

Mobilité et TIC - Master TMEC

Hadrien Commenges, Julie Fen-Chong

03/11/2014

Contents

Introduction	1
Mobilité à New York avec les données Citibike	2
Mobilité à Marseille avec les données Twitter	6
Fonctions utiles pour la cartographie	10

Introduction

Ce document présente quelques traitements pour le module Mobilité et TIC du master Transport Mobilité Environnement Climat (TMEC) de l'Université de Bourgogne (cf. Présentation pdf). Pour progresser dans l'utilisation du logiciel R, télécharger et/ou acheter le manuel du Groupe ElementR, *R et Espace. Traitement de l'information géographique* : <http://framabook.org/16-r-et-espace>.

Deux exercices sont proposés :

1. Mobilité quotidienne à New York d'après les données Citibike.
2. Mobilité quotidienne à Marseille d'après les données Twitter.

Dans les deux cas, les données disponibles pour l'exercice sont les suivantes:

1. Données spécifiques (Citibike ou Twitter, fichier .csv)
2. Navettes domicile-travail (fichier .csv)
3. Fond de carte de l'espace d'étude (fichier .shp)

Les deux exemples sont traités successivement, New York-Citibike puis Marseille-Twitter. Deux *packages* sont utilisés pour la manipulation des données, trois *packages* sont utilisés pour le traitement des données spatiales.

```
library(sp)          # gestion des formats spatiaux
library(rgdal)        # manipulation des formats spatiaux
library(OpenStreetMap) # manipulation des fonds OpenStreetMap
library(reshape2)      # transposition des matrices
library(dplyr)         # manipulation des données
```

Mobilité à New York avec les données Citibike

Présentation des données

On commence par charger toutes les données nécessaires. Le dossier DATA contient les fichiers bruts (format .csv pour les tableaux et .shp pour les données spatiales) mais aussi un fichier .RData qui rassemble toutes les données nécessaires et qui peut être chargé directement dans R. Les données sont les suivantes :

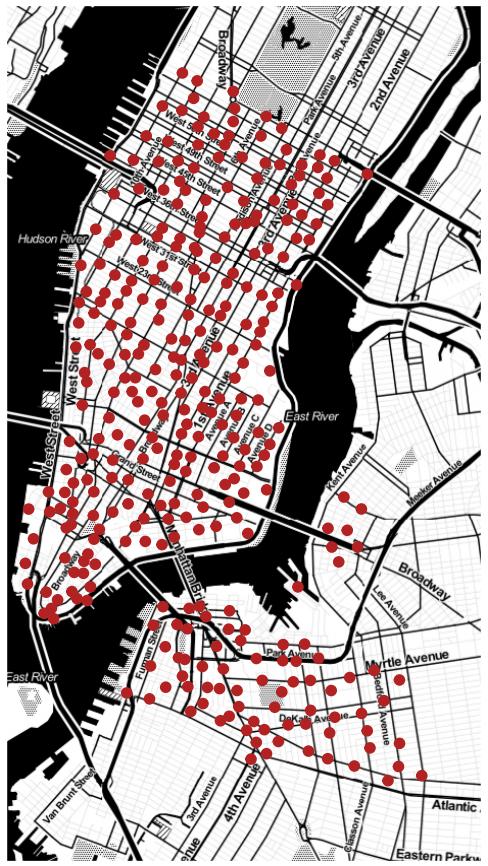
- tous les trajets réalisés sur le réseau Citibike durant le mois de mai 2014 (`data.frame`). Données téléchargeables sur le site <http://www.citibikenyc.com/system-data>
- les navettes domicile-travail produites par le recensement de 2010 entre les *counties* de l'aire métropolitaine new-yorkaise (CSA : *consolidated statistical area*) (`data.frame`). Données téléchargeables sur le site <http://www.census.gov/population/metro/data/other.html>
- les stations du réseau Citibike (`SpatialPoints`). Position extraite à partir du tableau téléchargé sur le site (variables long-lat).
- les *counties* de l'aire métropolitaine (`SpatialPolygons`). Données téléchargeables sur le site https://www.census.gov/geo/maps-data/data/cbf/cbf_counties.html.
- deux fonds extraits d'OpenStreetMap (OSM), correspondant à Manhattan et à l'aire métropolitaine (`OpenStreetMap`).

