WE Challenge : Mobilité en temps de Covid-19

- Maxime Allakere Hormo
- Lucas Billaud
- Siva Chanemougam
- Claire Mazzucato

Introduction

Dans le cadre de ce "WE Challenge", nous avons décidé d'apporter un élément de clarification à la mairie de Paris. Nous allons nous focaliser sur l'activité des cyclistes en période de covid, afin de mettre en place un certain nombre de mesures en leur faveur.

La problèmatique que nous allons traiter est la suivante : **La Covid a-t-elle eu un impact sur la façon de se déplacer des parisiens ?**

I) Description des Data sets utilisés

Afin d'avoir une overview globale et apporter une solution pertinente, nous avons décidé de mener une étude aussi bien **offline** qu'**online**. Nous avons utilisé cinq data sets :

- Covid-19
- Google trend
- Comptage Velo paris
- l'usage du métro à La Défense
- Velib

Le dataset sur la **Covid**, nous a permis de modéliser l'évolution de l'épidémie et de poser un contexte à notre étude. Les données que nous avons exploitées dans ce data set sont les suivantes :

- Date : Correspond au chaque jour du mois
- Décès_nb : Correspondant au nombre de décès par jour
- Décès : Correspondant au nombre de décès cumulés
- reanimation_nb : Correspond au nombre de personnes en soins intensifs par jour
- reanimation : Correspond au nombre cumulé de personnes en soins intensifs
- hospitalises_nb : nombre d'hospitalisés chaque jour
- hospitalises : nombre cumulé de personnes hospitalisées

Concernant **Google trends** nous avons pu avoir une idée des recherches online menées par les parisiens au cours de cette période pandémique. Les données que nous avons exploitées sont :

- Date : Correspond à la date hebdomadaire
- Variable: Correspond au mots-clés recherchés par les internautes
- Valeur : Correspond à la tendance des requêtes

Le 3ème Data Set, **Comptage Vélo**, nous apporte des renseignements très utiles concernant l'activité cycliste des parisiens :

- date : Correspond au dates journalières par mois
- Localisation : Lieu des bornes de comptage
- Coordonée géo : Coordonnée (latitude/ longitude)
- Comptage : Nombre de passages

Les dernières données que nous avons éxploitées, sont sur la fréquentation des **ligne de métro** autour de La défense:

- La date : date
- Trafic : Correspond au nombre de gens qui prennent le métro

II) La démarche méthodologique appliquée

Notre analyse porte sur la façon de se déplacer des parisiens en période de covid. Afin d'avoir un aperçu, nous avons préalablement recherché la tendance des mots en lien avec le vélo utilisés sur google trend.

Nous avons récolté des données issues de différentes sources pour essayer d'appréhender les impacts de la Covid 19 sur les déplacements (métro et vélo) à Paris.

Concernant les données open source, un important nettoyage a été effectué afin de les exploiter. Nous les avons épurées et aggrégées, pour ensuite pouvoir les analyser.

Après avoir identifié les principaux indicateurs, nous avons procédé à la création de notre dashboard sur **Google Data Studio**, afin de mieux suivre ces tendances et faire des recommendations à la mairie de Paris.

