Eco-Counter Exercise

Alice Clauss

2025-09-06

Exercise 1

Objectif 1

> Pensez-vous que les données de chaque année reflètent fidèlement la fréquentation cycliste de la ville ? Si ce n'est pas le cas, précisez pour quelles périodes elles vous semblent incorrectes et proposez des explications possibles.

Pour le plus part, je pense que les données de 2023 reflètent fidèlement la fréquentation cycliste pour ces stations à Paris, à l'exception de une petite pause dans les données pour la station *27 Quai de la Tournelle* pendant mi-Août. Je postule que la pause des données pour *Quai de la Tournelle* était une panne équipement ou une batterie à plat qui a été résolu avant le fin du mois ou que la voie cycliste était barrée pour des travaux temporaires.

Mais, pour 2024, les données deviennent bizarre. Pendant 2023, les stations *Quai d'Orsay* et *27 Quai de la Tournelle*, et *36 Quai de Grenelle* décrivent une tendance similaire, mais en 2024 le compte de *Quai d'Orsay* a rapidement augmenté d'environs 50% de plus que l'année prochaine. En même temps, le compte pour les *Quai de la Torunelle* et *Quai de Grenelle* sont restés plutôt similaire de leur tendance précédente. Je sais que les jeux olympiques passaient pendant l'été 2024 au centre-ville de Paris et cette situation pourrait expliquer l'augmentation de *Quai d'Orsay* si le compte des stations *243 Boulevard Saint-Germain* et *Pont d'Alma* n'étaient pas presque la même que 2023 (ou le reste de l'année pour la nouvelle *Pont d'Alma*). Donc, je propose que le compte de *Quai d'Orsay* a mal compté le passage de cycliste pendant les semaines précédentes et suivantes des jeux olympiques, ou que l'utilisation de cette voie pour accéder les sites olympiques a augmenté grandement. D'une façon similaire, j'avance que les voies cyclistes près de la Seine en centre-ville (les stations *Quai d'Orsay*, *Pont d'Alma*, *243 Boulevard Saint-Germain*, et *27 Quai de la Tournelle*) étaient bloquées pour les jeux pendant mi-Juillette au début d'Août. Cette hypothèse est soutenu par le compte normal à la station *173 Boulevard MacDonald* et un compte réduit à la station *36 Quai de Grenelle* (réduit à cause d'un manque des connections typique pour le réseau cycliste).

Il y a deux autres curiosités dans 2024: un manque des données complet à mi-Août et une grosse réduction de la fréquentation cycliste depuis mi-Décembre jusqu'au fin de l'année. Je pense que l'absence des données pendant quelque jours à mi-Août était en conséquence d'une panne du réseau informatique, parce que toutes les stations étaient interrompues. La réduction pendant Décembre est moins claire, mais je postule qu'une grande tempête hivernale est arrivée en mi-Décembre et a apporté des temps tellement frette à Paris.

Objectif 2

> Sur la base de ces seules informations, indiquez quelles recommandations vous feriez à l'organisation CDC si elle souhaite calculer l'évolution annuelle de la fréquentation de la ville entre 2023 et 2024 avec les données dans leur état actuel.

Avec toutes ces informations déjà présentées, je propose trois recommandations pour mieux évaluer l'évolution annuelle de la fréquentation de la ville. Au début, pour 2023, je propose qu'on remplace les données pour la station *27 Quai de la Tournelle* pendant le semaine inhabituel à mi-Août avec le moyen des données journalier pour les 4 semaines précédents. Aussi, je sensibilisais le CDC à l'introduction de la nouvelle station *173 Boulevard MacDonald* en Mars 2023, qui avait une base de fréquentation plus petite que les autres stations (et pourrait produire un moyen annuel plus réduit que prévu).

