("2024-02-28 23:05:00"), by = "1 min")

# Create a data.table with the time sequence
dt_times <- data.table(time = time_seq)

# Create a data.table to store the customer count for each minute
dt_customer_count <- data.table(time = time_seq, customer_count = 0)

# Iterate over each row in the original data.table
for (i in 1:nrow(rogers)) {
  # Get the entry and purchase datetimes for the current row
  entry_time <- rogers[i, entry_datetime]
  exit_time <- rogers[i, purchase_datetime]

  # Round down the entry time to the nearest minute
  entry_time_rounded <- floor_date(entry_time, "minute")

  # Find the corresponding time intervals in dt_times
  # time_indices <- which(dt_times$time >= entry_time_rounded & dt_times$time <= exit_time)
  # time_indices <- which(dt_times$time > entry_time_rounded & dt_times$time < exit_time)
  time_indices <- which(dt_times$time >= entry_time_rounded & dt_times$time < exit_time)

  # Increment the customer count for the corresponding time intervals
  dt_customer_count[time_indices, customer_count := customer_count + rogers[i, group_size]]
}

avg_inventory <- mean(dt_customer_count$customer_count)
print(avg_inventory)
```

```
## [1] 6.312629
```

~8

~10

```r
dt_customer_count |>
  ggplot(aes(x = time, y = customer_count)) +
  geom_line() +
  labs(title = "Inventory Build-up Diagram",
       x = "Time",
       y = "Number of Customers in the Store") +
  theme_minimal()
```

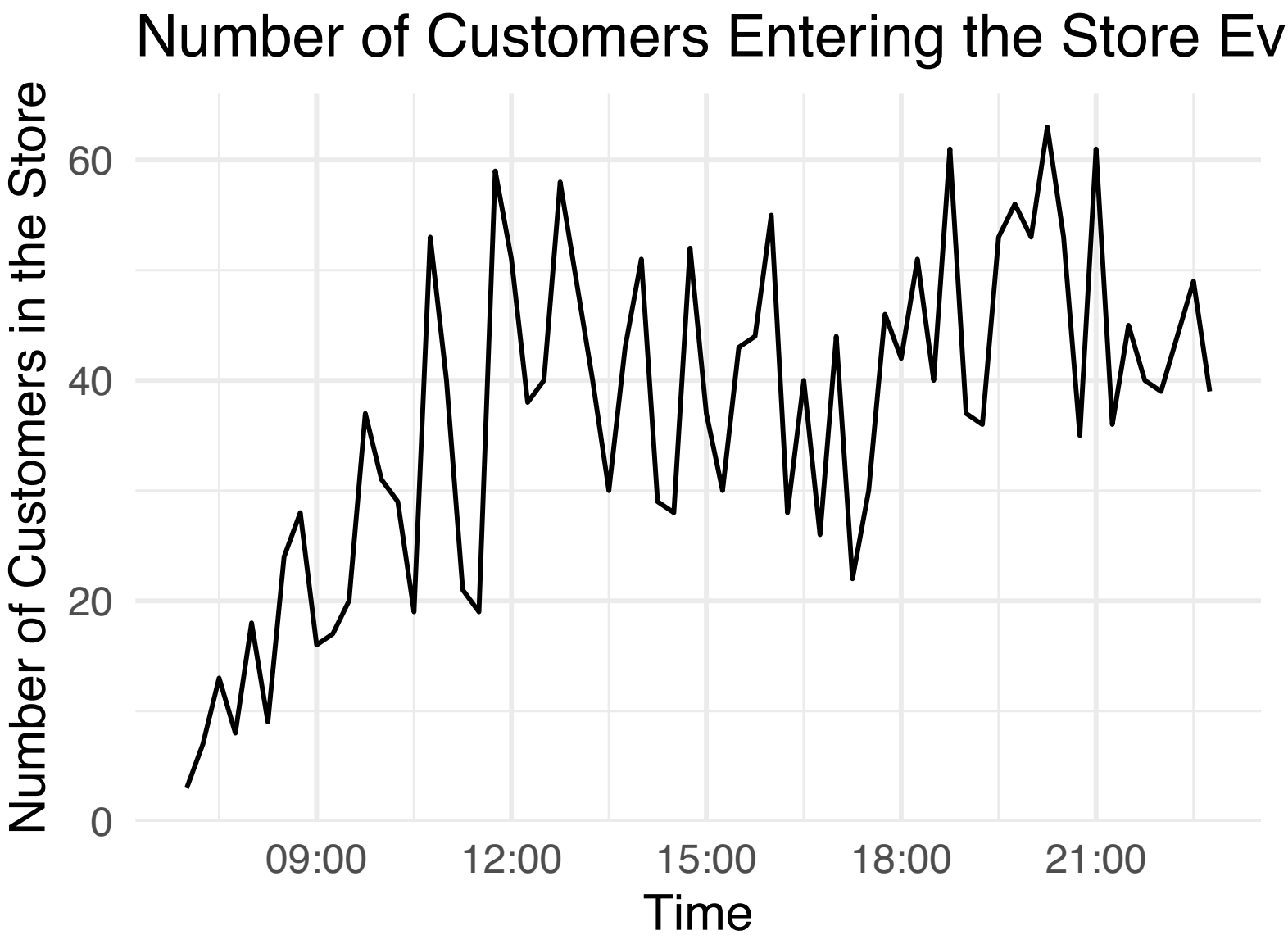


