RAPPORT DE PROJET TER

Sujet : Fouille et parking

Préparé par
BABOU Ghiles
CARDINALE-CORTES Melissa
HAMMAD Amir
RAMAROMANANATOANDRO Thomas
YOUSFI Yacine

Projet Master 1 Informatique 2021

Université de Versailles-Saint-Quentin-en-Yvelines

UVSQ université PARIS-SACLAY | UFR des Sciences CAMPUS DE VERSAILLES

Table des matières

1 Choix du sujet

Étudiants en première année de Master Informatique nous avons pour l'UE TER eu le choix parmi une vingtaine de sujets dont quelques-uns étaient conseillés pour les étudiants poursuivant un Master Datascale. Par ailleurs du fait que le groupe soit composé de 4 étudiants exclusivement en Master Datascale.
Et afin de se familiariser un peu plus avec les outils et de se préparer aux compétences que nous réinvestirons plus tard dans notre domaine d'étude.
Notre attention s'est toute portée vers ces sujets en priorité. Nous avons donc décidé de choisir un sujet en rapport avec la fouille et l'analyse de données et plus précisément : Fouille et parking.
Par la suite nous expliquerons les techniques et outils employées, les problèmes rencontrés et nos analyses afin de répondre au sujet.

2 Objectifs du projet

Le projet consiste à extraire des données d'un jeu de données, ici le SAEMES OpenData sur une durée conséquente qui sont rafraichies en temps réel (1 à 2 min).
Ces données sont relatives aux parkings de l'entreprise SAEMES situés dans la région Parisienne. En fonction de ce recueil de données, un nettoyage est accompli afin d'avoir des données fiables pour réaliser une analyse statistique dans l'optique de classifier chaque parking sur le critère de leur occupation.

3 Première partie : extraction des données

3.1 Techniques de récupération

Dans un premier temps, il s'agit d'établir une méthode afin de récupérer les données nécessaires de manière continue et journalière.
Le site https://opendata.saemes.fr/explore/dataset/places-disponibles-parkings-saemes/table/?sort=nom_parking met à disposition une api afin de télécharger les fichiers en csv ou json des places disponibles pour les 21 parkings parisiens.
Nous avons donc utilisé le lien de l'API afin de récupérer des résultats sous format csv, comme ci-dessous :

```
Date;Nom parking;Type de parc;Coordonnées;Horaires d'accès au public (pour les usagers non abonnés);Code parking;Facilityid;Type de compteur;Places disponibles
2020-06-24T23:08:00+02:00;;;;;PMU84;1604;MOTOR_BIKE;0
2020-06-24T23:08:00+02:00;;;;;PMO05;1452;DISABLED;0
2020-06-24T23:09:00+02:00;;;;;PMO84;1603;MOTOR_BIKE;8
2020-06-24T23:10:00+02:00;;;;;PMO62;1805;MOTOR_BIKE;0
2020-06-24T23:08:00+02:00;;;;;PMO05;1452;ELECTRIC_CAR;0
2021-03-12T20:18:00+01:00;Parking Rivoli-Sébastopol;public;48.858396,2.350484;24h/24, 7j/7;PMO23;1851;MOTOR_BIKE;11
2021-03-12T20:18:00+01:00;Parking Meyerbeer Opéra;public;48.871505,2.333759;24h/24, 7j/7;PMO08;1401;STANDARD;439
2021-03-12T20:17:00+01:00;Parking Lagrange-Maubert;public;48.850541,2.348671;24h/24, 7j/7;PMO06;1753;DISABLED;5
2021-03-12T20:17:00+01:00;Parking Anvers;public;48.882697,2.344013;24h/24, 7j/7;PMO04;1151;ELECTRIC_CAR;0
```

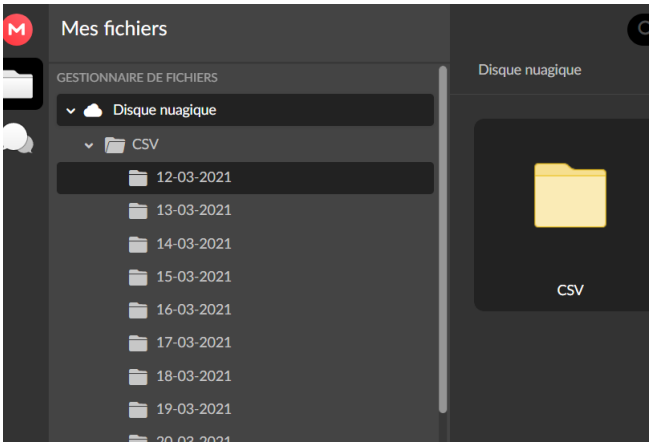
Pour l'obtention de ces données nous avons utilisé une solution de serveur hébergé cloud afin d'exécuter le script de récupération des données.
En effet, l'impossibilité d'utiliser une seule machine qui tourne en continu 24H/24 et 7J/7 nous a fait pencher pour cette solution.
Le script en question est écrit en python et utilise les librairies requests, pandas et sqlalchemy.

parkings.py

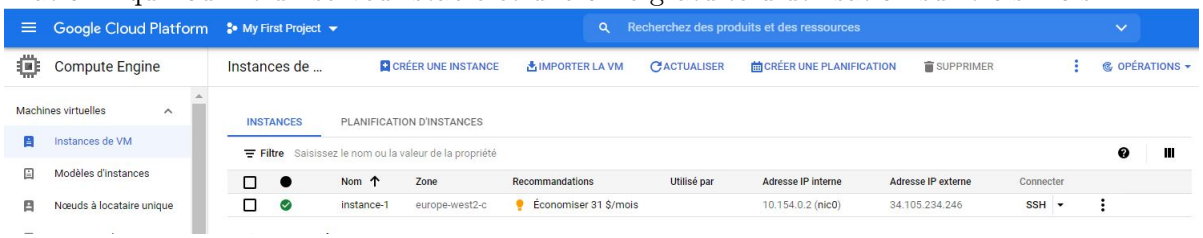
```python
import requests
import pandas as pd
import sqlalchemy as sqla

url = "https://opendata.saemes.fr/explore/dataset/places-disponibles-parkings-saemes/download/?format=csv&timezone=Europe/Berlin&lang=fr&use_labels_for_header=true&csv_separator=%3B
req = requests.get(url)
url_content = req.content
csv_file = open('downloaded.csv', 'wb')
csv_file.write(url_content)
csv_file.close()

df = pd.read_csv('downloaded.csv',sep=';',skiprows=range(1,6))


df['Date'] = df['Date'].str[0:10] +' '+df['Date'].str[11:19]
df = df.rename(columns={'Date': 'horodatage','Nom parking': 'nom','Type de parc': 'type_parking',"Horaires d'accès au public (pour les usagers non abonnés)": 'horaires',
    'Code parking': 'code_parking','Type de compteur': 'type_compteur', 'Places disponibles': 'places_disponibles'})
df['horodatage'] = pd.to_datetime(df['horodatage'])
df_occupation = df.loc[: ,['code_parking','type_compteur','horodatage','places_disponibles']]
print(df_occupation)



host = 'rt162565-001.dbaas.ovh.net'
port = '35219'
db = 'sppdter21'
user = 'admin'
psw = 'FppdsAMYTG5'
name_table = 'OCCUPATION_PARKING'

# dialect+driver://username:password@host:port/database
engine = sqla.create_engine('mysql://'+user+':'+psw+'@'+host+':'+port+'/'+db)
print('CONNECTED!')

df_occupation.to_sql(name_table,engine,if_exists='append',index=False,chunksize=1024,dtype={'id': sqla.Integer,'code_parking': sqla.String(255),'type_compteur': sqla.String(255),
    'horodatage': sqla.DateTime,'places_disponibles': sqla.Integer})
print('Finished export to Database!')
```