Pour 2024, il reste deux propositions. D'abord, avec l'interruption du réseau cycliste sur la Seine, il y a deux voies qu'on pourrait suivre. Si le client voudrait calculer la fréquentation attendu pendant Juillet à Août, je le suggérais d'utiliser les données de 2023 avec un augmentation basé sur le reste de les données de 2024. Par exemple, pour la station *Quai d'Orsay*, on pourrait calculer le moyen d'utilisation pendant 2024 (sauf Juillet à Août) et faire le même pour 2023 (encore sans Juillet à Août). Puis, on peut diviser le moyen de 2024 par le moyen de 2023 pour obtenir un facteur d'augmentation pour l'année 2024. Donc, pour cette approche, le CDC pourrait utiliser les données de 2023 avec ce facteur d'augmentation pour obtenir la fréquentation cycliste prévu (avec les ouvertures habituelles des compteurs cyclistes). Si le CDC ne serait pas d'accord avec un calcul des nombres prévus avec le fonctionnement normal, je le conseillerais de laisser les données de Juillet à Août intactes, mais de bien indiquer dans son rapport la durée des jeux olympiques et leur effet sur le réseau cycliste. Et, pour la panne informatique qui a interrompu les données à mi-Août, je suggérais au CDC de faire le calcul du facteur d'augmentation pour 2024 et 2023 et l'appliquer aux données de 2023 pour la même date.

Exercise 2

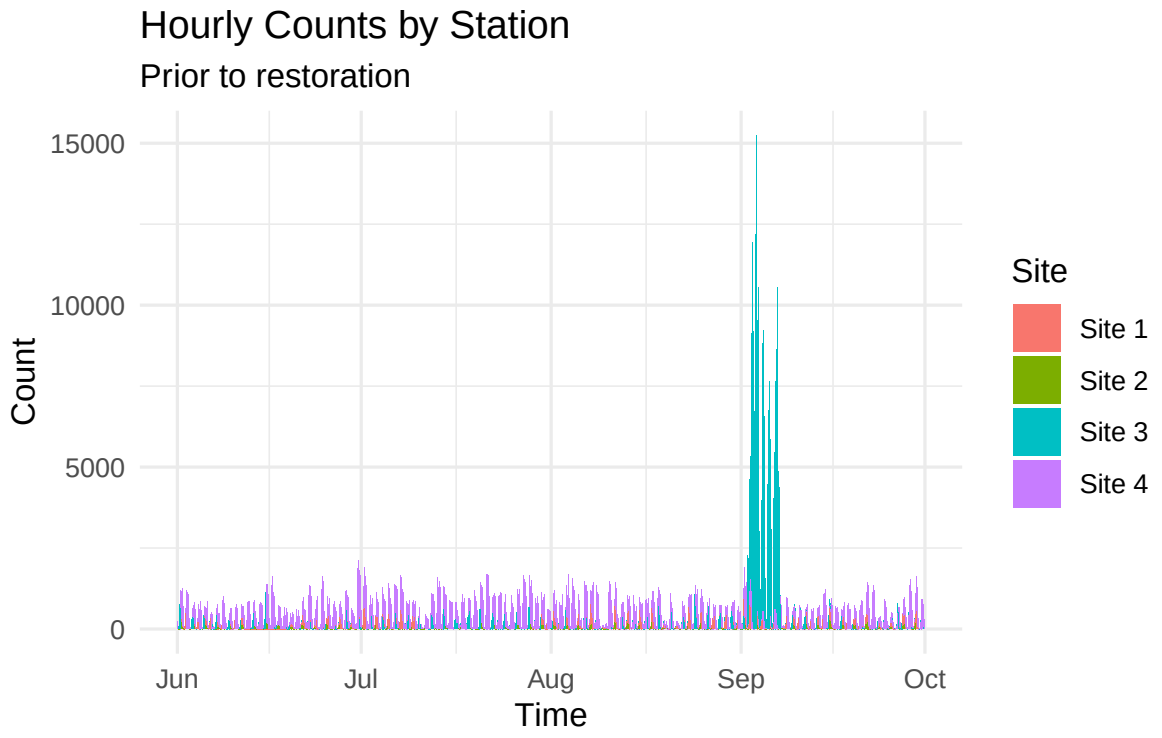
In addition to the bike analysis, the City of Paris wants to analyse pedestrian data in one of their parks during the summer months. But there's a problem (!) – two of their crucial counters have reported unusual counts during the analysis period. They have asked us to correct for this. Hourly count data at the affected counting site as well as two other counting sites in the park with clean data during the summer months is the only information provided.

Objective

Using the available data, provide a complete dataset of the summer months to the City of Paris so they can perform their analysis.