Toutes les données spatiales ont été transformées dans le système de projection NAD_1983_StatePlane_New_York_Long_Island code EPSG 102718.

```
load("DATA/NYC/DataNewYork.RData")
```

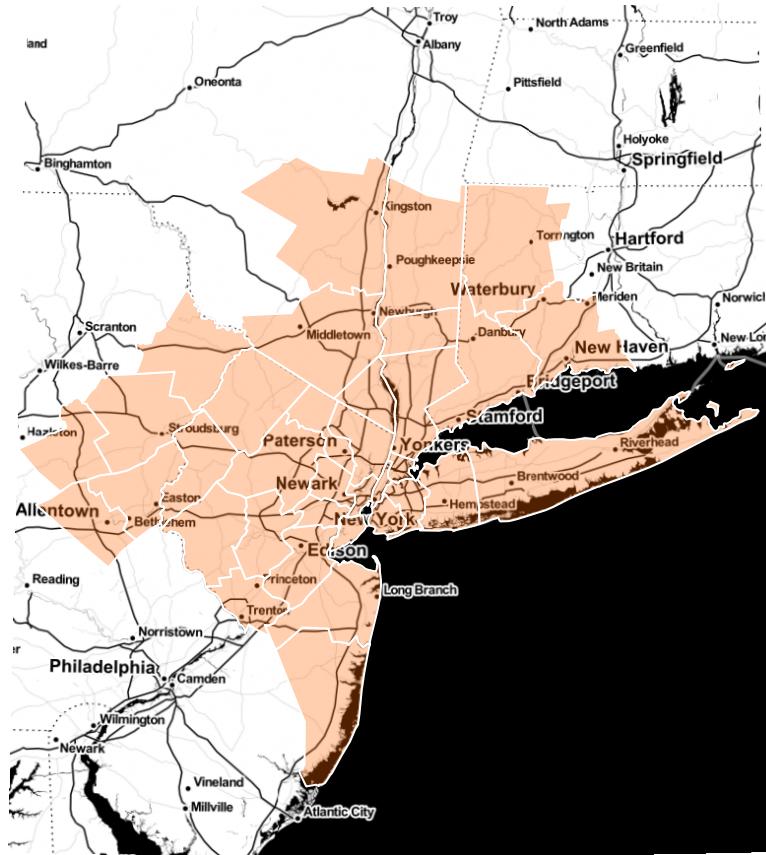
On peut cartographier les données spatiales, d'abord les stations sur le fond OSM (la fonction `plot()` ne fonctionnera que si les *packages* `OpenStreetMap` et `sp` sont installés et chargés) :

```
plot(mapManhattan)
plot(spStations, add = TRUE, pch = 20, col = "firebrick")
```



Puis les comtés sur le fond OSM correspondant :

```
plot(mapCsa)
plot(spCounties, add = TRUE, col = "#FF660050", border = "white")
```



Exploration des données de navettes domicile-travail

Le tableau `commutNycsa` comporte 1022 lignes, correspondant à 1022 flux entre comtés de la CSA de New York. Pour chaque flux est indiqué le code et le nom des comtés d'origine et de destination ainsi que le nombre de navetteurs (`flow`). Ce tableau est issu d'une sélection sur l'ensemble du pays des flux qui ont pour origine **et** pour destination l'un des 35 comtés de la CSA à l'étude. Il s'agit donc d'un tableau "long" où chaque ligne représente un couple de lieux.

Explorer la variable de flux avec `summary()`, `hist()`, etc. :

```
summary(commutNycsa$flow)
```

```
##      Min.    1st Qu.     Median      Mean    3rd Qu.      Max.
##      2.0     60.2     327.5   10210.0   2125.0  696100.0
```

Calculer les totaux marginaux, c.a.d. la somme des flux à l'origine et la somme des flux à destination. Sur un tableau long, il faut faire un résumé numérique de la variable `flow` selon une variable d'agrégation (origine ou destination). Ceci peut être fait avec la fonction `aggregate()` incluse dans la base R, avec la fonction `sqldf()` (du package du même nom) pour qui préfère s'exprimer en SQL, ou avec les fonctions du package `dplyr`. C'est cette dernière solution qui est adoptée ici :

```

totOri <- commutNycsa %>%
  group_by(ori.id) %>%
  summarize(totori = sum(flow))

totDes <- commutNycsa %>%
  group_by(des.id) %>%
  summarize(totdes = sum(flow))