La première idée que nous avons eu a été d'utiliser le service gratuit amazon aws EC2 instance afin d'exécuter notre script sur une instance VM de type Linux Debian, mais cela ne s'est pas dérouler sans encombre.
En effet, nous avons eu une interruption du serveur inopinée, il se trouve que notre instance a eu à plusieurs reprises des problèmes de checking de sécurité provoquant un arrêt de la récolte et une perte de données le Lundi 29 Mars de 00h00 à 15h15.

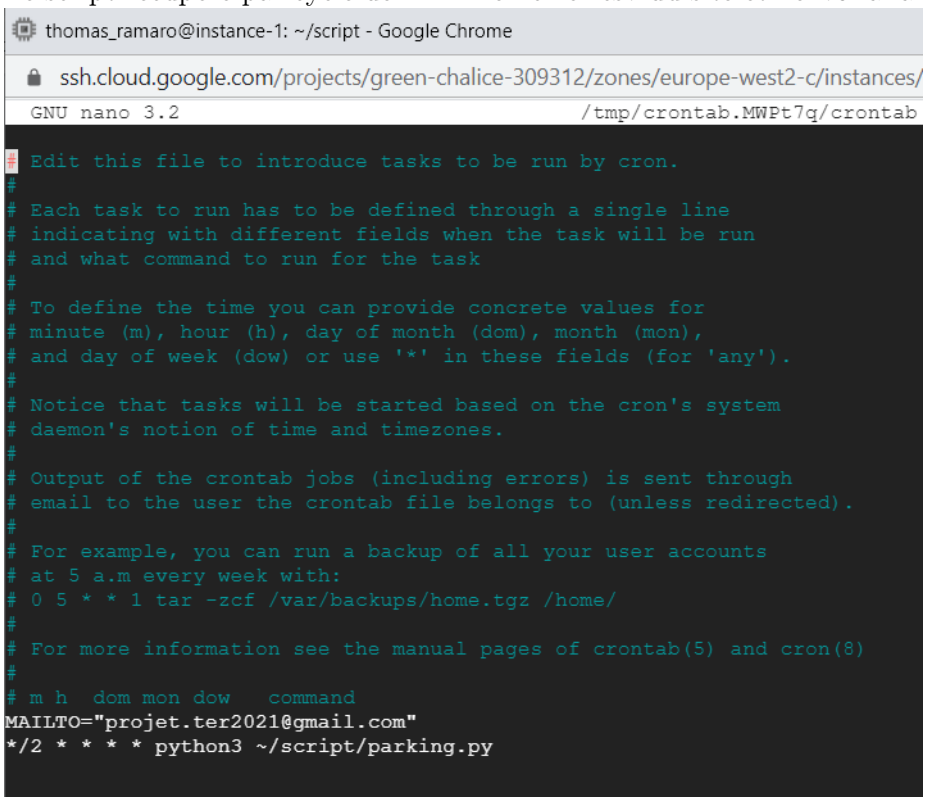
Par la suite, nous avons décidé d'utiliser un "filet" de sécurité afin d'éviter ce type de problème récurrent. Nous avons créé un script afin d'envoyer en parallèle les données csv dans un service d'hébergement de fichiers appelé Mega.io.



Par ailleurs nous sommes passés sur un autre service d'hébergement d'instance VM cloud avec Google Cloud Platform qui fournit un serveur stable et une offre gratuite d'utilisation sur trois mois.



Pour ce qui est de la partie de la récolte en continu 24H24 et 7J/7, nous avons utilisé un job/tâche CRON dans l'instance de VM Ubuntu de Google Cloud Platform qui est un programme qui permet aux utilisateurs des systèmes Unix d'exécuter automatiquement des scripts, des commandes ou des logiciels à une date et une heure spécifiée à l'avance, ou selon un cycle défini à l'avance.
Le script récupère par cycle de 2 min le fichier csv du site et l'envoi à la base de données décrite par la suite.



3.2 Stockage en base de données

Le choix des tables de la base de données nous a amené à une première idée au début du projet qui s'articule en une table comme suit :
PLACES_PARKING(id, code_parking, type_compteur, type_parking, nom, horaires,
horodatage, places_disponibles).

Cette solution n'a pas été retenue après test sur une première solution de stockage de données MySQL en ligne qu'est Heroku ClearDB.

En effet, l'espace de 500mb gratuit a vite été dépassé ce qui nous a conduit à revoir notre solution de stockage en base de données.

À l'aide des conseils du professeur référant nous avons refait notre schéma de table et avons opté pour une structure de deux tables afin de réduire au minimum l'espace de stockage nécessaire.
L'idée de ces deux tables étant de séparer la partie dite "statique" de la partie dite "dynamique".