III) Exploration de données

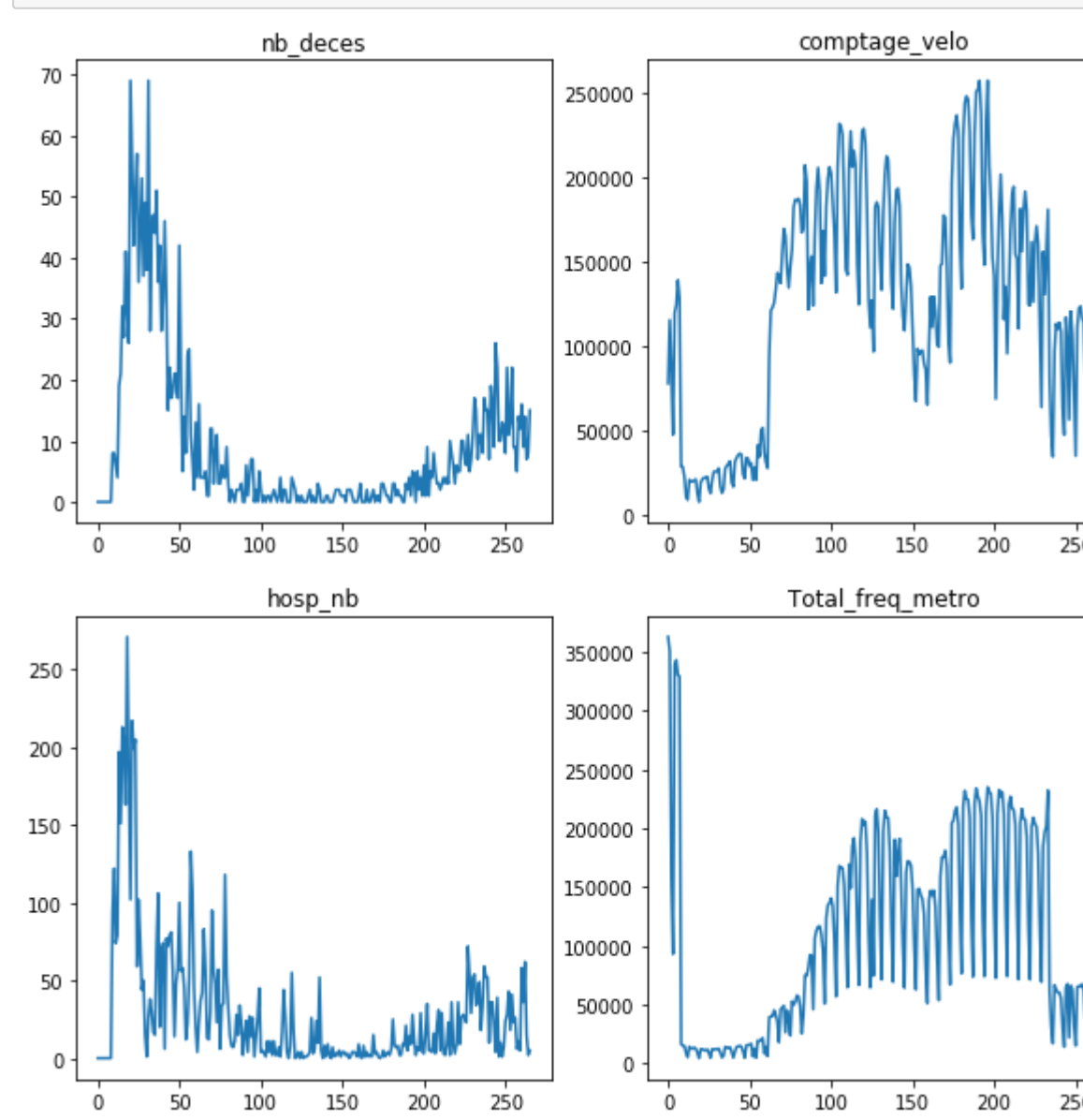
In [122]:
```
import pandas as pd
pd.set_option("display.max_rows",None)
pd.set_option("display.max_columns",None)
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [8]:
```
#import tous les fichiers
macro_velo_2020 = pd.read_csv("macro_2020.csv", sep=',')
macro_velo_2019 = pd.read_csv("macro_2019.csv", sep=',')
covid_paris = pd.read_csv("covid.csv", sep=',')
metro_defence = pd.read_csv("metro_1.csv", sep=',')
```

In [31]:
```
covid_mobilite = covid_paris.merge(macro_velo_2020, left_on="date",right_on ="Date_comptage" ,how="lef
t" )
covid_mobilite = covid_mobilite.merge(metro_defence, left_on="date",right_on ="Date" ,how="inner" )
covid_mobilite  = covid_mobilite .drop(columns = ['Date_comptage','Unnamed: 0_y','Unnamed: 0_x','Unname
d: 0',
                                                  "Type_Jour","Unnamed: 0.1","Date"])
covid_mobilite.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 266 entries, 0 to 265
Data columns (total 17 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   date                        266 non-null    object
 1   maille_nom                  266 non-null    object
 2   deces                       266 non-null    float64
 3   reanimation                 266 non-null    float64
 4   hospitalises                266 non-null    float64
 5   nouvelles_hospitalisations  266 non-null    float64
 6   nouvelles_reanimations      266 non-null    float64
 7   gueris                      266 non-null    float64
 8   evolution_deces             266 non-null    float64
 9   evolution_rea               266 non-null    float64
 10  evolution_hospitalises      266 non-null    float64
 11  deces_nb                    266 non-null    float64
 12  rea_nb                      266 non-null    float64
 13  hosp_nb                     266 non-null    float64
 14  comptage                    266 non-null    float64
 15  Total                       266 non-null    int64
 16  period                      109 non-null    float64
dtypes: float64(14), int64(1), object(2)
memory usage: 37.4+ KB
```

In [120]:
```
fig, axs = plt.subplots(2, 2,figsize=(10,10))
axs[0, 0].plot(covid_mobilite["deces_nb"])
axs[0, 0].set_title('nb_deces')
axs[1, 0].plot(abs(covid_mobilite["hosp_nb"]))
axs[1, 0].set_title('hosp_nb')
axs[0, 1].plot(covid_mobilite["comptage"])
axs[0, 1].set_title('comptage_velo')
axs[1, 1].plot(covid_mobilite["Total"])
axs[1, 1].set_title('Total_freq_metro')