```

Explorer ces deux variables (`totori` et `totdes`). Laquelle des deux variables semble la plus concentrée (concentré = peu d'unités spatiales concentrent la plus grande partie des flux) ? Comment interpréter cette concentration ?

```
summary(totOri$totori)
```

```

##      Min. 1st Qu. Median   Mean 3rd Qu.   Max.
##    21300 107000 222600 298100 400400 1053000

```

```
summary(totDes$totdes)
```

```

##      Min. 1st Qu. Median   Mean 3rd Qu.   Max.
##    11300  84470 165500 298100 362100 2291000

```

Faire le même traitement mais en transformant le tableau (liste de couple de lieux) en une matrice origine-destination (matrice carrée de 35 * 35). Dans ce cas les totaux marginaux sont calculés avec la fonction `apply()` qui applique des sommes sur les lignes (`MARGIN = 1`) ou sur les colonnes (`MARGIN = 2`) :

```

oriDesMat <- dcast(data = commutNycsa, formula = ori.id ~ des.id, value.var = "flow", fill = 0)
row.names(oriDesMat) <- oriDesMat[, 1]
oriDesMat <- oriDesMat[, -1]
totOriBis <- apply(oriDesMat, MARGIN = 1, FUN = sum)
totDesBis <- apply(oriDesMat, MARGIN = 2, FUN = sum)

```

Faire une jointure entre les origines et les destinations et calculer des soldes absolu et relatifs. Quels sont les comtés les plus attractifs, les moins attractifs ? Comment l'interpréter ?

```

totFlows <- merge(totOri, totDes, by.x = "ori.id", by.y = "des.id")
totFlows$total <- totDes$totdes + totFlows$totori
totFlows$solde.abs <- totDes$totdes - totFlows$totori
totFlows$solde.rel <- round(100 * (totFlows$solde.abs / totFlows$total), digits = 2)

```

Idées de traitements à développer (liste purement indicative, à prendre entièrement, partiellement ou à laisser) :

- Calculer d'autres indicateurs, par exemple :
 - Autocontention : pourcentage de résidents qui travaillent dans leur commune de résidence.
 - Autosuffisance : pourcentage d'emplois occupés par des résidents de la commune.
- Cartographier les indicateurs, dans R ou en faisant un import et export dans un logiciel de SIG.
- S'intéresser à la structure des flux et non plus seulement aux stocks à l'origine et à destination.

Exploration des données Citibike

Le tableau est composé de 641 329 lignes et 12 colonnes. Chaque ligne représente un trajet. Ce tableau est une sélection, à partir de l'ensemble des trajets du mois de mai 2014, des trajets réalisés en jour ouvrable (lundi-vendredi).

Commencer par explorer les variables quantitatives avec les fonctions `summary()`, `hist()`, etc. Explorer ensuite les variables qualitatives avec la fonction `table()`.

Idées de traitements à développer (liste purement indicative, à prendre entièrement, partiellement ou à laisser) :

- Choisir une ou plusieurs focales : les trajets, les individus, les stations, les flux, les vélos.
- Créer une matrice de flux de station à station.
- Créer une matrice moyenne de flux de station à station.
- Créer une matrice de distance entre les stations.
- Mettre en relation le temps et la distance.

Éléments à discuter dans la présentation et dans le dossier (à prendre en compte absolument) :

- Trouver un lien interprétatif, même un lien ténu, entre l'exploitation des navettes domicile-travail et l'exploitation des données Citibike.
- Discuter de l'utilité et des limites des deux sources de données.
- Bien calibrer le discours : on voudrait parler de la mobilité quotidienne dans son ensemble, mais on ne dispose que de données spécifiques, qui ne montrent qu'un pan de ce phénomène.