Code set-up

```
original <- readxl::read_xlsx("./Paris park count data summer 2024.xlsx")
  # calling readxl without importing for ease
library(dplyr) # dplyr for data manipulation
library(lubridate) # lubridate for time stuff
library(ggplot2) # ggplot for visualisation

original %>%
  tidyr::pivot_longer(cols = !Time, names_to = "Site", values_to = "Count") %>%
  ggplot() +
  geom_col(aes(x = Time, y = Count, fill = Site), position = "dodge") +
  theme_minimal() +
  labs(
    title = "Hourly Counts by Station",
    subtitle = "Prior to restoration"
  )
```



From the beginning, I identified a few key issues. - Site 1 is missing data for a week in mid-June and more than two weeks in July. - Site 2 includes some extremely high outliers in its data. - Site 3 has extremely high data for the start of September. - Site 4 has some extremely low outliers in its data. - Outages at Site 1, 2, and 3 on August 9, 19, and 21.

Site 1, pt 1.

```
# Site 1
  ## Problem 1: Missing data for 13:18 in June
site1_june_avgs <- original %>%
  select(Time, `Site 1`) %>%
  filter(month(Time) == 6) %>%
  mutate(Day = day(Time),
         Wday = wday(Time),
         Hour = hour(Time)) %>%
  filter(Day != 13:18) %>%
  group_by(Wday, Hour) %>%
  summarise(
    `Average Count` = round(mean(`Site 1`))
```

```
  )

site1_junecleaned <- original %>%
  select(Time, `Site 1`) %>%
  mutate(
    Wday = wday(Time),
    Hour = hour(Time)) %>%
  left_join(., site1_june_avgs, join_by(Wday == Wday, Hour == Hour)) %>%
  mutate(
    `Site 1` = ifelse(month(Time) == 6 & day(Time) %in% 13:18,
                      `Average Count`, # Use average June value
                      `Site 1` # Otherwise, use the existing Site 1 value
  )) %>%
  select(Time, `Site 1`)
```

For Site 1's missing data in June, I calculated the average count for each weekday's hour and substituted this data during the missing period.

Site 2.

```
# Site 2
  ## Problem: extremely high outliers throughout the summer.
    ### 15/6, 16/6, 30/6, 1/7, 21/7, 11/8, 1/9, 2/9, 15/9, 21/9, 28/9, 29/9
#have about eight hours worth of outlying data, but applying the approach
#to all 95th percentile outliers out of an abundance of caution.

site2_95thpct <- original %>%
  pull(`Site 2`) %>%
  quantile(., probs = 0.95)
  # Using a 95th percentile approach to identify 'extreme' outlier values.

site2_avgs <- original %>%
  select(Time, `Site 2`) %>%
  mutate(outlier = ifelse(`Site 2` >= site2_95thpct, T, F),
         Wday = wday(Time),
         Hour = hour(Time)) %>%
  filter(outlier == F) %>%
  group_by(Wday, Hour) %>%
  summarize(
    `Average Weekday Hourly Count` = round(mean(`Site 2`))
```

```
  )

site2_clean <- original %>%
  select(Time, `Site 2`) %>%
  mutate(
    Wday = wday(Time),
    Hour = hour(Time)
  ) %>%
  left_join(., site2_avgs, join_by(Wday == Wday, Hour == Hour)) %>%
  mutate(
    `Site 2` = ifelse(`Site 2` >= site2_95thpct,
                      `Average Weekday Hourly Count`,
                      # Replace with average count if above 95th percentile
                      `Site 2`) # Otherwise keep the same
  ) %>%
  select(Time, `Site 2`)
```

For Site 2, I applied an approach based on identifying its extreme outliers (95th quantile or above) and substituting each weekday's hourly average for the summer period during the outlying dates.