plt.show()
```



On a décidé de comparer les variables sur les hospitalisations, décès, fréquentations du métro et passages de vélo. On observe qu'après le confinement l'usage du vélo a explosé, le trafic du métro a certes repris mais reste en dessous de la période pré-covid. Notre but est de définir si l'augmentation de l'usage du vélo prime sur celle du métro.

In [97]:
```
covid_mobilite[["deces_nb","hosp_nb","comptage","Total"]].describe()
```

Out[97]:

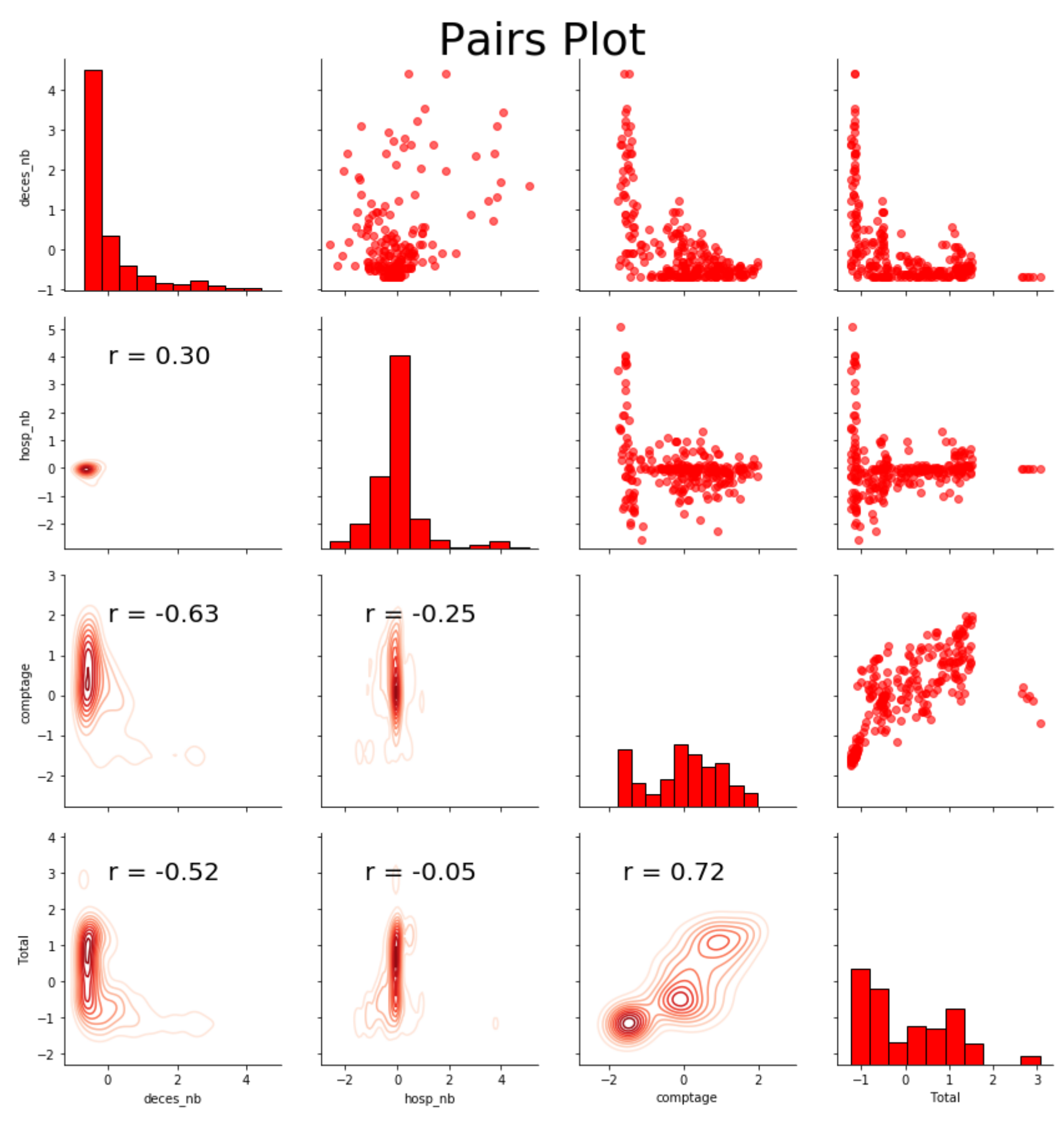
	deces_nb	hosp_nb	comptage	Total
count	266.000000	266.000000	266.000000	266.000000
mean	9.285714	2.447368	125155.154135	108074.507519
std	13.524302	52.698422	66746.801430	82826.451603
min	-0.000000	-133.000000	8108.000000	4359.000000
25%	1.000000	-15.000000	74037.750000	33822.750000
50%	4.000000	0.000000	128776.500000	90418.000000
75%	11.000000	8.000000	179481.250000	179320.000000
max	69.000000	271.000000	256796.000000	362614.000000

En observant les statistiques de chaque variable, on a décidé de les standardiser afin de pouvoir les comparer.

In [149]:
```
# Function to calculate correlation coefficient between two columns
def corr_func(x, y, **kwargs):
    r = np.corrcoef(x, y)[0][1]
    ax = plt.gca()
    ax.annotate("r = {:.2f}".format(r),
                xy=(.2, .8), xycoords=ax.transAxes,
                size = 20)