Mobilité à Marseille avec les données Twitter

Présentation des données

On commence par charger toutes les données nécessaires. Le dossier `DATA` contient les fichiers bruts (format `.csv` pour les tableaux et `.shp` pour les données spatiales) mais aussi un fichier `.RData` qui rassemble toutes les données nécessaires et qui peut être chargé directement dans R. Les données sont les suivantes :

- tous les tweets géolocalisés envoyés depuis le périmètre à l'étude au cours des mois de juin-juillet 2014 (`data.frame`). Ce type de données peut être extrait de l'API Twitter, disponible à l'adresse <https://dev.twitter.com>
- les navettes domicile-travail produites par le recensement de 2010 entre les communes (arrondissements pour la commune de Marseille) de l'aire urbaine d'Aix-Marseille-Arles (`data.frame`). Il ne s'agit pas de l'aire urbaine "définition Insee" : l'espace d'étude dans cet exercice comprend une bonne partie de l'aire urbaine d'Aix-Marseille, plus d'autres aires plus petites (Arles ou Salons par exemple) plus quelques communes multipolarisées limitrophes. Cet délimitation n'existe pas en dehors de cet exercice, elle délimite un espace qui sera désigné par le terme "aire urbaine Aix-Marseille-Arles". Données téléchargeables sur le site <http://www.insee.fr/fr/bases-de-donnees>
- les points d'envoi des tweets (`SpatialPoints`). Position extraite à partir du tableau téléchargé sur l'API Twitter (variables long-lat).
- les communes de l'aire urbaine d'Aix-Marseille-Arles (`SpatialPolygons`). Données téléchargeables sur le site <http://professionnels.ign.fr/geofla>.

- deux fonds extraits d'OpenStreetMap (OSM), correspondant à Manhattan et à l'aire métropolitaine ([OpenStreetMap](#)).

Toutes les données spatiales ont été transformées dans le système de projection RGF93 / Lambert93, code EPSG 2154.

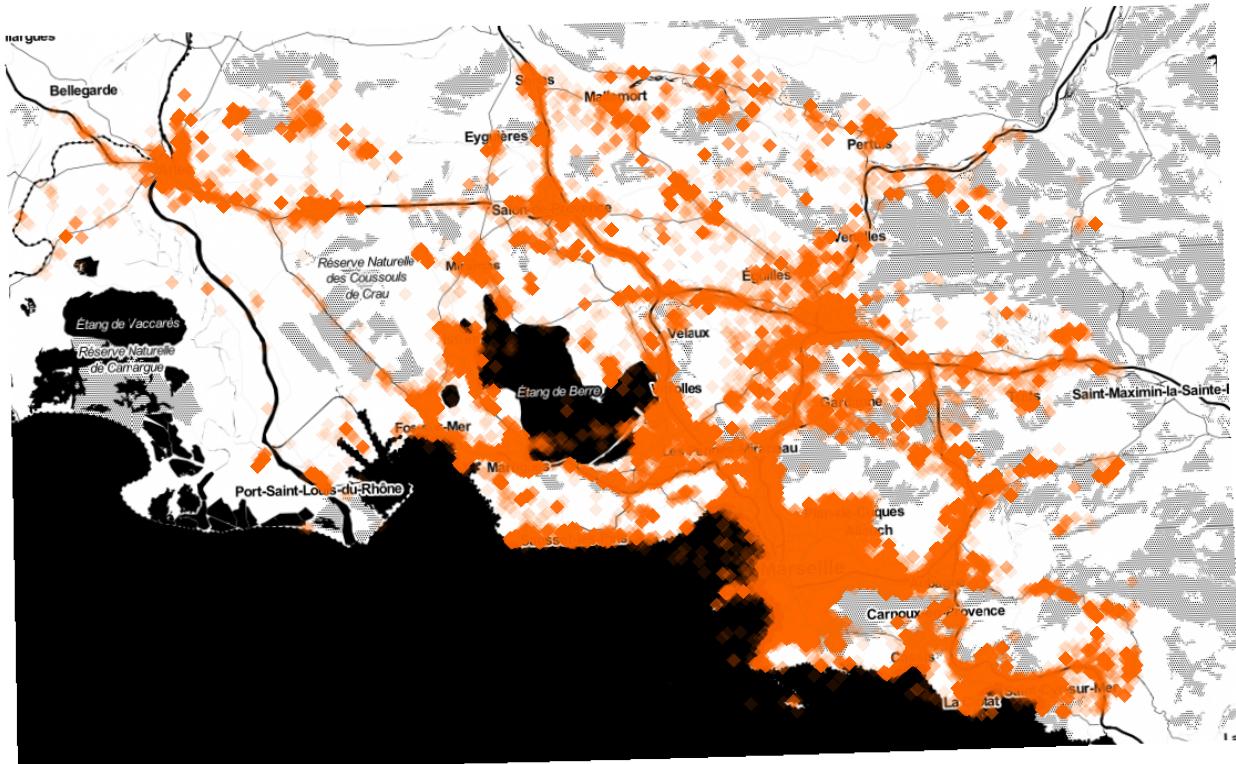
Attention, les tweets sont l'exemple même de données qu'on peut recycler dans une analyse de la mobilité mais avec prudence. **Ne jamais oublier que l'essence du tweet est le message et non la localisation :**

```
Ptofodrrr une pote a moi elle est enceinte plus jamais jlui parle a cte pute
Je chie sur les assureurs
Je vais voir l'homme de ma vie Papa
Je donne limite ma main à couper qu'on aura un sujet Histoire sur les mémoires de la 2GM
En faite hier ils ont passé un seul épisode de The Good Wife
J'ai eu 16 en français grâce a l'OM
Mon bébé me manque
Mon fils tu regale
```

```
load("DATA/Marseille/DataMarseille.RData")
```