La partie statique correspond à la donnée ne changeant pas (ou très possiblement peu) de valeur dans le temps, elle est décrite par la table :
PARKING(code_parking, nom, zone, etiquette, horaires)
Les attributs zone et étiquette ont pour but de situer approximativement dans la zone géographique le parking, ils prennent les valeurs suivantes :
ATTRIBUT ZONE :
CENTRE
EXTRA-MUROS

ATTRIBUT ETIQUETTE :
CULTURE
RELAIS
HOPITAL
AUTRE

ZONE centre :
Parking Anvers
Parking Bercy Seine
Parking Charléty Coubertin
Parking Hôpital Robert-Debré
Parking Hôpital Saint Louis
Parking Hôpital Sainte Anne
Parking Hôtel de Ville
Parking Lagrange-Maubert
Parking Mairie du 17ème
Parking Maubert Collège des Bernardins
Parking Meyerbeer Opéra
Parking Méditerranée - Gare de Lyon
Parking Odéon-Ecole de Médecine
Parking Porte d'Orléans
Parking Pyramides
Parking Reuilly-Diderot
Parking Rivoli-Sébastopol

ZONE extra-muros :
Parking Hôpital Henri Mondor
Parc Relais Vaires Torcy
Parking Vaires Centre Ville
Parc Relais Val d'Europe - Serris Montévrain

ETIQUETTE CULTURE :
Parking Bercy Seine
Parking Maubert Collège des Bernardins
Parking Meyerbeer Opéra
Parking Odéon-Ecole de Médecine
Parking Pyramides
Parking Charléty Coubertin

ETIQUETTE TRANSPORT :
Parc Relais Vaires Torcy
Parc Relais Val d'Europe - Serris Montévrain
Parking Méditerranée - Gare de Lyon

ETIQUETTE HOPITAL :
Parking Hôpital Henri Mondor
Parking Hôpital Robert-Debré
Parking Hôpital Saint Louis
Parking Hôpital Sainte Anne

ETIQUETTE AUTRE :
Parking Hôtel de Ville
Parking Lagrange-Maubert
Parking Mairie du 17ème
Parking Porte d'Orléans
Parking Reuilly-Diderot
Parking Rivoli-Sébastopol
Parking Vaires Centre Ville
Parking Anvers

Par la suite dans la partie analyse, nous avons retiré l'étiquette autre et mis les parkings correspondants en étiquette hopital, transport ou administratif.

La table de la partie dynamique ci-dessous porte la donnée qui est centrale dans notre étude qui est le nombre de places disponibles, l'horodatage et le type de compteur.
OCCUPATION_PARKING(id, code_parking, type_compteur, horodatage, places_disponibles)

Le type de compteur décrit le type de parking concerné, les informations du site SAEMES nous indique que :
STANDARD = places pour les voitures (véhicules légers)
MOTOR_BIKE = places pour les motos et les scooters
ELECTRIC_CAR = places équipées de recharges pour les véhicules électriques
TRUCK = places pour les camions

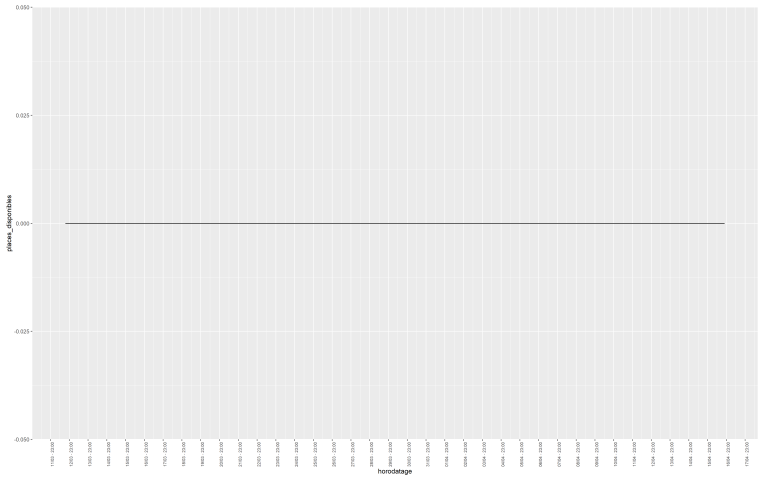
Quelques informations complémentaires à notifier sont que l'extraction des données s'est déroulée du 12 mars À 18h54 au 24 mai à 11h32 et le confinement lié à la covid-19 en France
a eu lieu du 3 avril au 3 mai 2021.

4 Nettoyage des données stockées

4.1 Détection des anomalies

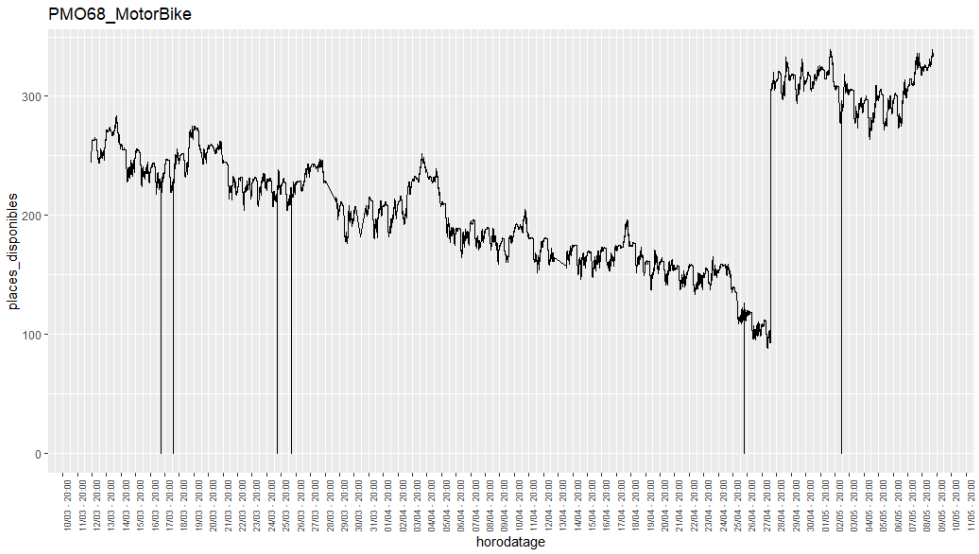
Afin de détecter les possibles anomalies dans les données collectées, nous avons d'abord généré les courbes des séries temporelles de chaque parking à l'aide de R. Pour les générer nous avions d'abord utilisé la fonction geom_smooth mais nous nous sommes rendus compte qu'elle lissait la courbe et faisait disparaître les points trop écartés de la moyenne. Cela ne nous permettait donc pas de repérer des anomalies et nous avons par la suite opté pour la fonction geom_line qui relie simplement chaque point.
Grâce aux courbes obtenues, il a été plus simple de détecter des comportements suspects.