# Create the pairgrid object
grid = sns.PairGrid(data = temp, size = 3)

# Upper is a scatter plot
grid.map_upper(plt.scatter, color = 'red', alpha = 0.6)

# Diagonal is a histogram
grid.map_diag(plt.hist, color = 'red', edgecolor = 'black')

# Bottom is correlation and density plot
grid.map_lower(corr_func);
grid.map_lower(sns.kdeplot, cmap = plt.cm.Reds)

# Title for entire plot
plt.suptitle('Pairs Plot', size = 36, y = 1.02);

col = covid_mobilite[["deces_nb","hosp_nb","comptage","Total"]]
temp = col.sub(col.mean())
temp = temp.div(temp.std())
display(temp.corr())
```

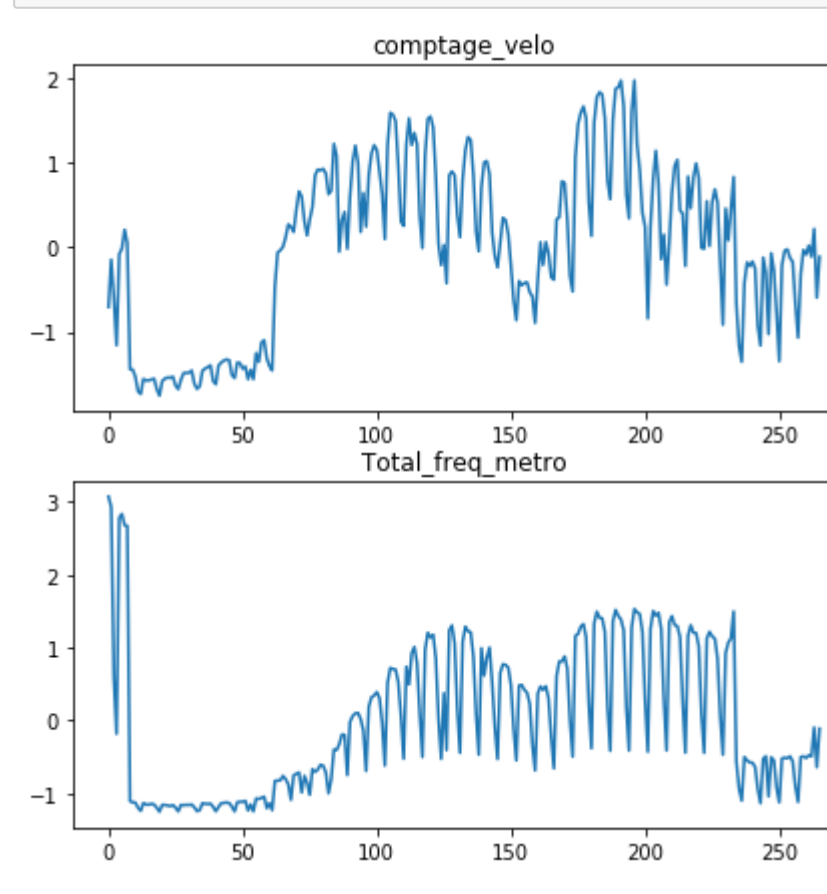
```
C:\Users\lucas\Anaconda3\lib\site-packages\seaborn\axisgrid.py:1241: UserWarning: The `size` paramter
has been renamed to `height`; please update your code.
  warnings.warn(UserWarning(msg))
```

	deces_nb	hosp_nb	comptage	Total
deces_nb	1.000000	0.299585	-0.630525	-0.522409
hosp_nb	0.299585	1.000000	-0.253799	-0.049874
comptage	-0.630525	-0.253799	1.000000	0.720160
Total	-0.522409	-0.049874	0.720160	1.000000



Le but est de trouver les relations entres les différentes variables. On observe que le nombre de décès est corrélé négativement avec l'utilisation du transport (vélo et métro). Ce phénomène peut s'expliquer par les différentes initiatives du gouvernement et de la peur face au covid des Parisiens. A partir de là, on veut identifier si à la suite d'un déconfinement les parisiens se tournent plus vers le vélo ou le métro.

In [161]:
```
fig, axs = plt.subplots(2, 1,figsize=(7,7))
axs[0].plot(temp["comptage"])
axs[0].set_title('comptage_velo')
axs[1].plot(temp["Total"])
axs[1].set_title('Total_freq_metro')
plt.show()

macro_velo_2020["Date_comptage"] = pd.to_datetime(macro_velo_2020["Date_comptage"])
velo_2020 = macro_velo_2020.loc[macro_velo_2020["Date_comptage"].dt.year == 2020]
velo_2019 = macro_velo_2020.loc[macro_velo_2020["Date_comptage"].dt.year == 2019]
evo =(sum(velo_2020["comptage"]) - sum(velo_2019["comptage"]))/sum(velo_2019["comptage"])
print(str(evo*100))
```