```r
# Create a sequence of 15-minute intervals from 7AM to 11:05PM
time_seq <- seq(as.POSIXct("2024-02-28 07:00:00"), as.POSIXct("2024-02-28 23:05:00"), by = "15 min")

# Create a data.table with the time sequence
dt_times <- data.table(start_time = time_seq, end_time = time_seq + minutes(15))

# Create a data.table to store the customer count for each 15-minute interval
dt_customer_count_15min <- data.table(start_time = time_seq, customer_count = 0)

# Iterate over each row in the original data.table
for (i in 1:nrow(rogers)) {
  # Get the entry and purchase datetimes for the current row
  entry_time <- rogers[i, entry_datetime]

  # Find the corresponding time intervals in dt_times
  time_indices <- which(dt_times$start_time <= entry_time & dt_times$end_time > entry_time)

  # Increment the customer count for the corresponding time intervals
  dt_customer_count_15min[time_indices, customer_count := customer_count + rogers[i, group_size]]
}

dt_customer_count_15min <- dt_customer_count_15min |> filter(start_time != "2024-02-28 23:00:00")
print(mean(dt_customer_count_15min$customer_count))
```

```
## [1] 36.84375
```

Aggregated 15 min

```r
dt_customer_count_15min |>
  ggplot(aes(x = start_time, y = customer_count)) +
  geom_line() +
  labs(title = "Number of Customers Entering the Store Every 15 Minutes",
       x = "Time",
       y = "Number of Customers in the Store") +
  theme_minimal()
```



```r
# how many customers entered the store per hour on average.
# Extract the hour from entry_datetime
rogers[, entry_hour := hour(entry_datetime)]
# Calculate the total number of customers entering per hour
hourly_customer_count <- rogers[, .(total_customers = sum(group_size)), by = entry_hour]

hourly_customer_count <- hourly_customer_count |> filter(entry_hour != 23)
# Calculate the average number of customers entering per hour
avg_customers_per_hour <- mean(hourly_customer_count$total_customers)
print(avg_customers_per_hour)
```

```
## [1] 147.375
```

```r
rogers |>
  summarise(duration_min = weighted.mean(trip_duration_mins, group_size))
```

```
##   duration_min
## 1      2.10918
```

Data

```r
# throughput (customers/min)
R <- avg_customers_per_hour/60

# inventory (customers)
I <- avg_inventory

I / R
```

```
## [1] 2.570027
```

Little's Law