Pour certaines courbes, nous avons observé un signal plat, sans variation pouvant être expliqué par des capteurs désactivés ou un parking fermé.
En voici un exemple (parking Maubert Collège des Bernardins - PMO05 Standard) :



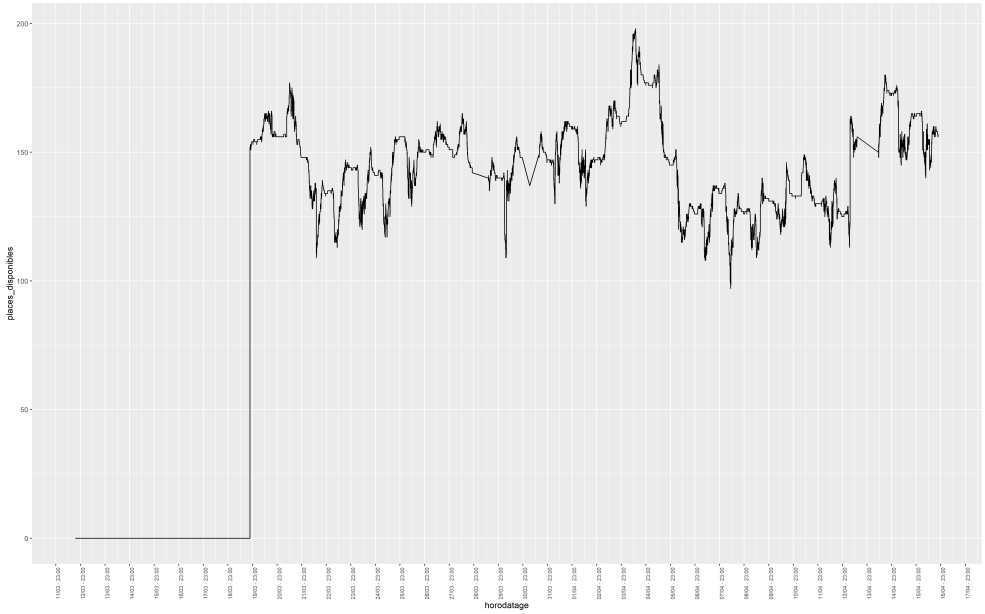
Parking PMO05 inactif

Nous avons également pu observer de grandes variations du nombre de places disponibles en très peu de temps ou une descente subite du nombre de places disponibles à 0 sûrement dues a un capteur défectueux.



parking PMO68 avant nettoyage

Aussi, on peut par exemple constater ici pour le parking PMO04 (parking Anvers) que du 12/03 au 19/03 il n'y a aucune place disponible puis on passe subitement à 150 places disponibles. Nous pouvons faire l'hypothèse que les capteurs était éteints pendant cette période puis ont été activés à nouveau



Parking PMO04 avant nettoyage

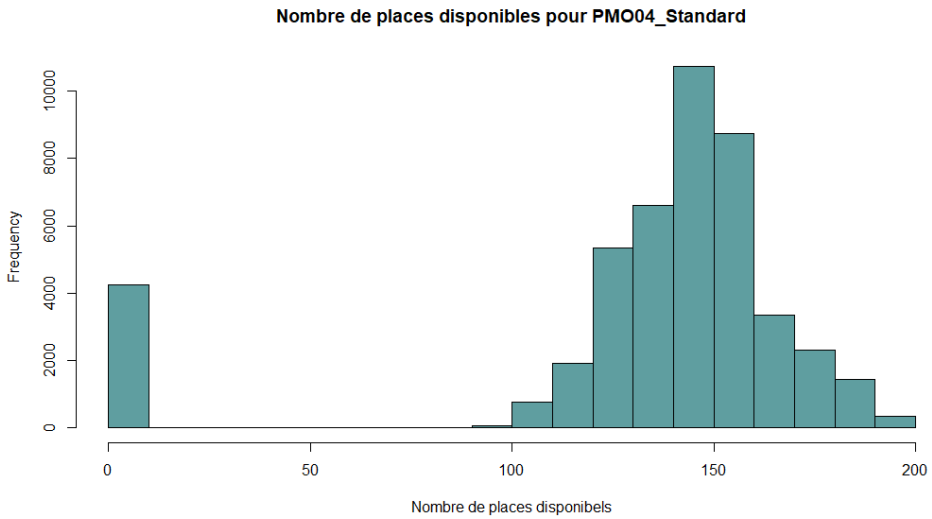
Nous avons ainsi pu éliminer un certain nombre de parkings inactifs, inexploitables ou ayant peu de nombre de places tels que les parkings de type motor_bike ou disabled où généralement le nombre de places ne dépasse pas 10, et nous avons sélectionné les parkings pertinents à nettoyer. Exemple de parkings éliminés : Bercy Seine : le type de parking “STANDARD” est à 254 du 19/03 à 21 :57 puis 275 à 21 :59 (on estime un bug ou une expiration de réservation massive)

Odéon-Ecole de Médecine : le type de parking “DISABLED” et “ELECTRIC_CAR” constamment à 0 du 12/03/2021 au 16/04/2021 Le type de parking “STANDARD” est à 0 le 14/04/2021 à 12 :03 puis à 92 à 12 :07 (on estime un bug capteur ou une expiration de réservation massive)

Pour agrémenter ces premières analyses, nous avons généré des histogrammes ainsi que des boxplots afin de repérer et supprimer les "outliers", c’est-à-dire les valeurs trop éloignées de la moyenne et donc probablement erronées.

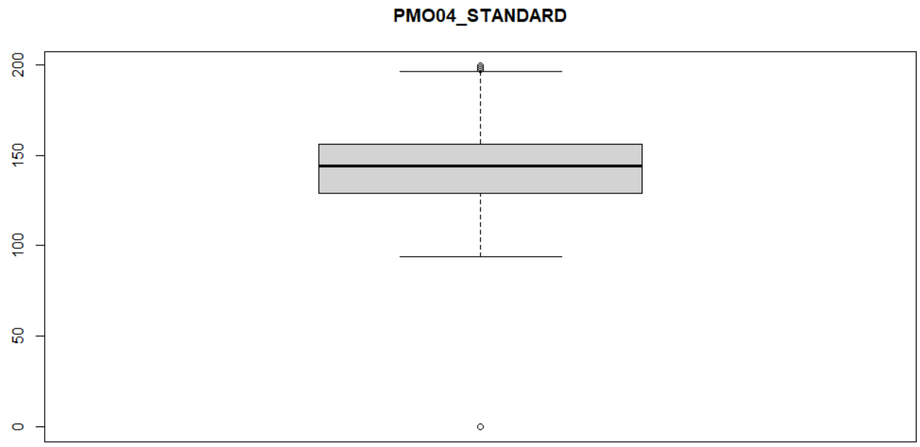
Sur les histogrammes, il est facile de repérer des valeurs aberrantes, car ce sont généralement des barres isolées aux extrémités.

On peut repérer sur cet exemple une barre éloignée représentant le nombre de fois que le nombre de places disponibles est à 0. En effet, cela correspond bien aux valeurs de 0 du début de la série temporelle présentée ci-dessus.



Histogramme du nombre de places disponibles pour PMO04 avant nettoyage

Les boxplots (ou boîtes à moustache) permettent d'afficher la médiane, les 1^{ers} et 3^{eme} quartiles ainsi que les valeurs minimums et maximums. Les outliers sont facilement repérables, car ils sont représentés par des points au-dessus du maximum ou en dessous du minimum.
On voit ici par exemple des outliers de valeurs 0 qui correspondent aux 0 au début de la série temporelle présentée plus haut ainsi qu'a la barre à l'extrémité gauche de l'histogramme ci-dessus.