```
128.85513298915393
```

Nous remarquons que les parisiens empruntent toujours le métro. En revanche, on note une évolution de l'utilisation du vélo qui est plus importante que celle du métro. Plus précisement, en sortie du confinement, on note une augmentation de 5% de l'usage du métro contre une augmentation de 14% pour le vélo. D'autant plus, on observe que l'usage du vélo a doublé entre 2019(no covid) et 2020(covid).

IV) Pourquoi notre solution est unique ?

Nous avons eu une approche à 360°, en exploitant des données online grâce à Google trends pour comprendre les tendances de recherche des parisiens. Puis nous avons exploré des données issues de notre monde physique afin d'identifier et observer des faits ! A partir de là, nous avons établis des indicateurs et conçu un dashboard, pour piloter de façon dynamique l'optimsation des espaces "cyclable" de la ville.

Il s'agit d'une contribution qui est unique, car il n'y à pas de dashboard public qui fait le lien entre la mobilité des parisiens et l'épidémie du Covid-19 et qui soit à la disposition de la mairie de Paris.

Notre dashboard comprend les informations suivantes :

- Une carte indiquant les zones les plus fréquentées par les cyclistes
- l'évolution de l'usage du vélo et du métro en période de covid-19
- L'évolution de l'usage du vélo par apport à l'année précédente
- Les tendances de recherches des parisiens concernant la mobilité urbaine
- l'évolution du nombre d'hospitalisations lier au coronavirus
- Des indicateurs clés : *Hospitalisation Covid, Total des comptages Velo à Paris, tendance moyenne des mot clés Vélo*

V) Quel sera le principal impact de votre solution

Nous apportons un nouvel outil permettant d'analyser la situation actuelle concernant la mobilité parisienne. Dans cette continuité, cet outil permet de mettre en place un certain nombre d'actions. Par exemple :

- mettre en place des garages à vélos équipés de gel hydro dans les endroits les plus fréquentés

Nous avons écarté l'utilisation de la voiture personnelle dans notre l'étude, afin de nous concentrer sur l'utilisation du métro et des vélos, l'automobile n'étant pas une solution pertinante (hypothèse à vérifier dans une autre analyse) pour la circulation dans Paris intra-muros.

Nous savons que la Covid-19 a complètement modifié les habitudes de déplacement des Parisiens, notamment vis à vis des transports en commun par peur des contaminations. C'est donc pour cette raison que nous avons mis en lumière le nombre de cyclistes qui a connu une augmentation non négligable à la sortie du confinement.

A travers le dashboard que nous avons conçu, la mairie de Paris sera en mesure d'identifier les zones à fort trafic de vélo. Ainsi grâce à ces informations elle pourra mettre en place une série d'actions (localisées dans le temps) qui favoriseraient encore plus l'utilisation de vélos pendant cette période d'épidémie. Cela dans le but de limiter l'utilisation du métro sur les courtes distances. Le métro étant un endroit où il est plus facile d'être contaminé.

Bien evidement, d'autres indicateurs liés à l'usage des métros, l'évolution du covid-19 et les tendances de recherches permettront de poser un cadre / contexte et apporter une aide à la prise de décision.

Une analyse débouche souvent sur une autre ! Il peut y avoir une réelle valeur ajoutée dans l'analyse des données liées aux bornes de Vélib parisien. En effet, en croisant les données liées à la localisation des bornes vélib, et les zones de trafic des vélos, la mairie de Paris pourrait ajuster la disposition des bornes Velib, voir en ajouter de nouvelles dans des zones trafic important.

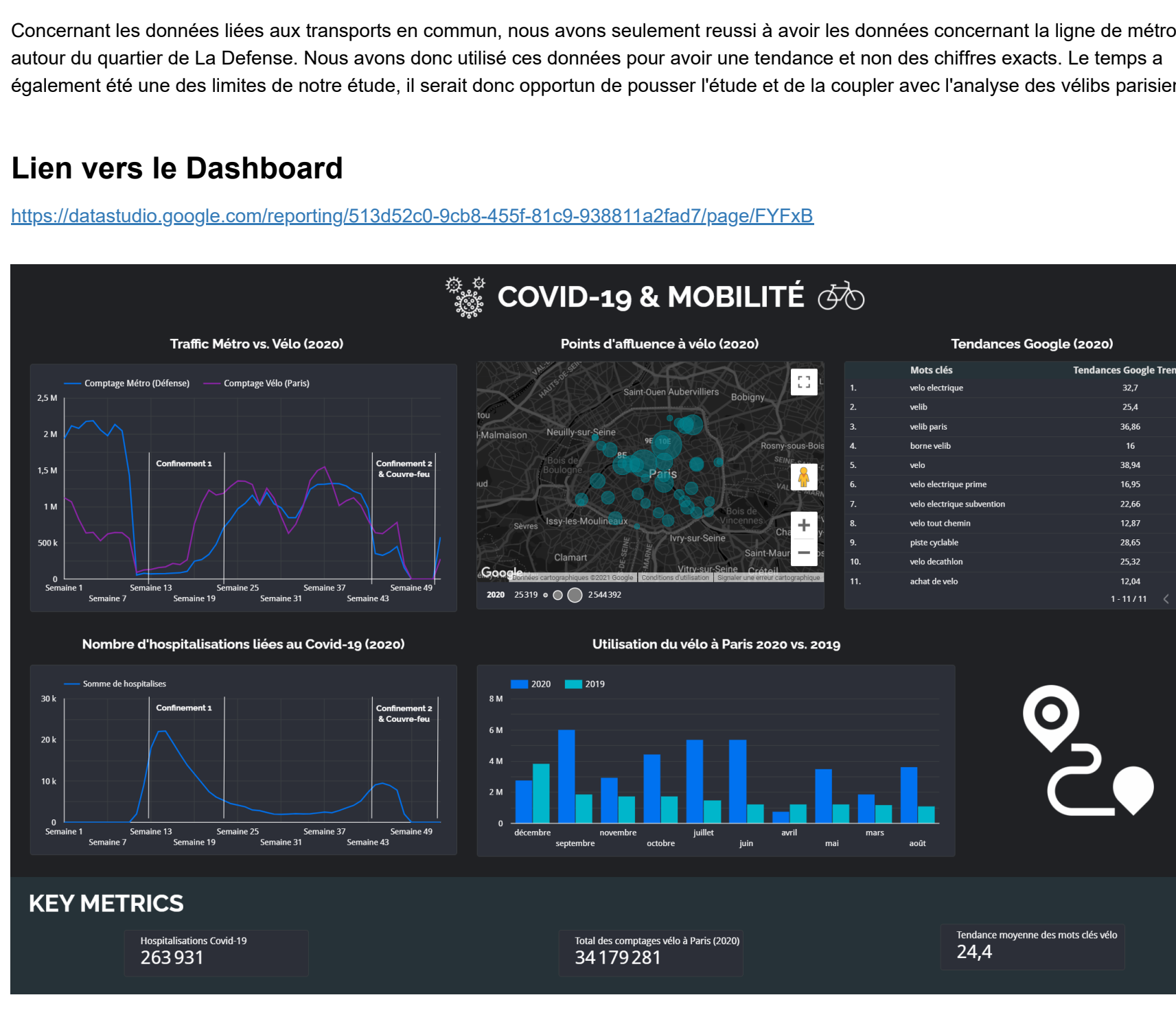
VI) Limites

Nous souhaitons tout de même souligner les limites de notre recherche. Premièrement, nous n'avons pas reussi à obtenir toutes les données souhaitées, notamment celles liées au nombre du cas de covid-19. Nous avons donc dû faire une analogie avec le nombre de décès liés au Covid afin d'identifier les périodes où la croissance de la pandémie était forte.

Concernant les données liées aux transports en commun, nous avons seulement reussi à avoir les données concernant la ligne de métro autour du quartier de La Defense. Nous avons donc utilisé ces données pour avoir une tendance et non des chiffres exacts. Le temps a également été une des limites de notre étude, il serait donc opportun de pousser l'étude et de la coupler avec l'analyse des vélibs parisien.

Lien vers le Dashboard

https://datastudio.google.com/reporting/513d52c0-9cb8-455f-81c9-938811a2fad7/page/FYFxB



source de données

- https://opendata.paris.fr/explore/dataset/comptage-velo-historique-donnees-compteurs/information/
- https://www.data.gouv.fr/fr/datasets/chiffres-cles-concernant-lepidemie-de-covid19-en-france/
- https://data.iledefrance-mobilites.fr/
- https://trends.google.fr/

Week-end Challenge - Covid et Mobilité

Annexe Data cleaning / Data preparation.

Dataset bike preparation

In [31]:
```
#packages used
import pandas as pd
pd.set_option("display.max_rows",None)
import re
import numpy as np
from pytrends.request import TrendReq
import pytrends
import io
import requests
```

In [4]:
```
df_velo_2019 = pd.read_csv("2019_comptage_velo.csv", sep = ";")
df_velo_2020 = pd.read_csv("2020_comptage_velo_old.csv", sep = ";")
```

In [5]:
```
df_velo_2019 = df_velo_2019.drop(columns = ["Identifiant du site de comptage",
                                  "Nom du site de comptage",
                                  "Date d'installation du site de comptage",
                                  "Lien vers photo du site de comptage"])