Site 3.

```
# Site 3
  ## Problem: Abnormally high weekly stats at the start of September

  # Identifying the problem period
original %>%
  select(Time, `Site 3`) %>%
  filter(month(Time) == 9 & day(Time) %in% 1:7) %>%
  ggplot() +
  geom_col(aes(x = Time, y = `Site 3`))
```

```
  ## So 23:00 1 Sept-5:00 7 Sept

site3_interval <- ymd_hms("2024-09-01 23:00:00") %--% ymd_hms("2024-09-07 5:00:00")
# Creating a lubridate::interval object for the period

site3_avgs <- original %>%
  select(Time, `Site 3`) %>%
```

```
  filter(!(Time %within% site3_interval) & month(Time) == 9) %>%
  mutate(
    Hour = hour(Time),
    Wday = wday(Time)
  ) %>%
  group_by(Wday, Hour) %>%
  summarize(`Wday Hourly Average` = round(mean(`Site 3`)))
  # Making hourly weekday averages of values in September to use in place of affected values

site3_clean <- original %>%
  select(Time, `Site 3`) %>%
  mutate(
    Hour = hour(Time),
    Wday = wday(Time),
  ) %>%
  left_join(., site3_avgs, join_by(Hour == Hour, Wday == Wday)) %>%
  # Joining the averages and the original data, clarifiying `Hour == Hour` and `Wday == Wday
  mutate(
    `Site 3` = ifelse(Time %within% site3_interval,
                      `Wday Hourly Average`, # Within interval, use average
                      `Site 3` # Otherwise keep original value
                      )) %>%
  select(Time, `Site 3`)
```

For Site 3, I identified the impacted period visually with a plot, and then inspected the data directly. I then replaced the aberrations with the average hourly weekday counts for September.

Site 4.

```
# Site 4
  ## Problem: abnormally low readings during some daytime periods
site4_5thpct <- original %>%
  filter(hour(Time) %in% 8:20) %>% # Making the assumption to only detect
  #outliers during the daytime due to similarities in normal low readings
  #at night and abnormal ones
  pull(`Site 4`) %>%
  quantile(., probs = 0.05) # 5th percentile during daytime

site4_avgs <- original %>%
```

```
  select(Time, `Site 4`) %>%
  mutate(outlier = ifelse(hour(Time) %in% 8:20 & `Site 4` <= site4_5thpct, T, F),
         Hour = hour(Time),
         Wday = wday(Time) # Weekday, to provide more precise averages
         ) %>%
  filter(outlier == F) %>% # Exclude outliers from averages calculation
  group_by(Wday, Hour) %>%
  summarize(
    `Wday Hourly Average` = round(mean(`Site 4`))
  )

site4_clean <- original %>%
  select(Time, `Site 4`) %>%
  mutate(
    outlier = ifelse(hour(Time) %in% 8:20 & `Site 4` <= site4_5thpct, T, F),
    Hour = hour(Time),
    Wday = wday(Time)
  ) %>%
  left_join(., site4_avgs, join_by(Hour == Hour, Wday == Wday)) %>%
  mutate(`Site 4` = ifelse(outlier == T,
                           `Wday Hourly Average`, # If an outlier, replace with average
                           `Site 4` # Otherwise keep the same
                           )) %>%
  select(Time, `Site 4`)
```

For Site 4's extremely low observations, I applied a similar outlier approach by identifying all counts during the day (between 08h00 and 20h00) that fell within the 5th percentile and replaced them with the summer's average hourly weekday counts for the station.

August aberrations.