Boxplot pour PMO04 avant nettoyage

4.2 Nettoyage des données

```r

#cette ligne de code nous permet de generer le boxplot ci-dessus
boxplot(PMO04$places_disponibles,main="PMO04_STANDARD")

#le vect4 recupère les données aberante du boxplot
vec4<-boxplot.stats(PMO04$places_disponibles)$out;

#remplacement des valeurs aberantes qu'on a reccuperer su vec4 par la valeur NA
PMO04$places_disponibles[PMO04$places_disponibles %in% vec4]<-NA

#sauvegarde des nouvelles donnees dans un nouveau fichier CSV
write.csv(PMO04,file="C:/Users/Ghiles BM/Desktop/R_Dossier/Données_Nettoyer/PMO04.csv")
```

script de nettoyage en R

Sur la première ligne du code nous générons un boxplot qui nous permet de visualiser les valeurs aberrantes (Outliers), ces valeurs sont récupérées ensuite par le deuxième ligne de code et sont mises dans un vecteur(ici vec4).
L'image ci-dessous montre l'ensemble des valeurs aberrantes récupérées par le vecteur (vec4).

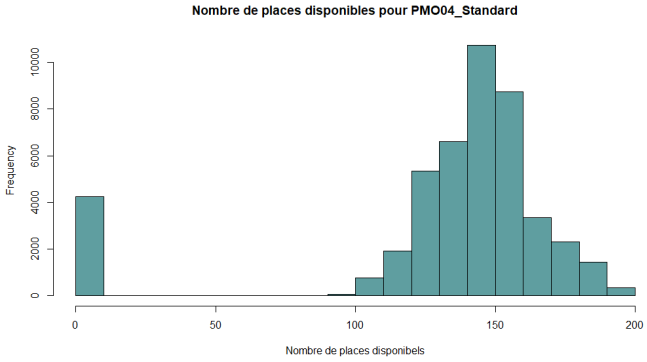
```
> vec4
   [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  [48] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  [95] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [142] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [189] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [236] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [283] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [330] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [377] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [424] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [471] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [518] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [565] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [612] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [659] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [706] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [753] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [800] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [847] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [894] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [941] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [988] 0 0 0 0 0 0 0 0 0 0 0 0 0
 [ reached getOption("max.print") -- omitted 3287 entries ]
> 
```

Contenu du vecteur vec4

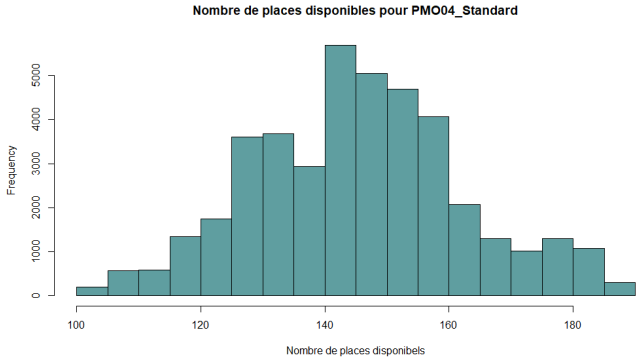
Ensuite sur le dataframe de ce parking, les données qui se trouvent dans ce vecteur sont remplacées par NA (not available) et sont sauvegardées à nouveau sur un fichier csv que nous avons exploité par la suite pour

l'analyse de données.

On peut également constater le résultat du nettoyage des données en générant à nouveau un histogramme de fréquences des places disponibles du dataframe nettoyé.

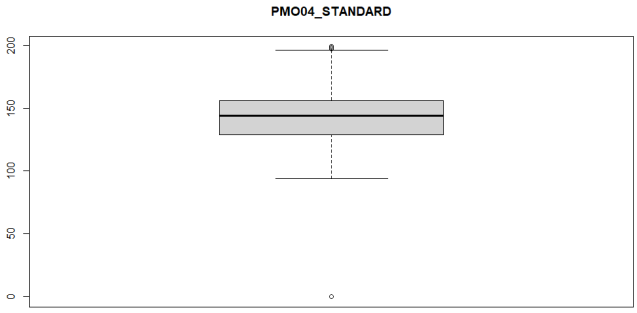


Boxplot du parking PMO04 avant nettoyage

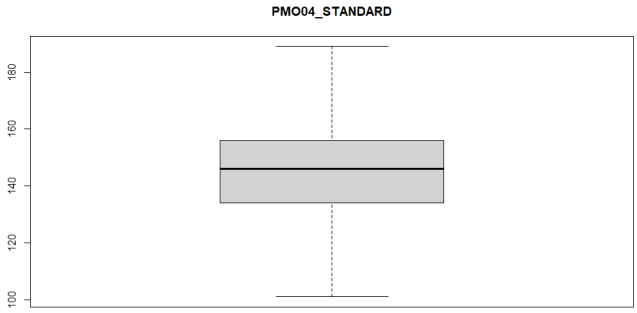


Histograme du parking PMO04 après nettoyage

Après nettoyage des données abérrantes, on peut générer un nouveau box plot des places disponibles du nouveau jeu de données obtenu. On peut alors constater l'absence des points qu'on pouvait voir sur le boxplot généré avant le nettoyage au-dessus de la valeur maximale du jeu de données ou en-dessous de sa valeur minimale.