df_velo_2020 = df_velo_2020.drop(columns = ["Identifiant du site de comptage",
                                  "Nom du site de comptage",
                                  "Date d'installation du site de comptage",
                                  "Lien vers photo du site de comptage"])
```

In [7]:
```
df_velo_2019 = df_velo_2019.rename(columns = {"Identifiant du compteur":"ID_compteur",
                                              "Nom du compteur":"location_compteur",
                                              "Date et heure de comptage":"Date_comptage",
                                              "Coordonnées géographiques":"lat_long",
                                              "Comptage horaire":"comptage"})

df_velo_2020 = df_velo_2020.rename(columns = {"Identifiant du compteur":"ID_compteur",
                                              "Nom du compteur":"location_compteur",
                                              "Date et heure de comptage":"Date_comptage",
                                              "Coordonnées géographiques":"lat_long",
                                              "Comptage horaire":"comptage"})
display(df_velo_2019.head())
display(df_velo_2020.head())
```

	ID_compteur	location_compteur	comptage	Date_comptage	lat_long
0	100047539-SC	21 Boulevard Auguste Blanqui SO-NE	0	2019-01-01T07:00:00+01:00	48.830449,2.353199
1	100047539-SC	21 Boulevard Auguste Blanqui SO-NE	31	2019-01-02T18:00:00+01:00	48.830449,2.353199
2	100047539-SC	21 Boulevard Auguste Blanqui SO-NE	24	2019-01-03T16:00:00+01:00	48.830449,2.353199
3	100047539-SC	21 Boulevard Auguste Blanqui SO-NE	2	2019-01-04T04:00:00+01:00	48.830449,2.353199
4	100047539-SC	21 Boulevard Auguste Blanqui SO-NE	0	2019-01-04T08:00:00+01:00	48.830449,2.353199

	ID_compteur	location_compteur	comptage	Date_comptage	lat_long
0	NaN	97 avenue Denfert Rochereau SO-NE	2.0	2019-12-01T04:00:00+01:00	NaN
1	NaN	97 avenue Denfert Rochereau SO-NE	0.0	2019-12-01T03:00:00+01:00	NaN
2	NaN	97 avenue Denfert Rochereau SO-NE	0.0	2019-12-01T05:00:00+01:00	NaN
3	NaN	97 avenue Denfert Rochereau SO-NE	2.0	2019-12-01T08:00:00+01:00	NaN
4	NaN	97 avenue Denfert Rochereau SO-NE	0.0	2019-12-01T06:00:00+01:00	NaN

In [10]:
```
def deal_with_date(df) :

    "remove useless information in date columns"

    df = df.astype(str).str.split("T")
    year_month_day = []
    year_month_day = [date[0] for date in df]

    return year_month_day


def sub_number(df):

    """ remove number from string"""

    new_location = []

    for location in df :
        location = re.sub("\d+"," ",location)
        new_location.append(location)

    liste_location = new_location

    return liste_location


def sub_card_direction(df):