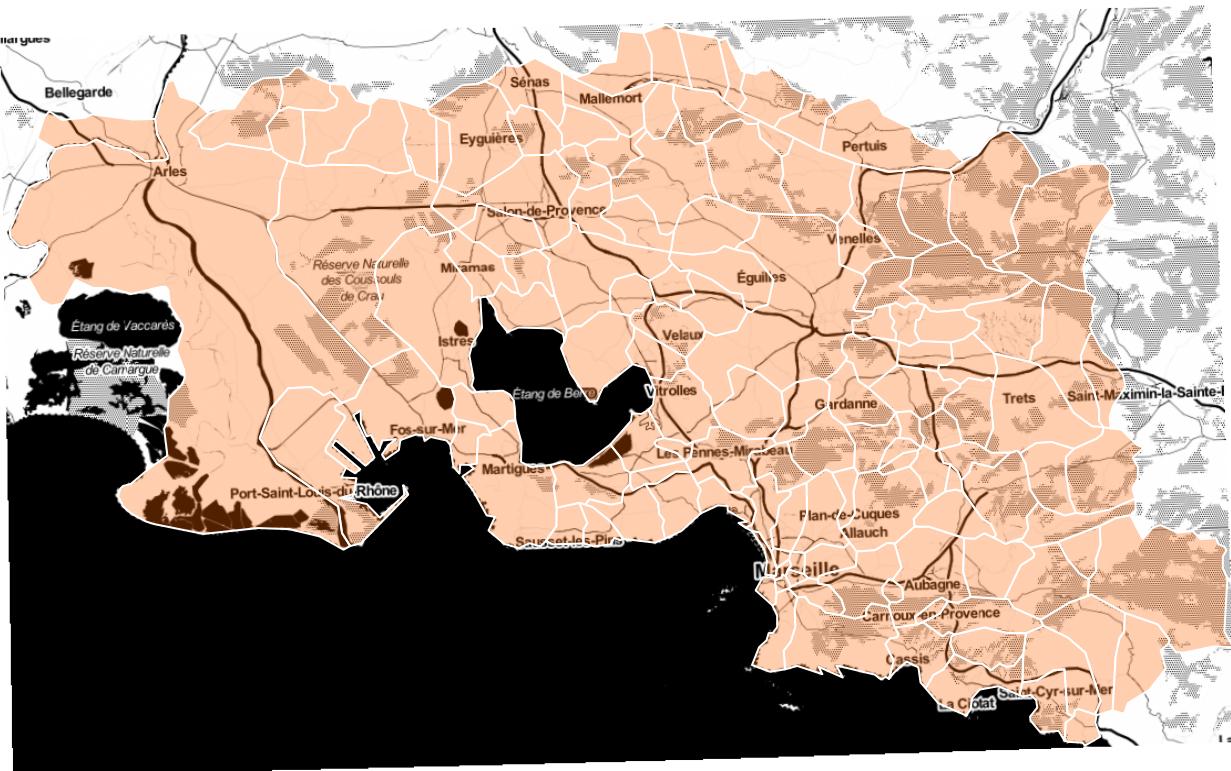
On peut cartographier les données spatiales, d'abord les tweets géolocalisés sur le fond OSM (la fonction `plot()` ne fonctionnera que si les *packages* `OpenStreetMap` et `sp` sont installés et chargés) :

```
plot(mapMars)
plot(spTweets, add = TRUE, pch = 18, col = "#FF660020")
```