Boxplot du parking PMO04 avant nettoyage

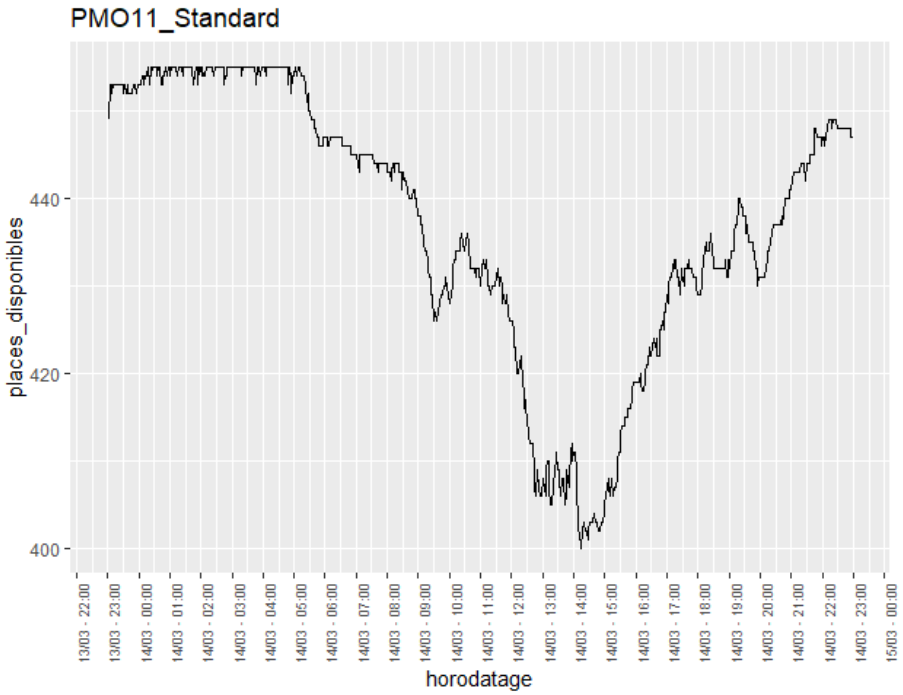


Histograme du parking PMO04 après nettoyage

5 Analyse des données

L'objectif de cette partie est de tracer des graphes avec R afin d'observer le taux d'occupation des différents parkings. En effet l'analyse est basée sur les deux modèles suivants : le taux d'occupation de ces parkings pendant les week-ends et en semaine puis les parkings qui sont à proximité. Les graphes réalisés avec RStudio sont des graphes de parkings de type STANDARD.

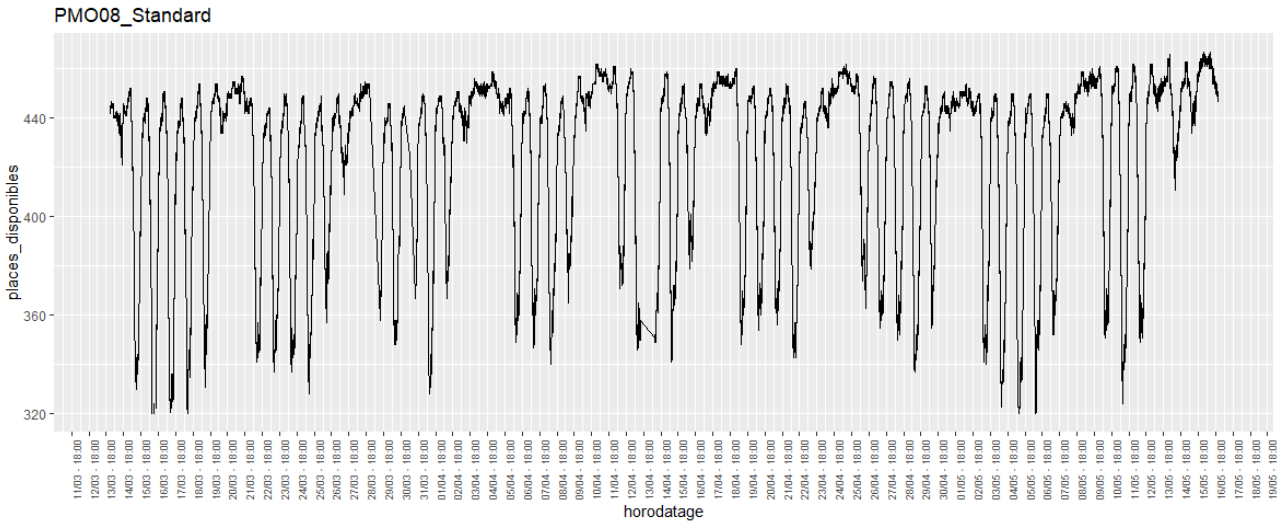
5.1 Taux d'occupation pendant la journée



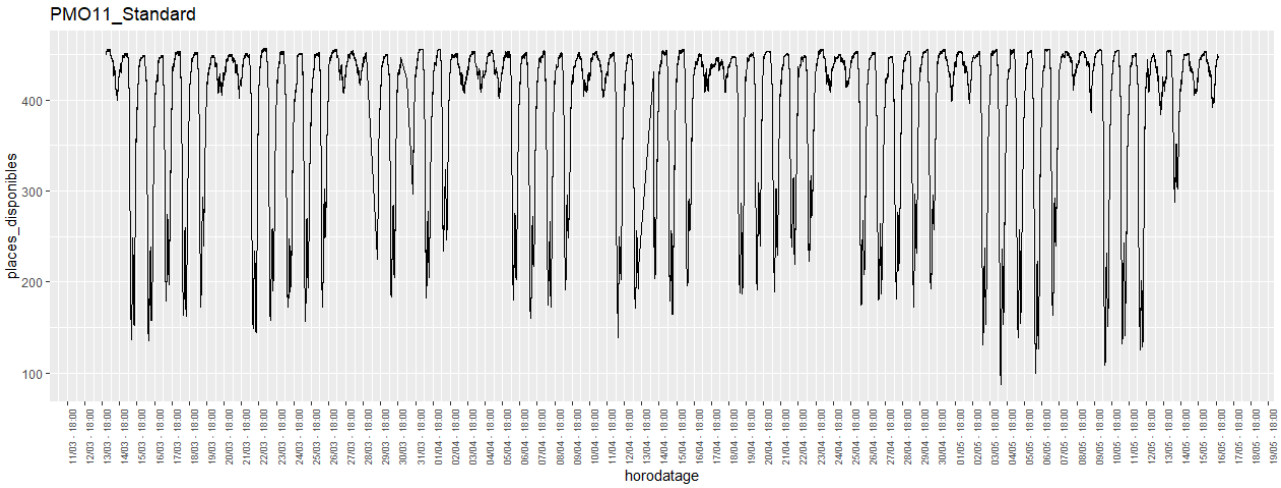
Parking Hôpital Henri Mondor

Ce graphe représente le taux d'occupation du parking Hôpital Henri Mondor pendant toute la journée du 14 mars. Tout d'abord de minuit à 6h du matin, il y a peu d'affluence étant donné que c'est la nuit ce qui se reflète sur le graphe par un nombre de places disponibles restant globalement autour de 455 places. Par la suite on voit une diminution du nombre de places disponibles qui s'explique par des entrées de véhicules dans ce parking. À 15h, on voit une croissance du nombre de places disponibles engendrée par la sortie de véhicules de ce parking. Le graphe illustre donc ce qui se passe généralement à l'echelle d'une journée : un nombre de places disponibles qui varie très peu pendant la nuit puis qui diminue durant la journée et finit par augmenter à nouveau le soir.

5.2 Taux d'occupation pendant le week-end



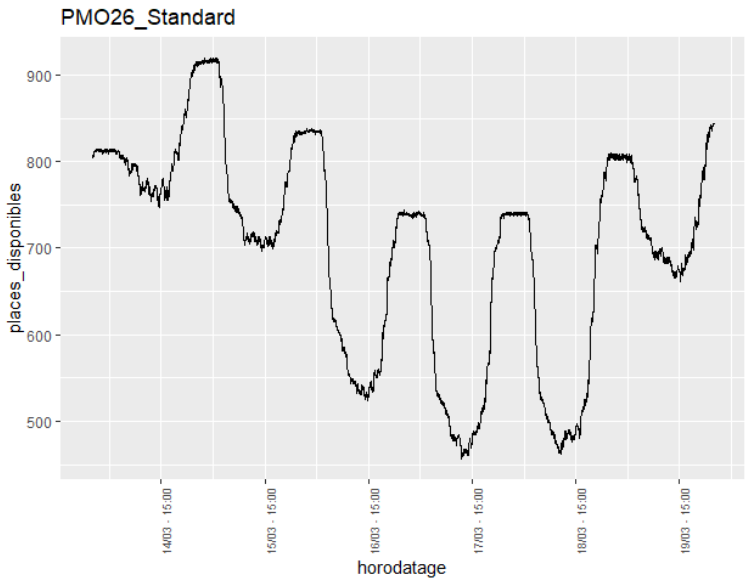
Meyerbeer Opéra



Henri Mondor

Sur ces deux graphes représentant des données récoltées du 11/03 au 19/05, on peut observer des variations du nombre de places disponibles suivant un cycle régulier. Il s'agit d'une période de deux jours avec un nombre élevée de places libres, qui correspondent aux jours du week-end.
Le premier exemple est pendant la période du 19/03 au 21/03 où on observe que le week-end, le nombre de places disponible augmente considérablement comparé à la semaine. En semaine le nombre de places disponibles minimum est de 320 pour Meyerbeer Opéra et environ 100 pour Henri Mondor. En week-end, cependant on a au minimum 410 places disponibles pour le parking Meyerbeer Opéra et 400 pour Henri Mondor.
On en déduit qu'il existe une régularité du nombre de places durant la semaine par rapport aux week-ends.
On aurait pu penser que le nombre de places le week-end soit supérieur à celui de la semaine cependant la récolte de données a été effectuée en période de confinement et de couvre-feu ce qui explique le nombre de places disponible assez élevé les week-ends.

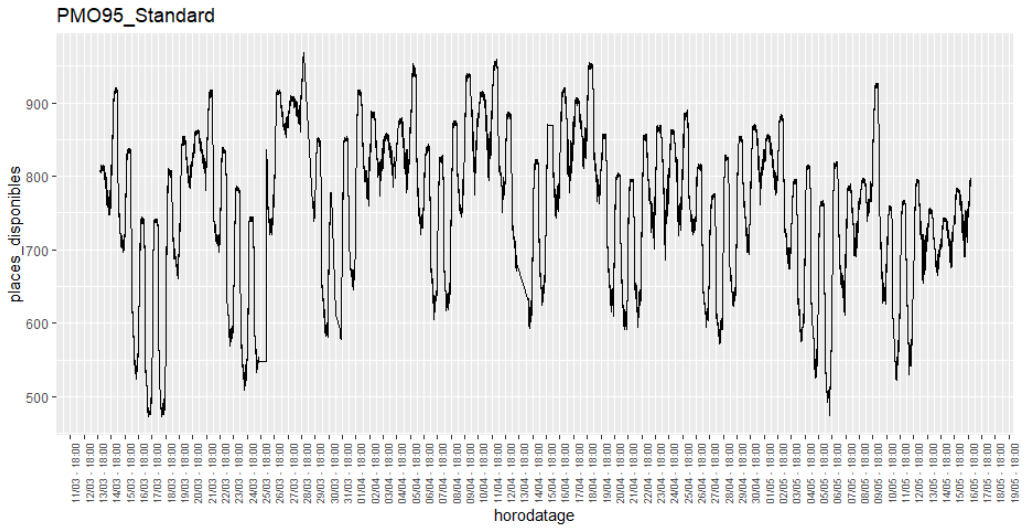
5.3 Taux d'occupation pendant la semaine



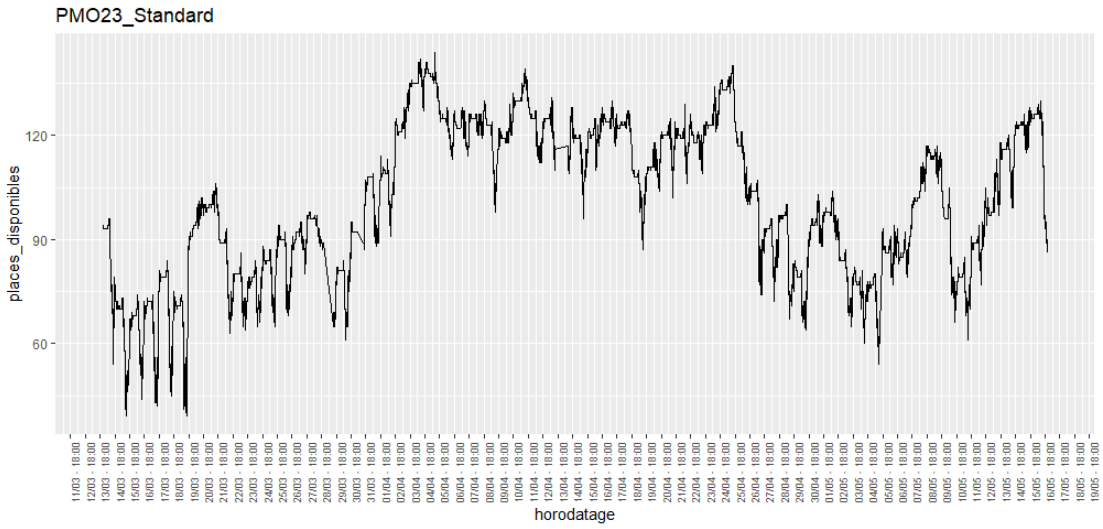
Méditerranée-Gare de Lyon

Ce graphe représente des données récoltées du 14/03 (dimanche) au 19/03 (vendredi) pour le parking PMO26 Méditerranée-Gare de Lyon.
On observe un motif qui se répète pendant cette période : le nombre de places disponibles diminue (arivée de voitures le matin) puis évolue peu (en journée). Ensuite, le nombre de places disponibles augmente (départ de voitures en fin de journée) et forme un plateau (nombre de places évolue peu pendant la nuit).
Bien qu'on observe ce motif pour tous les jours de cette période on peut observer que le nombre de places disponibles le dimanche est le plus élevé sur cette période, car les données ayant été récoltées durant la pandémie de Covid-19, il n'était pas recommandé de sortir le week-end à ce moment-là. Le nombre de places disponibles diminue ensuite à partir de lundi tout en suivant le motif décrit plus haut et ce jusqu'au vendredi ou il augmente à nouveau.
On peut justifier cette diminution en faisant l'hypthèse que les propriétaires des véhicules arrivant le lundi dans le parking, le quittent le vendredi ou alors que le parking a tout simplement une plus grande affluence en semaine et qu'il est occupé par des personnes travaillant à proximité.