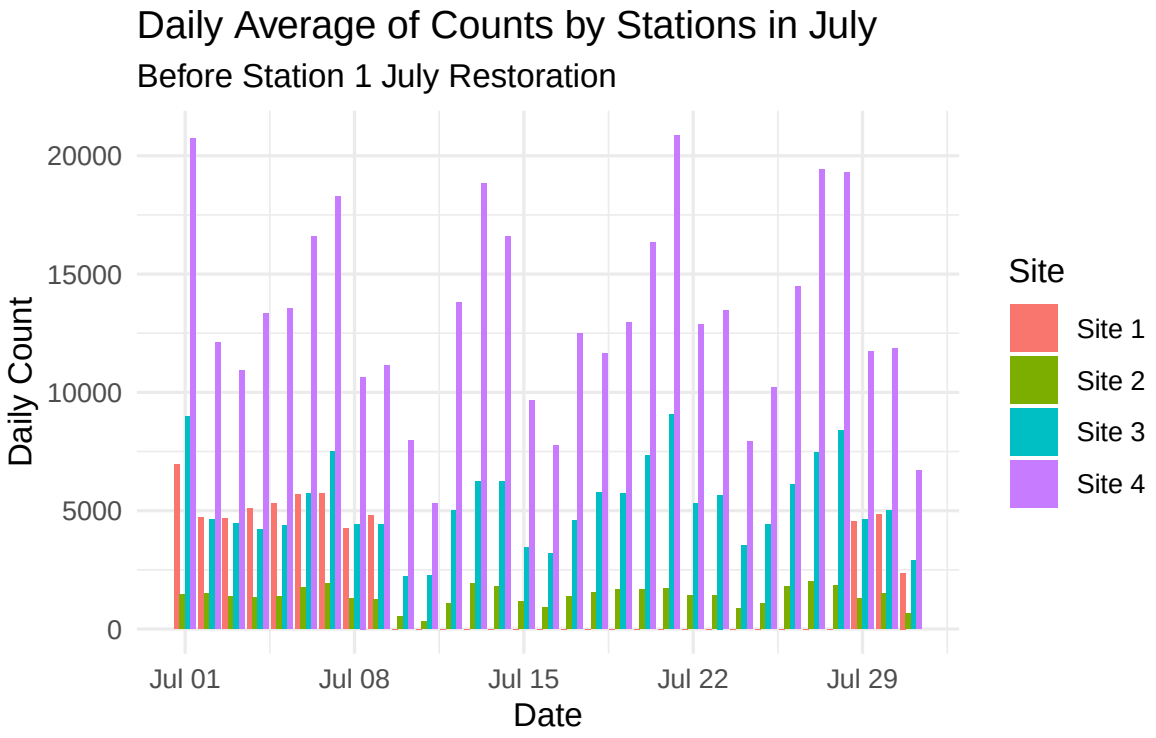
```
# Sites 1, 2, and 3 in August
  ## Outages at Stations 1, 2, and 3 on August 9, 19, and 21

stations_semicleaned <- left_join(
  left_join(site1_junecleaned, site2_clean),
  left_join(site3_clean, site4_clean)
)

  # Applying average August values for these stations
```

```r
problem_dates <- purrr::map_vec(c("2024-08-09", "2024-08-19", "2024-08-21"), ymd)

august_avgs <- stations_semicleaned %>%
  filter(month(Time) == 8 & !(date(Time) %in% problem_dates)) %>%
  select(!`Site 4`) %>%
  tidyr::pivot_longer(cols = !Time, names_to = "Site", values_to = "Count") %>%
  mutate(
    Wday = wday(Time),
    Hour = hour(Time)
  ) %>%
  group_by(Site, Wday, Hour) %>%
  summarize(
    `Average Count` = round(mean(Count))
  )

stations_cleaned <- stations_semicleaned %>%
  tidyr::pivot_longer(cols = !Time, names_to = "Site", values_to = "Count") %>%
  mutate(
    Wday = wday(Time),
    Hour = hour(Time)
  ) %>%
  left_join(., august_avgs, join_by(Site, Wday, Hour)) %>%
  tidyr::pivot_wider(names_from = Site, values_from = Count) %>%
  mutate(
    `Site 1` = ifelse((date(Time) %in% problem_dates) & !is.na(`Site 1`),
                      `Average Count`,
                      `Site 1`),
    `Site 2` = ifelse((date(Time) %in% problem_dates) & !is.na(`Site 2`),
                      `Average Count`,
                      `Site 2`),
    `Site 3` = ifelse((date(Time) %in% problem_dates) & !is.na(`Site 3`),
                      `Average Count`,
                      `Site 3`)
  ) %>%
  select(c(Time, `Site 1`, `Site 2`, `Site 3`, `Site 4`)) %>%
  group_by(Time) %>%
  summarize(
    `Site 1` = sum(`Site 1`, na.rm = T),
    `Site 2` = sum(`Site 2`, na.rm = T),
    `Site 3` = sum(`Site 3`, na.rm = T),
    `Site 4` = sum(`Site 4`, na.rm = T)
  ) # Cleaning data which resulted from `pivot_longer()` and `pivot_wider()`
```

For the abnormal counts at Stations 1, 2, and 3 in August, I identified the impacted dates (August 9, 19, and 21) and replaced each station with its respectly average hourly weekday average for August.

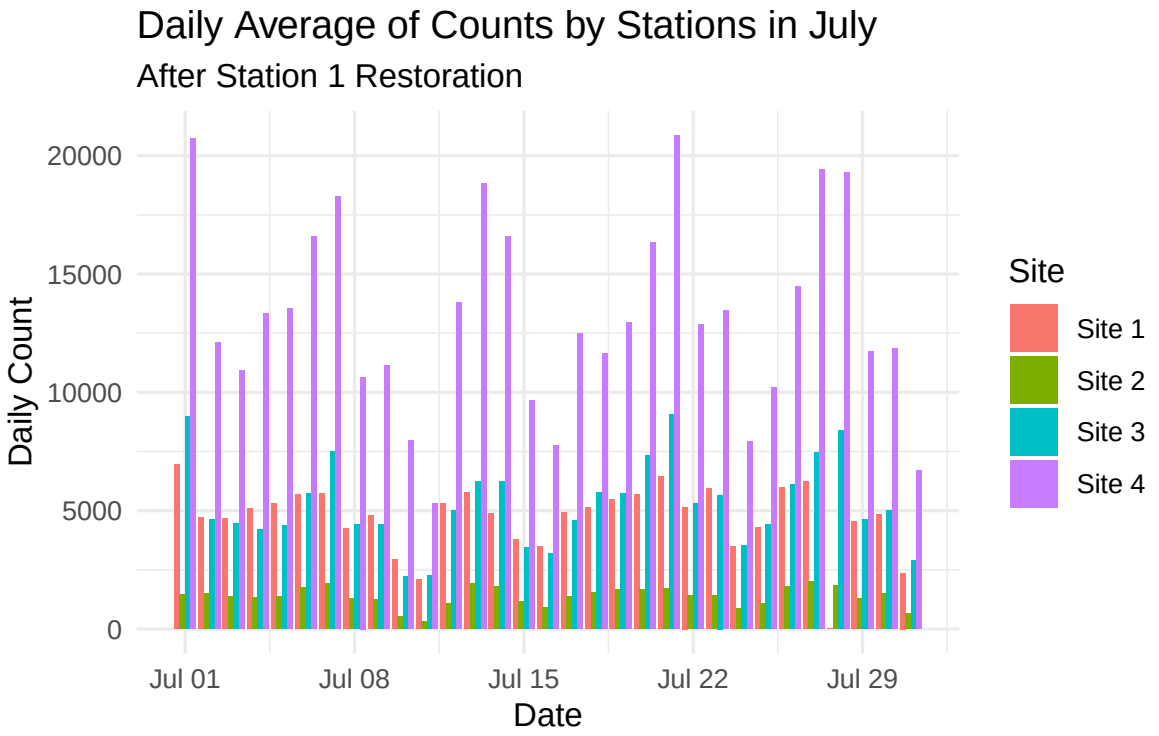
Site 1, pt 2.

```
# Site 1, again
  ## Where are we at now?

stations_cleaned %>%
  filter(month(Time) == 7) %>%
  mutate(
    Date = date(Time)
  ) %>%
  tidyr::pivot_longer(cols = !c(Time, Date), names_to = "Site", values_to = "Count") %>%
  group_by(Site, Date) %>%
  summarize(`Daily Count` = sum(Count)) %>%
  ggplot() +
  geom_col(aes(x = Date, y = `Daily Count`, fill = Site), position = "dodge") +
  theme_minimal() +
  labs(
    title = "Daily Average of Counts by Stations in July",
    subtitle = "Before Station 1 July Restoration"
  )
```




```
  ## Broad approach: comparing the proportional difference of Station 1 to the other station

station1_july_dates <- ymd("2024-07-10") %--% ymd("2024-07-28")

average_proportional_factor <- stations_cleaned %>%
  mutate(
    Wday = wday(Time),
    Hour = hour(Time)
  ) %>%
  filter(!(Time %within% station1_july_dates)) %>%
  mutate(
    `Proportional Factor` = `Site 1` / (`Site 2` + `Site 3` + `Site 4`)
  ) %>%
  filter(`Proportional Factor` != Inf & !is.na(`Proportional Factor`)) %>%
  group_by(Wday, Hour) %>%
  summarize(
    `Average Proportional Factor` = mean(`Proportional Factor`, na.rm = T)
  )

stations_restored <- stations_cleaned %>%
  mutate(
```

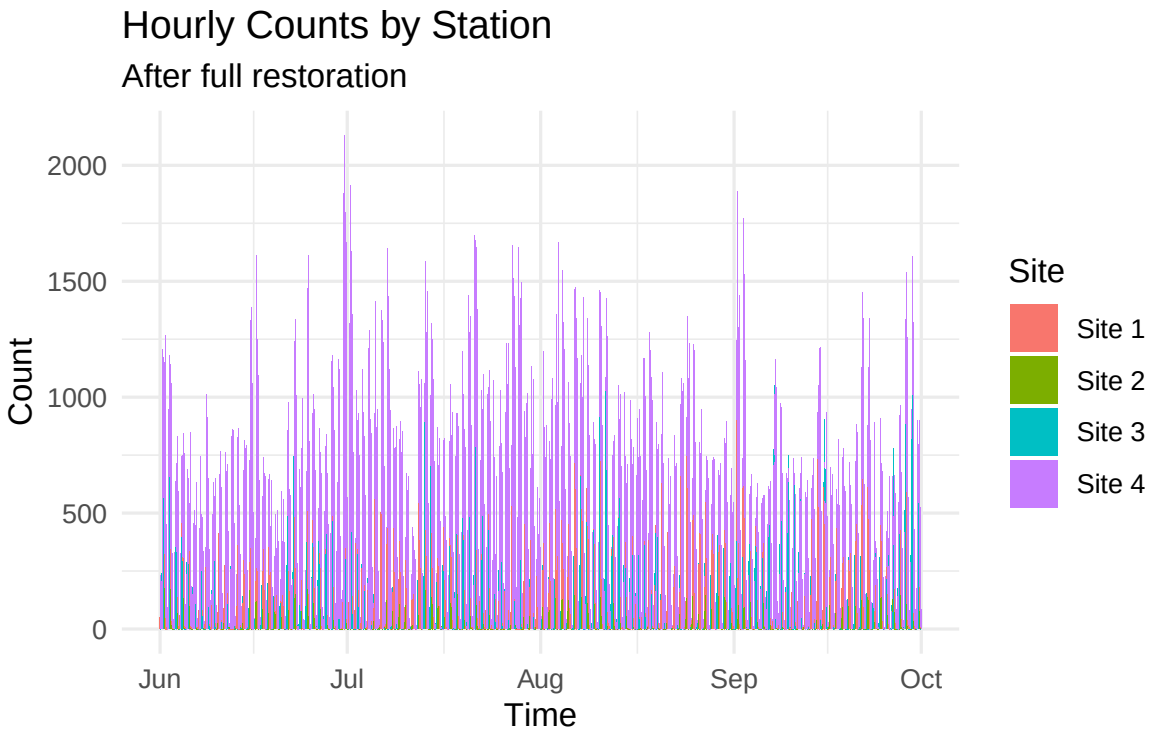
```
    Wday = wday(Time),
    Hour = hour(Time)
  ) %>%
  left_join(., average_proportional_factor, join_by(Wday == Wday, Hour == Hour)) %>%
  mutate(
    `Site 1` = ifelse(Time %within% station1_july_dates,
                      round((`Site 2` + `Site 3` + `Site 4`) * `Average Proportional Factor`)
                      `Site 1`)
  ) %>%
  select(!c(`Average Proportional Factor`, Wday, Hour))

stations_restored %>%
  filter(
    month(Time) == 7
  ) %>%
  mutate(
    Date = date(Time)
  ) %>%
  select(!Time) %>%
  tidyr::pivot_longer(cols = !Date, names_to = "Site", values_to = "Count") %>%
  group_by(Site, Date) %>%
  summarize(`Daily Count` = sum(Count)) %>%
  ggplot() +
  geom_col(aes(x = Date, y = `Daily Count`, fill = Site), position = "dodge") +
  theme_minimal() +
  labs(
    title = "Daily Average of Counts by Stations in July",
    subtitle = "After Station 1 Restoration"
  )
```



For the longer period of missing data at Station 1 in July, I took a different approach. After restoring the rest of the data, I calculated the average proportional difference between the sum of all other sites' hourly counts and Site 1's. I did this while accounting for the day of the week as well. I then applied this over the impacted period in July for Site 1, summing the counts of all other sites and applying this average 'Proportional Factor'.

Final product

```
stations_restored %>%
  tidyr::pivot_longer(cols = !Time, names_to = "Site", values_to = "Count") %>%
  ggplot() +
  geom_col(aes(x = Time, y = Count, fill = Site), position = "dodge") +
  theme_minimal() +
  labs(
    title = "Hourly Counts by Station",
    subtitle = "After full restoration"
  )
```




```
readr::write_csv(stations_restored, "Restored Paris park count data 2024.csv")
```

Ultimately, the data is much cleaner and includes no significant gaps in observations or significant aberrations.