    """ remove cardinal direction
    in location"""

    new_liste = []

    for row in df :
        row = re.sub(r"(?=\S*[-])([SEON-]+)"," ",row)
        row = row.strip().lower()
        new_liste.append(row)

    return new_liste
```

In [11]:
```
#date 2019
date_velo_2019 = deal_with_date(df_velo_2019["Date_comptage"])
df_velo_2019["Date_comptage"] = date_velo_2019
df_velo_2019["Date_comptage"] = pd.to_datetime(df_velo_2019["Date_comptage"])

#date 2020
date_velo_2020 = deal_with_date(df_velo_2020["Date_comptage"])
df_velo_2020["Date_comptage"] = date_velo_2020
df_velo_2020["Date_comptage"] = pd.to_datetime(df_velo_2020["Date_comptage"])
```

In [15]:
```
#select only year 2020 within df_velo_2020
year_2020 = df_velo_2020["Date_comptage"].dt.year == 2020
df_velo_2020 = df_velo_2020.loc[year_2020]
```

In [16]:
```
#location compteur 2019
clean_loc_2019_1 = sub_number(df_velo_2019["location_compteur"])
df_velo_2019["location_compteur"] = clean_loc_2019_1
clean_loc_2019_2 = sub_card_direction(df_velo_2019["location_compteur"])
df_velo_2019["location_compteur"] = clean_loc_2019_2

#location compteur 2020
clean_loc_2020_1 = sub_number(df_velo_2020["location_compteur"])
df_velo_2020["location_compteur"] = clean_loc_2020_1
clean_loc_2020_2 = sub_card_direction(df_velo_2020["location_compteur"])
df_velo_2020["location_compteur"] = clean_loc_2020_2
```

In [19]:
```
#aggregate data "comptage" 2019
clean_df_velo_2019 = df_velo_2019.groupby(["Date_comptage","location_compteur","lat_long"])[["comptage"
]].sum()
clean_df_velo_2019 = clean_df_velo_2019.reset_index()

#aggregate data "comptage" 2020
clean_df_velo_2020 = df_velo_2020.groupby(["Date_comptage","location_compteur","lat_long"])[["comptage"
]].sum()
clean_df_velo_2020 = clean_df_velo_2020.reset_index()
```

In [21]:
```
display(clean_df_velo_2019.head())
display(clean_df_velo_2020.head())
```

	Date_comptage	location_compteur	lat_long	comptage
0	2019-01-01	avenue daumesnil	48.843435,2.383378	350
1	2019-01-01	avenue de la grande armée	48.87451,2.29215	155
2	2019-01-01	avenue de la grande armée	48.874716,2.292439	124
3	2019-01-01	avenue de la porte des ternes	48.8818100845,2.28154603106	75
4	2019-01-01	avenue denfert rochereau	48.834695,2.332968	140

	Date_comptage	location_compteur	lat_long	comptage
0	2020-01-01	avenue d'italie	48.82024,2.35902	560.0
1	2020-01-01	avenue d'italie	48.82026,2.3592	572.0
2	2020-01-01	avenue daumesnil	48.843435,2.383378	432.0
3	2020-01-01	avenue de flandre	48.889046,2.374872	977.0
4	2020-01-01	avenue de flandre	48.88926,2.37472	986.0

In [22]:
```
#combine dataset 2019 & 2020
df_final_velo = pd.concat([clean_df_velo_2019,clean_df_velo_2020])
```

In [24]:
```
#standardise coordinate
mapping = df_final_velo[["location_compteur",'lat_long']]
mapping = mapping.drop_duplicates("location_compteur")
location = list(mapping["location_compteur"])
lat_long = list(mapping["lat_long"])
dicti_to_map = {loc : lat for loc, lat in zip(location,lat_long)}
df_final_velo["lat_long"] = df_final_velo["location_compteur"].map(dicti_to_map)
```

In [27]:
```
df_final_velo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 38050 entries, 0 to 23986
Data columns (total 4 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Date_comptage      38050 non-null  datetime64[ns]
 1   location_compteur  38050 non-null  object
 2   lat_long           38050 non-null  object
 3   comptage           38050 non-null  float64
dtypes: datetime64[ns](1), float64(1), object(2)
memory usage: 1.5+ MB
```

In [28]:
```
#final group by after coordinate have been standardized
df_final_velo = df_final_velo.groupby(["Date_comptage","location_compteur","lat_long"])[["comptage"]].s
um()
df_final_velo = df_final_velo.reset_index()
```

In [67]:
```
#create global bike traffic 2019
macro_2019 = df_final_velo.groupby("Date_comptage")[["comptage"]].sum()
macro_2019 = macro_2019.reset_index()

#create global bike traffic 2019
macro_2020 = df_final_velo.groupby("Date_comptage")[["comptage"]].sum()
macro_2020 = macro_2020.reset_index()
```

In [56]:
```
#mask 1st and second lockdown
date_conf_1 = (df_final_velo["Date_comptage"] > "2020-03-16") & (df_final_velo["Date_comptage"] < "2020
-06-02")
date_conf_2 = (df_final_velo["Date_comptage"] >"2020-10-28") & (df_final_velo["Date_comptage"] < "2020-
12-31")
```

In [57]:
```
df_final_velo.loc[date_conf_1,'period_conf'] = 1
df_final_velo.loc[date_conf_2,'period_conf'] = 1
```

In [69]:
```
df_final_velo.info()

#1st data set that contains the total number of bike by date
df_final_velo.to_csv("comptage_velo_agg_2019_2020.csv")