5.4 Parkings voisins



Hôtel de Ville

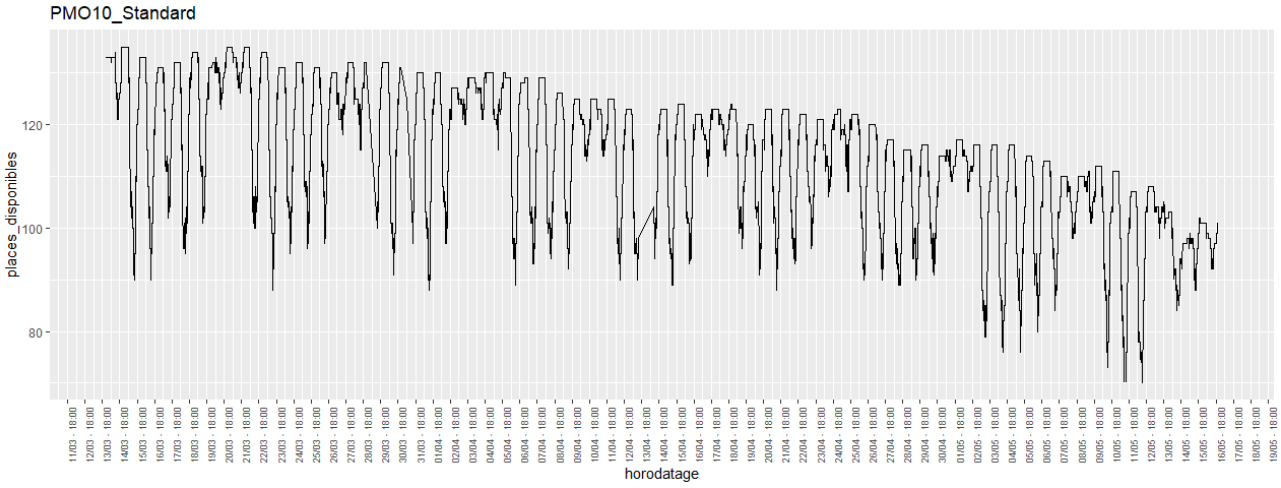


Rivoli Sébastopol

Les deux parkings ci-dessus sont à proximités l'un de l'autre. Chacun d'eux présente un comportement et un fonctionnement unique comme on peut le remarquer sur les deux figures ci-dessus et ce, malgré leur proximité (quelques mètres).
Le parking hôtel de ville présente une saisonnalité dans son occupation durant les semaines et on observe un motif qui se répète. Le parking de rivoli Sébastopol, lui a changé complètement son comportement pendant le confinement, on remarque une augmentation globale du nombre de places disponible.
Pour cet exemple, on peut conclure que la proximité n'a pas d'influence sur le fonctionnement des deux parkings et que la forte demande de stationnement est repartie sur les deux parkings. Une étude du marché a certainement était réalisé pour vérifier la nécessité d'un second parking et explique la présence de deux parkings, en l'occurrence on constate qu'un troisième parking n'est pas nécessaire après notre analyse.

5.5 Parkings des Hopitaux

Nous avons jugé intéressent d'analyser les parkings des hôpitaux durant la période de confinement qui a eu lieux entre le 28 mars 2021 et le 30 mai 2021 suivis d'un couvre-feu.



Sainte-Anne

Le parking ci-dessus se situe à côté du centre hospitalier Sainte-Anne, d'une cité Universitaire, d'un centre commercial et d'un parc. On constate une régularité entre la semaine et le week-end. De plus on remarque une baisse petit à petit du nombre de places disponibles jusqu'au 30 mai puis une reprise d'activité se traduisant par le nombre de places disponibles inférieures à 80, les deux dernières semaines de la récolte des données. Une récolte des données plus longue nous aurait permis d'étudier la reprise sur une plus grande période et analyser une reprise des activités les week-ends à la réouverture du centre commercial et du parc.

6 Conclusion sur l'analyse

Malgré un bon nombre de graphiques générés inexploitables ou inintéressants, les données que nous avons récoltées pendant ces deux mois nous ont permis d'analyser et de caractériser le comportement de plusieurs parkings. En effet, nous avons pu constater que plusieurs parkings présentaient le même comportement durant les week-ends par exemple contrairement à d'autres parkings qui présentaient un comportement différent. Ces comportements sont influencés par l'emplacement des parkings, les évènements organisés à proximité et la période à laquelle les données sont observées.

7 Conclusion sur nos résultats

Pour conclure, l'étude montre une certaine régularité malgré la pandémie. L'influence du confinement entre le 28 mai 2021 au 30 juin 2021 puis le couvre-feu reste tout de même visible. Il est tout de même intéressant de voir le taux d'occupations des parkings parisiens durant cet événement exceptionnel.

Nos analyses se sont portés sur la différence du taux d'occupation pendant le week-end et en semaine, de même en semaine et pour des parkings à proximité nous a permis d'en déduire une régularité du nombre de places disponibles.

Par ailleurs certains parkings ont été jugés inutilisables, car il y avait des anomalies sur plusieurs jours de suite. La réalisation de ce projet nous a donné l'opportunité d'apprendre ou d'approfondir un nouveau langage de programmation "R" et de nous familiariser avec la librairie Panda de python. Ce projet nous a permis de réaliser des scripts de collecte et de nettoyage de données pour ensuite générer des graphiques nous permettant d'analyser et de caractériser ces données.

8 Bibliographie

— Mega.io : https://mega.io/
— Time series R : https://moodle.uvsq.fr/moodle2021/content/1/time_series_R.pdf
— Cours R : https://moodle.uvsq.fr/moodle2021/content/1/cours1_R_serie_temp.pdf
— Python panda : https://pandas.pydata.org/
— Python mega.py : https://pypi.org/project/mega.py/
— Python requests : https://pypi.org/project/requests/
— Python sqlalchemy : https://pypi.org/project/SQLAlchemy/
— Scripts Parking en python https://github.com/tedr5/parkings-script
— Executer un script 24/24, 7/7 sur AWS Ubuntu Serveur : https://www.youtube.com/watch?v=BYvKv3kM9pk
— Executer un script par tâche CRON sur Google Cloud : https://www.youtube.com/watch?v=5OL7fu2R4M8