Puis les communes (arrondissements pour Marseille) sur le fond OSM correspondant :

```
plot(mapMars)
plot(spCommunes, add = TRUE, col = "#FF660050", border = "white")
```



Exploration des données de navettes domicile-travail

Le tableau `navMarseille` comporte 6 671 lignes, correspondant à 6 671 flux entre communes de l'aire urbaine d'Aix-Marseille-Arles. Pour chaque flux est indiqué le code des communes d'origine et de destination ainsi que le nombre de navetteurs (`flow`). Ce tableau est issu d'une sélection sur l'ensemble du pays des flux qui ont pour origine **et** pour destination l'une des 133 communes de l'aire urbaine d'Aix-Marseille-Arles. Il s'agit donc d'un tableau “long” où chaque ligne représente un couple de lieux.

Explorer la variable de flux avec `summary()`, `hist()`, etc. :

```
summary(navettesMars$flow)
```

```
##      Min.    1st Qu.     Median      Mean    3rd Qu.      Max.
##      0.00     4.00    12.00   91.91   37.00 31890.00
```

Calculer les totaux marginaux, c.a.d. la somme des flux à l'origine et la somme des flux à destination. Sur un tableau long, il faut faire un résumé numérique de la variable `flow` selon une variable d'agrégation (origine ou destination). Ceci peut être fait avec la fonction `aggregate()` incluse dans la base R, avec la fonction `sqldf()` (du package du même nom) pour qui préfère s'exprimer en SQL, ou avec les fonctions du package `dplyr`. C'est cette dernière solution qui est adoptée ici :

```

totOri <- navettesMars %>%
  group_by(ori.id) %>%
  summarize(totori = sum(flow))

totDes <- navettesMars %>%
  group_by(des.id) %>%
  summarize(totdes = sum(flow))

```

Explorer ces deux variables (`totori` et `totdes`). Laquelle des deux variables semble la plus concentrée (concentré = peu d'unités spatiales concentrent la plus grande partie des flux) ? Comment interpréter cette concentration ?

```
summary(totOri$totori)
```

```

##      Min. 1st Qu. Median     Mean 3rd Qu.    Max.
##        20    1100   2131     4610   4028   45180

```

```
summary(totDes$totdes)
```

```

##      Min. 1st Qu. Median     Mean 3rd Qu.    Max.
##        4     502   1229     4610   4007   67480

```

Faire le même traitement mais en transformant le tableau (liste de couple de lieux) en une matrice origine-destination (matrice carrée de 133*133). Dans ce cas les totaux marginaux sont calculés avec la fonction `apply()` qui applique des sommes sur les lignes (`MARGIN = 1`) ou sur les colonnes (`MARGIN = 2`) :

```

oriDesMat <- dcast(data = navettesMars, formula = ori.id ~ des.id, value.var = "flow", drop = FALSE, fill = 0)
row.names(oriDesMat) <- oriDesMat[, 1]
oriDesMat <- oriDesMat[, -1]
totOriBis <- apply(oriDesMat, MARGIN = 1, FUN = sum)
totDesBis <- apply(oriDesMat, MARGIN = 2, FUN = sum)

```

Faire une jointure entre les origines et les destinations et calculer des soldes absolus et relatifs. Quels sont les comtés les plus attractifs, les moins attractifs ? Comment l'interpréter ?

```

totFlows <- merge(totOri, totDes, by.x = "ori.id", by.y = "des.id", all = TRUE)
totFlows$total <- totDes$totdes + totFlows$totori
totFlows$solde.abs <- totDes$totdes - totFlows$totori
totFlows$solde.rel <- round(100 * (totFlows$solde.abs / totFlows$total), digits = 2)

```

Idées de traitements à développer (liste purement indicative, à prendre entièrement, partiellement ou à laisser) :

- Calculer d'autres indicateurs, par exemple :
 - Autocontention : pourcentage de résidents qui travaillent dans leur commune de résidence.
 - Autosuffisance : pourcentage d'emplois occupés par des résidents de la commune.
- Cartographier les indicateurs, dans R ou en faisant un import et export dans un logiciel de SIG.
- S'intéresser à la structure des flux et non plus seulement aux stocks à l'origine et à destination.

Exploration des données Twitter

Le tableau est composé de 393 722 lignes et 7 colonnes. Chaque ligne représente un tweet. Ce tableau contient l'ensemble des tweets géolocalisés envoyés depuis le périmètre à l'étude au cours de mois de juin-juillet. Plusieurs champs ont été effacés (le contenu du message en particulier), les identifiants des tweets et des utilisateurs ont été modifiés pour garantir l'anonymat.

Idées de traitements à développer (liste purement indicative, à prendre entièrement, partiellement ou à laisser) :

- Choisir une ou plusieurs focales : les tweets, les individus, les trajets, les communes.
- Créer une variable de temps utilisable pour l'exploration temporelle des tweets et faire cette exploration.
- Dresser des profils d'individus à partir de leurs pratiques de tweet/mobilité.
- Faire une sélection d'individus, les hypermobiles par exemple.
- Faire une jointure spatiale entre tweets et communes puis créer une matrice de flux de commune à commune.
- Explorer cette matrice de la même façon que la matrice de navettes domicile-travail.
- Comparer la matrice de flux intercommunaux ainsi créée avec celle des navettes.

Éléments à discuter dans la présentation et dans le dossier (à prendre en compte absolument) :

- Trouver un lien interprétatif, même un lien ténu, entre l'exploitation des navettes domicile-travail et l'exploitation des tweets géolocalisés.
- Discuter de l'utilité et des limites des deux sources de données.
- Bien calibrer le discours : on voudrait parler de la mobilité quotidienne dans son ensemble, mais on ne dispose que de données spécifiques, qui ne montrent qu'un pan de ce phénomène.

Fonctions utiles pour la cartographie

Ces deux fonctions permettent de faire une jointure de données externes sur la table attributaire des données spatiales (`AttribJoin`) et de discréteriser automatiquement une variable quantitative (`Discretimatic`). Pour exporter l'objet spatial et le travailler dans un logiciel de SIG, utiliser la fonction `writeOGR()` en précisant le format, par exemple `driver = "ESRI Shapefile"`.

```
AttribJoint <- function(df,           # Tableau externe, data.frame
                        spdf,         # Objet spatial
                        df.field,    # Identifiant dans le tableau externe
                        spdf.field)  # Identifiant dans l'objet spatial

{
  if(is.factor(spdf@data[, spdf.field]) == TRUE) {
    spdf@data[, spdf.field] <- as.character(
      spdf@data[, spdf.field]
    )
  }
  if(is.factor(df[, df.field]) == TRUE) {
    df[, df.field] <- as.character(df[, df.field])
  }
  spdf@data <- data.frame(
    spdf@data,
```

```

df[match(spdf@data[, spdf.field], df[, df.field]), ]
)
return(spdf)
}

DiscretiMatic <- function(vec)  # variable quantitative à discréteriser
{
normTest <- shapiro.test(vec)
if(normTest$p.value > 0.1) {
  print("Discrétisation autour de la moyenne")
  valBreaks <- c(min(vec),
                 mean(vec) - sd(vec),
                 mean(vec),
                 mean(vec) + sd(vec),
                 max(vec))
  varDiscret <- cut(vec,
                     breaks = valBreaks,
                     include.lowest = TRUE,
                     right = FALSE)
} else {
  print("Discrétisation en quartiles")
  valBreaks <- quantile(vec,
                        probs = c(0, 0.25, 0.5, 0.75, 1))
  varDiscret <- cut(vec,
                     breaks = valBreaks,
                     include.lowest = TRUE,
                     right = FALSE)
}
return(varDiscret)
}

```