#2st data global traffic 2019
macro_2019.to_csv("macro_2019.csv")

#3rd data global traffic 2019
macro_2020.to_csv("macro_2020.csv")
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29588 entries, 0 to 29587
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Date_comptage      29588 non-null  datetime64[ns]
 1   location_compteur  29588 non-null  object
 2   lat_long           29588 non-null  object
 3   comptage           29588 non-null  float64
 4   period_conf        7220 non-null   float64
dtypes: datetime64[ns](1), float64(2), object(2)
memory usage: 1.1+ MB
```

dataset covid preparation

In [32]:
```
df_covid = pd.read_csv("covid_paris.csv")
```

In [33]:
```
df_covid.head()
```

Out[33]:

	Unnamed: 0	date	maille_nom	deces	reanimation	hospitalises	nouvelles_hospitalisations	nouvelles_reanimations	gueris
0	0	2020-03-05	Paris	0.0	0.0	0.0	0.0	0.0	0.0
1	1	2020-03-06	Paris	0.0	0.0	0.0	0.0	0.0	0.0
2	2	2020-03-07	Paris	0.0	0.0	0.0	0.0	0.0	0.0
3	3	2020-03-08	Paris	0.0	0.0	0.0	0.0	0.0	0.0
4	4	2020-03-09	Paris	0.0	0.0	0.0	0.0	0.0	0.0

In [36]:
```
def var_day(df):

    """create a list that contains the % var"""

    liste_var = []
    series = pd.Series(df)
    liste_var.append(series.pct_change()*100)

    return liste_var


def var_abs(df_cumul, df_evo) :

    """find the number between cumulative number
    ex 14,22 => 8 """

    return (df_cumul - (df_cumul/(100 + df_evo))*100).round()
```

In [38]:
```
var_deces = var_day(df_covid["deces"])
var_rea = var_day(df_covid["reanimation"])
var_hosptitalises = var_day(df_covid["hospitalises"])
```

In [42]:
```
df_covid["evolution_deces"] = var_deces[0]
df_covid["evolution_rea"] = var_rea[0]
df_covid["evolution_hospitalises"]= var_hosptitalises[0]
```

In [43]:
```
df_covid = df_covid.replace([np.inf,-np.inf], 0)
df_covid= df_covid.fillna(0)
```

In [48]:
```
df_covid["deces_nb"] = var_abs(df_covid["deces"],df_covid["evolution_deces"])
df_covid["rea_nb"] = var_abs(df_covid["reanimation"],df_covid["evolution_rea"])
df_covid["hosp_nb"] = var_abs(df_covid["hospitalises"],df_covid["evolution_hospitalises"])
```

In [50]:
```
df_covid.head(10)

#4th dataset that contains covid data
df_covid.to_csv("covid.csv")
```

Dataset metro Paris

In [55]:
```
df_metro = pd.read_csv("frequentation_metro_defense.csv",sep=";")
df_metro.head()
```

Out[55]:

	Date	Type_Jour	Total
0	2019-03-16	SA	171004
1	2019-03-18	JOHV	366748
2	2019-03-21	JOHV	378217
3	2019-03-26	JOHV	383891
4	2019-03-29	JOHV	379865

In [60]:
```
date_conf_1 = (df_metro["Date"] > "2020-03-16") & (df_metro["Date"] < "2020-06-02")
date_conf_2 = (df_metro["Date"] >"2020-10-28") & (df_metro["Date"] < "2020-12-31")
```

In [61]:
```
df_metro.loc[date_conf_1,'period'] = 1
df_metro.loc[date_conf_2,'period'] = 1
```

In [62]:
```
#5th data set on La Defense traffic
df_metro.to_csv("metro_1.csv")
```

Dataset velib Paris

In [63]:
```
df_velib = pd.read_csv("velib.csv", sep=";")
```

In [64]:
```
df_velib_less = df_velib[["Capacité de la station","Vélos mécaniques disponibles","Vélos électriques di
sponibles",
                          "Coordonnées géographiques",
          "Nom communes équipées","Station en fonctionnement"]]

df_velib_less = df_velib_less.loc[df_velib_less["Nom communes équipées"] == "Paris"]
```

In [65]:
```
#6th data set on velib
df_velib_less.to_csv("velib_clean.csv")
```

Dataset google trend

In [ ]:
```
searches = ["velo electrique","velib","velib paris","borne velib","velo","velo electrique prime",
"velo electrique subvention","velo tout chemin","piste cyclable","velo decathlon","achat de velo"]
```

In [ ]:
```
def search_gtrend_df(l) :

    """use a list to search words on G trends
    and return a df with date, search and the trend score"""

    pytrend = TrendReq()

    dicti = { }
    for trending in l :
        pytrend.build_payload([trending], timeframe = '2018-12-31 2020-12-31', geo = "FR-J")
        dicti[trending] = pytrend.interest_over_time()

    df = pd.concat(dicti,axis=1)
    df.columns = df.columns.droplevel(0)
    df = df.drop('isPartial', axis = 1)
    df = df.reset_index()
    df = pd.melt(df, id_vars='date', value_vars=l)

    return df
```

In [ ]:
```
df_gtrends = search_gtrend_df(searches)
df_gtrends.info()

df_gtrends.to_csv("google_trend.csv",index=False)
```

In [ ]:

In [ ]:

In [ ]: