Introduction à la géomatique pour le statisticien : quelques concepts et outils innovants de gestion, traitement et diffusion de l'information spatiale

Documents de travail

N° 2022-01 – Février 2022

François SEMECURBE
Elise COUDIN

Insee
Mesurer pour comprendre

INSTITUT NATIONAL DE LA STATISTIQUE ET DES ÉTUDES ÉCONOMIQUES
Série des documents de travail « Méthodologie Statistique »
de la Direction de la Méthodologie et de la Coordination Statistique et Internationale

M 2022/01

Introduction à la géomatique pour le statisticien : quelques concepts et outils innovants de gestion, traitement et diffusion de l'information spatiale

François SEMECURBE
Elise COUDIN

Insee

Février 2022

Direction de la méthodologie et de la coordination statistique et internationale
Département des Méthodes Statistiques -Timbre L001 -
88 Avenue Verdier - CS 70058 - 92541 Montrouge Cedex - France -
Tél. : 33 (1) 87 69 55 00 - E-mail : -DG75-L001@insee.fr - Site Web Insee : http://www.insee.fr

Introduction à la géomatique pour le statisticien : quelques concepts et outils innovants de gestion, traitement et diffusion de l'information spatiale*

François Sémécurbe[‡] et Elise Coudin[§]

Janvier 2022

Résumé

Ce document vise une présentation simple à l'attention des statisticiens des outils géomatiques récents qui permettent de stocker, traiter et diffuser l'information spatiale. Les logiciels comme R ou Python intègrent désormais les caractéristiques géographiques rendant plus accessibles leur traitement. Pour autant, le foisonnement des technologies, les possibilités offertes par le web, les technologies web, sont autant d'obstacles à dépasser pour celles et ceux souhaitant réaliser des cartographies thématiques percutantes. Ce document propose une présentation unifiée des concepts géomatiques, avec des extraits de code en R, Python et PostGIS. Il se concentre sur les données vectorielles et décrit les traitements et manipulations classiques à connaître pour construire une statistique spatiale. Il aborde aussi les outils et les flux permettant une diffusion dynamique (cartes interactives) de l'information spatiale. Il discute enfin le rôle de la spatialisation dans la représentation des données statistiques.

Mots clés : géomatique, cartographie dynamique, cartographie thématique, statistiques spatiales, données spatiales, systèmes d'information géographique.

*Ce document s'appuie sur et complète les travaux du groupe de travail Great sur les nouveaux outils cartographiques qui s'est tenu en 2019. Les auteurs remercient Julie Djiriguian, Yves-Laurent Benichou, Sébastien Calvet, Sébastien Novella, Fabrice Garnes, David Levy, Vianney Costemalle, Sonia Oujia, Marc Branchu, Julien Jamme, Arlindo Dos Santos, Aurélien D'Isanto, Frédéric Comte, Christophe Yon, Lionel Cacheux, Olivier Levitt pour leur contribution aux travaux de ce groupe. Julie Djiriguian et François Sémécurbe ont poursuivi ces travaux sous la forme d'un site web de prise en main de ces outils, dont certaines parties de ce document s'inspirent très largement. Les auteurs remercient à ce titre une deuxième fois Julie Djiriguian. Ce document a bénéficié des relectures d'Aurélien Fortin, Clément Guillot, Benoit Hurpeau, Patrick Sillard que les auteurs remercient. Elise Coudin tient à préciser qu'elle a tenu un rôle de relectrice active plus qu'autre chose et que le mérite revient au premier auteur.

[‡] Insee et ThèMA - Besançon

[§] Insee et CREST

Introduction to geomatics for statisticians: some innovative concepts and tools for storing, processing and disseminating spatial information

Abstract

This document is intended as a simple presentation for statisticians of recent geomatics tools, which allow the storage, processing and dissemination of spatial information. Softwares like R or Python now integrate directly geographical characteristics making their processing more accessible. However, the proliferation of technologies, the possibilities offered by the web, web technologies, are all obstacles to overcome for those wishing to produce powerful thematic maps. This document offers a unified presentation of geomatics concepts, with code snippets in R, Python and PostGIS. It focuses on vector data and describes the classic treatments and manipulations to know to build a spatial statistic. It also addresses the tools and flows allowing dynamic dissemination (interactive maps) of spatial information. Finally, it discusses the role of spatialization in the representation of statistical data.
Keywords: geomatics, dynamic cartography, thematic cartography, spatial statistics, spatial data, geographic information systems.

Table des matières

Introduction **8**

1 Les données spatiales **12**
1.1 La géographie 12
1.2 Stockage des données spatiales 13
1.2.1 Fichiers cartographiques 14
1.2.2 PostGIS, une base de données spatiales 16
1.2.3 Système de projection 19
1.3 Exemple : création d'une couche carroyée à partir de données ponctuelles 21

2 Traitement des données spatiales **24**
2.1 Fonctions et transformations géométriques élémentaires 24
2.1.1 Transformation de données ponctuelles en données surfaciques 24
2.1.2 Simplification de données surfaciques ou transformation de données surfaciques en données ponctuelles 27
2.2 Transformations complexes mettant en relation plusieurs éléments d'une couche ou de plusieurs couches 29
2.2.1 Agrégation des données spatiales 29
2.2.2 Intersection et jointure des données spatiales 30
2.3 Exemples de manipulation de données spatiales 31
2.3.1 Définir l'emprise spatiale des zones bâties 31
2.3.2 Déterminer l'accessibilité à un équipement 33
2.3.3 Agréger les données carroyées à l'échelle des communes . . . 38

3 Diffusion dynamique de l'information spatiale **40**
3.1 Les flux géographiques 40
3.1.1 Les flux WMS 40
3.1.2 Les flux WFS 42
3.1.3 Exemple d'application dynamique 44
3.2 GeoServer : diffuser des flux géographiques 46
3.3 Pour aller plus loin : les tuiles vectorielles 47
3.3.1 Le principe des tuiles vectorielles 47
3.3.2 Mise en oeuvre avec Tegola 49

Discussion : le territoire des statisticiens **53**

Table des figures

1 Exemples de cartes thématiques 8
2 Répartition des terrasses éphémères parisiennes en 2021 - données ponctuelles ou surfaciques 13
3 Les trois principales projections utilisées par les statisticiens : Lambert 93, LAEA (projection européenne) et WGS 84 (utilisée par le système de positionnement par satellite GPS) 20
4 Associer une surface à un point : exemple avec les terrasses éphémères 25
5 Emprises géométriques minimales de la ville de Paris 26
6 Simplification d'un fond de carte : simplifier le contour des arrondissements parisiens 28
7 Les étapes de construction des taches bâties 33
8 Localisation des terrasses éphémères disposant de plus de 100 terrasses voisines dans un rayon de 200m 35
9 Carreaux à moins de 500m d'un parc 36
10 Contenu du flux IGN WMS 41
11 Bâtiments autour de la Direction Générale de l'Insee à Montrouge suivant différentes définitions de l'image (paramètres *width* et *height*) 42
12 Résultat de la requête *getfeature* précédente 43
13 Interface de GeoServer 47
14 Indexation des tuiles en fonction de l'échelle et de leur localisation. Source : Documentation QGIS 48
15 Application de prévisualisation des tuiles vectorielles de Tegola. . . 52
16 "Land doesn't vote, people do." 55
17 Deux visions d'un même phénomène : répartition des ménages pauvres dans l'agglomération parisienne. Source : Filosofi 2015 56
18 MAUP : effet de zonage 57
19 MAUP : effet d'échelle. 58
20 *Are larger cities greener or smoggier ?* Relation entre l'émission de CO_2 et la taille des villes aux Etats-Unis. 59

Liste des tableaux

1 Exemple de table spatiale : Position des terrasses éphémères parisiennes en 2021 13
2 Précision géographique en fonction de l'échelle d'observation. Source : Mapbox 48

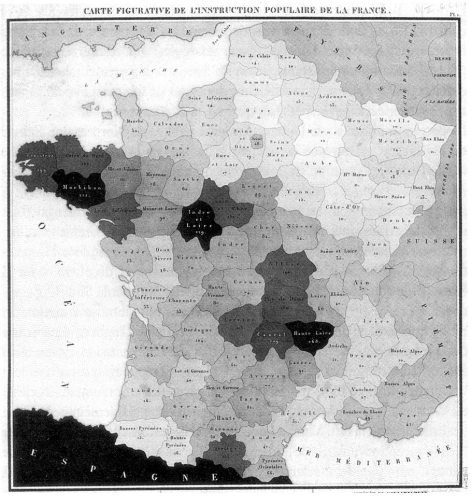
Liste des codes

1 Lire et écrire des données spatiales en R . . . 15
2 Lire et écrire des données spatiales en Python . . . 15
3 Transformer des coordonnées géographiques en données spatiales avec R . . . 15
4 Transformer des coordonnées géographiques en données spatiales avec Python . . . 15
5 Rajouter l'extension PostGIS à une base PostgreSQL depuis R . . . 16
6 Rajouter l'extension PostGIS à une base PostgreSQL depuis Python 17
7 Envoyer et récupérer une table sous PostGIS depuis R . . . 18
8 Envoyer et récupérer une table sous PostGIS depuis Python . . . 18
9 Reprojection d'un fond départemental en R . . . 21
10 Reprojection d'un fond départemental en Python . . . 21
11 Reprojection d'un fond départemental en PostGIS . . . 21
12 Transformation de données ponctuelles en données surfaciques en R 22
13 Transformation de données ponctuelles en données surfaciques en Python . . . 22
14 Transformation données ponctuelles en données surfaciques en PostGIS . . . 23
15 Diagramme de Voronoi des terrasses éphémères en R . . . 25
16 Diagramme de Voronoi des terrasses éphémères en Python . . . 25
17 Diagramme de Voronoi des terrasses éphémères en PostGIS . . . 26
18 Polygones d'emprise spatiale minimale en R . . . 26
19 Polygones d'emprise spatiale minimale en Python . . . 27
20 Polygones d'emprise spatiale minimale en PostGIS . . . 27
21 Compression du fond de carte des arrondissements parisiens à l'aide de mapshaper . . . 28
22 Agrégation d'une couche géographique en R . . . 29
23 Agrégation d'une couche géographique en Python . . . 29
24 Agrégation d'une couche géographique en SQL . . . 30
25 Intersection spatiale en R . . . 30
26 Intersection spatiale en Python . . . 30
27 Intersection spatiale en PostGIS . . . 31
28 Délimitation des taches en R . . . 32
29 Délimitation des taches en Python . . . 32
30 Délimitation des taches en PostGIS . . . 33
31 Requête spatiale reposant sur les structures **k-d tree** en R . . . 34
32 Requête spatiale reposant sur les structures **k-d tree** en Python . . . 34
33 Déterminer les carreaux à moins de 500 m d'un parc ou d'un jardin en R . . . 35

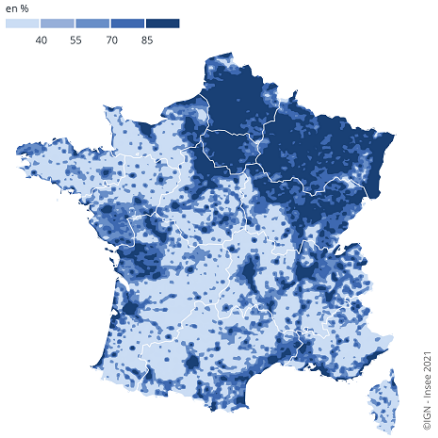
34 Déterminer les carreaux à moins de 500 m d'un parc ou d'un jardin en Python 36
35 Application de la DBSCAN à la détection des agrégats de terrasses en R 37
36 Application de la DBSCAN à la détection des agrégats de terrasses en Python 38
37 Calcul d'une population communale à partir de données carroyées au prorata de la surface commune 39
38 Requête WMS pour récupérer une image 41
39 Téléchargement d'une image à partir d'une requête WMS en R . . . 41
40 Téléchargement d'une image à partir d'une requête WMS en Python 42
41 Requête WFS pour récupérer les données vectorielles 43
42 Traitement d'une requête WFS en R 43
43 Traitement d'une requête WFS en Python 44
44 Cartographie dynamique d'un flux WFS à l'aide de la librairie OpenLayer 44

Introduction

Les **cartes thématiques**, appelées également **cartes statistiques** ou **quantitatives**, sont des médiations symboliques qui visent à restituer graphiquement l'organisation spatiale des phénomènes sociaux (Palsky, 1990). Leur origine est étroitement liée à celle de la statistique. La première carte thématique moderne est attribuée au statisticien Charles Dupin. Publiée en 1826, elle décrit la répartition de l'instruction populaire en France (fig 1a).



(a) Une première carte thématique : instruction populaire en France, Charles Dupin, 1826
Lecture : nombre d'élèves masculins scolarisés en primaire rapporté au nombre d'habitants dans le département selon dix tranches en aplat de couleur. Ce rapport est faible dans les zones les plus foncées.



(b) Part du bâti concentré en 2005, Himpens, Poulhes, et Sémécurbe (2021)
Lecture : en 2005, dans la majorité des territoires du quart nord-est de la France, plus de 85 % du bâti est concentré dans des noyaux villageois.

FIGURE 1 – Exemples de cartes thématiques

Cette carte marque une rupture avec les cartes topographiques. Pour la première fois, une représentation graphique vise à représenter les variations spatiales d'une grandeur abstraite, en l'occurrence une statistique sociale [1]. Ceci est rendu

1. La carte figurative de l'instruction populaire sépare la France en deux : dans le nord-est la France « éclairée » s'oppose à la France « obscure » dans le reste du pays selon les commentaires de Dupin lors de la présentation de sa carte au Conservatoire des Arts et Métiers (Palsky, 1996). Le message est clair et a nourri tout le long du 19^e les représentations politiques et sociales, ce

possible grâce à ce qui s'appellera à la fin du 20^e siècle, la **sémiologie cartographique**. Plus les teintes sont claires et plus la part de l'instruction populaire est importante dans les départements. Contrairement, aux cartes topographiques dont la lecture est plus ou moins immédiate, la compréhension des cartes thématiques nécessite d'acquérir certains réflexes perceptifs [2]. En contrepartie de cet effort interprétatif, la cartographie thématique produit une vision utile à l'action publique et à l'appropriation par les citoyens des enjeux sociaux. En particulier, la comparaison de cartes permet de mettre en relation des phénomènes sociaux *a priori* indépendants : tels que l'instruction populaire et la concentration de l'habitat [3] (fig 1b).

Autrefois réalisées à la main, les cartographies thématiques actuelles nécessitent l'acquisition de savoir-faire **géomatiques**. La **géomatique**, qui provient de la contraction des termes « géographie » et « informatique », désigne l'ensemble des outils, technologies et méthodes permettant de représenter, d'analyser et d'intégrer des données géographiques [4]. Elle fait appel aux sciences, aux technologies de mesure de la terre ainsi qu'aux technologies de l'information pour faciliter l'acquisition, le traitement et la diffusion de ces données. Elle couvre les bases de données spatiales, les systèmes d'information géographique (SIG) [5], l'imagerie spatiale, mais aussi les technologies du web, la géolocalisation [6]...

La profusion actuelle de l'information spatiale n'a pas encore favorisé sa démocratisation. Au contraire, le foisonnement des technologies et la nécessité d'acquérir des compétences plurielles en analyse spatiale, en gestion des réseaux et en développement web représentent des obstacles à dépasser. Si des solutions simples existent pour diffuser de l'information spatiale à petite échelle, telle que Rshiny, la diffusion massive d'une information nécessite des infrastructures et des connaissances spécifiques. C'est à ce besoin que ce document cherche à répondre. Ainsi, il ne constitue pas un guide en matière de cartographie spatiale, de sémiologie spatiale, ou encore d'analyses statistiques spatiales. Pour ce faire, le lecteur pourra se référer au très

qui fait de la carte un outil au statut particulier dont la lecture a un impact supérieur à celle d'un simple tableau. En tant qu'objet symbolique la carte peut être porteuse de présupposés voire de préjugés, ainsi le recours aux couleurs sombres peut refléter « l'obscurantisme ».

2. Palsky (1991) reprend Playfair (1805) qui écrit « the reader will find, five minutes attention to the principle on which they are constructed, a saving of much labor and time ; but without that trifling attention, he may as well look at a blank sheet of paper. »

3. voir Le Bras et Todd (2013) pour une explication à cette corrélation spatiale.

4. voir le site de l'Institut National de Géographie : `https://www.sig-geomatique.fr/sig-geomatique.html`

5. Un système d'information géographique ou SIG (en anglais, geographic information system ou GIS) est un système d'information conçu pour recueillir, stocker, traiter, analyser, gérer et présenter tous les types de données spatiales et géographiques Wikipédia (2021c).

6. Pour une revue des outils de géolocalisation disponibles sur le web ou l'Insee et d'autres exemples cartographiques voir `https://juliedjidji.github.io/memocarto/geocodage.html`

complet manuel d'analyse spatiale, Loonis et Bellefon (2018), pour les méthodes d'analyse, à Lambert et Zanin (2016), ou encore au portail de la cartographie proposé par le pôle de service de l'action régionale sur l'analyse territoriale de l'Insee. Ce document s'inscrit en complément comme une initiation, une explicitation, une présentation, à destination d'un public de statisticiens des outils et les méthodes actuels et innovants de la géomatique, en se concentrant sur le cas des données spatiales vectorielles [7], c'est-à-dire définies en contours des territoires.

Le traitement des données spatiales a été bouleversé par une prise en compte intégrée de la géographie dans les langages de programmation. Autrefois, la prise en compte de la géographie sous R reposait sur le *package* sp qui traitait séparément les données statistiques et les données spatiales (Giraud et Pecout, 2021). La manipulation simultanée de l'information statistique et spatiale était délicate et lente. La géométrie, c'est-à-dire l'information purement spatiale, est à présent une information comme une autre dans les tables de données ce qui simplifie considérablement les traitements. Cette approche a été rendue possible par le recours systématique aux librairies spatiales GEOS, GDAL et PROJ dans les logiciels Python avec le *package* geopandas (extension spatiale de la librairie de traitement des tables de données pandas), et pour R avec le *package* sf compatible avec la suite tidyverse et PostgreSQL et l'extension spatiale PostGIS. Ces solutions logicielles étant libres et basées sur les mêmes librairies garantissent l'uniformité des traitements cartographiques. Ce document essaye tant que faire se peut d'illustrer cette proximité en proposant des exemples de codes dans les trois langages R, Python et SQL.

La suite de ce document se compose de quatre parties. La première partie sur les **données spatiales** décrit la façon dont l'espace est pris en compte dans les systèmes d'information géographique. La deuxième est consacrée aux **traitements et manipulations sur les données spatiales**. Elle décrit les opérations couramment utilisées pour produire une connaissance statistique à l'aide de données géographiques et de statistiques élémentaires. La troisième s'intéresse à la **diffusion dynamique de l'information spatiale**. Elle explore les outils disponibles pour mettre à disposition du grand public des cartes thématiques interactives.
La dernière partie de ce document plus conclusive discute des mésusages de la cartographie thématique. La dimension graphique voire esthétique des cartes thématiques en fait des outils puissants pour « faire parler les chiffres » et raconter une histoire et ce, quelle que soit la qualité des analyses sous-jacentes. Il est donc toujours primordial pour les statisticiens de se questionner sur le rôle de la spatialisation et celui de la représentation des données statistiques dans la production d'une connaissance géographique.

7. Ce document n'aborde pas les images *raster* qui recourent en effet à des outils différents : reconnaissance d'images...

L'ensemble des outils présentés dans ce document peuvent être exécutés en local sur un ordinateur personnel et dans ce cas, sauf manipulation spécifique, ils sont accessibles uniquement à son utilisateur. Il est également possible de les déployer sur des serveurs *adhoc* ou à l'aide de conteneurs Docker, ceci afin de disposer d'une puissance de traitement plus importante et de mettre à disposition d'un large public l'information spatiale.

1 Les données spatiales

Les données spatiales désignent les informations liées à des objets ou éléments présentant une composante spatiale. Elles recouvrent deux types de données :

Vectoriel Ce type de données repose sur l'usage de figures géométriques telles que des points, des lignes et des polygones définis à l'aide de listes de coordonnées pour représenter les objets spatiaux, tels que la répartition des équipements ou de la population dans l'espace ou les contours des départements et des régions.

Raster Ce type utilise des images, soit des matrices de nombres, pour décrire les différents constituants de l'espace. Les images aériennes mais aussi les plans d'occupation du sol sont généralement diffusées en format raster.

Ce document se concentre sur les **données spatiales vectorielles**. Les bases de données spatiales vectorielles sont optimisées pour stocker et interroger des données reliées à des objets référencés géographiquement, comme des points, les lignes et des polygones, communément appelés **géométrie** de la base. Les données numériques et textuelles de la base spatiale sont dénommées **données attributaires**.

1.1 La géographie

Dans les logiciels R, Python et les bases de données PostGIS, l'information spatiale une fois chargée en mémoire se présente sous la forme de tableaux de données classiques (*dataframe*) auxquels a été rajoutée la colonne de géométrie (*geometry*), laquelle décrit la représentation spatiale des objets. Cette géométrie suit une norme commune, le standard ISO 19125 *simple feature access*, appelé plus simplement Simple Features et défini conjointement par l'*Open Geospatial Consortium* (OGC) et l'*International Organization for Standardization* (ISO). Cette norme permet de retranscrire en format numérique la géométrie des objets et des entités, ce qui garantit la reproductibilité des traitements cartographiques entre les trois logiciels.

La grammaire de la norme Simple Features est décrite sur la page Wikipédia associée. On décrit ci-dessous ses représentations les plus courantes. Les points et les polygones sont les représentations des objets les plus utilisées dans les cartes thématiques. Le point sert à représenter l'emplacement des équipements ou des activités dont la surface est négligeable par rapport aux phénomènes d'intérêt. Il est codé à l'aide du mot clef $POINT(X_1, Y_1)$. Un polygone est codé à l'aide du mot clef $POLYGON(X_1, Y_1, X_2, Y_2 ... X_1, Y_1)$ suivi d'une liste de coordonnées [8]. On l'utilise pour décrire l'emprise spatiale d'une commune ou d'un carreau appartenant à un carroyage par exemple. Les fleuves et les routes qui servent à agrémenter une

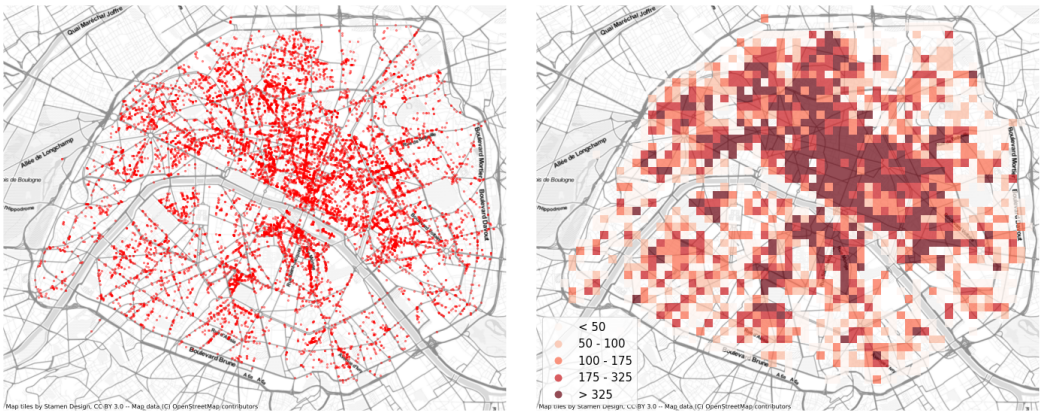
8. la première coordonnée listée correspond à la dernière, de façon à clore le polygone.

Nom du commerce	geometry
Kifolie	POINT (654406.186 6863626.372)
Le bistrot d'André	POINT (646934.770 6859933.893)
Les frérots	POINT (649919.421 6864956.475)
PRET A MANGER	POINT (650048.122 6864069.131)
la Traversée	POINT (652155.733 6865449.423)

Source : https ://opendata.paris.fr/explore/dataset/terrasses-ephemeres

TABLEAU 1 – Exemple de table spatiale : Position des terrasses éphémères parisiennes en 2021

carte sont généralement représentés sous la forme de structure linéaire codée à l'aide du mot clef $LINE(X_1, Y_1, X_2, Y_2...X_n, Y_n)$. Des structures plus complexes, tels que des objets composés de plusieurs parties (territoires avec des exclaves), peuvent être décrites à l'aide des mots clefs $MULTIPOINT$, $MULTILINE$ et $MULTIPOLYGON$.



(a) chaque point représente une terrasse (b) densité carroyée (200m)

Source : https ://opendata.paris.fr/explore/dataset/terrasses-ephemeres

FIGURE 2 – Répartition des terrasses éphémères parisiennes en 2021 - données ponctuelles ou surfaciques

1.2 Stockage des données spatiales

Les données spatiales sont stockées soit dans des fichiers cartographiques soit dans des bases de données. Les fichiers cartographiques sont appelés **fonds de cartes** en géomatique. Le type de stockage conditionne l'accès aux données et les

traitements géomatiques possibles ensuite de réaliser. Les fichiers cartographiques sont efficaces pour diffuser l'information spatiale mais ils s'avèrent inefficaces pour traiter et manipuler de très gros volumes de données. Les bases de données (PostGIS par exemple) répondent à ce besoin mais nécessitent en contrepartie de maîtriser un langage de requêtage (le langage SQL pour PostGIS) et de savoir se connecter à une base de données.

1.2.1 Fichiers cartographiques

Une multitude de formats de fichiers géographiques existent. Le plus connu et le plus utilisé jusqu'à présent est le format *shapefile* associé au logiciel ArcGIS de la société ESRI. Dans ce format, un fond de carte est composé de cinq fichiers : le *.shp* qui est la porte d'entrée sur les données, un fichier *.dbf* qui contient les données attributaires (les données associées aux objets contenus dans le *shapefile*) [9] et d'autres fichiers qui gèrent la projection et la géométrie. Tous les fichiers doivent être localisés au même endroit pour que le fond puisse être ouvert correctement.

D'autres formats tendent progressivement à s'imposer tel que celui proposé par la communauté QGIS, le Geopackage, qui a l'avantage de pouvoir intégrer plusieurs bases/fonds de carte en même temps. Les données sont contenues dans une base de données SQLite, l'extension est $*.gpkg$. Le format geojson est très utilisé par les librairies de cartographie dynamique telles que Leaflet, mais est particulièrement volumineux. *geojson* est une extension spatiale de *json*, les données ne sont pas compressées et sont organisées selon une logique de dictionnaire, ce qui permet de le lire facilement, comme en témoigne le fichier ci-dessous contenant deux points et deux variables (*lat* pour la latitude et *lon* pour la longitude) [10].

```
1 {
2 "type": "FeatureCollection",
3 "crs":{"type":"name", "properties":{"name":"urn:ogc:def:crs:EPSG
    ::4328"}},
4 "features": [
5 { "type": "Feature", "properties": { "lon": 2.416, "lat": 48.848
    },
6 "geometry": { "type": "Point", "coordinates": [ 2.416, 48.848 ] }
    },
7 { "type": "Feature", "properties": { "lon": 2.361, "lat": 48.888
    },
8 "geometry": { "type": "Point", "coordinates": [ 2.361, 48.888 ] }
    }
9 ]
```

9. L'ouverture de ce fichier dans Python ou R pour permet d'accéder aux données attributaires sans avoir à charger les données spatiales.

10. A la ligne 3, apparaît *crs :EPSG : :4328* qui détermine la projection du fond de carte. Les projections sont abordées dans la partie 1.2.3.

```
}
```

Les outils modernes sont généralement capables de lire tous les formats disponibles. Une fois ouvert sous la forme d'une table spatiale, le fond peut être ensuite enregistré dans un autre format.

En R, les commandes *st_read* et *st_write* servent à ouvrir et sauvegarder un fond spatial.

```
library(sf)
url = 'https://france-geojson.gregoiredavid.fr/repo/regions.geojson'
region = st_read(url)
st_write(region, 'region.shp')
```

Code 1 – Lire et écrire des données spatiales en R

En Python, l'ouverture et l'enregistrement d'un fond de carte s'effectue à l'aide des commandes *read_file* et *to_file*.

```
import geopandas as gpd
url = 'https://france-geojson.gregoiredavid.fr/repo/regions.geojson'
region = gpd.read_file(url)
region.to_file('region.shp')
```

Code 2 – Lire et écrire des données spatiales en Python

Très souvent, les données ponctuelles, telles que l'emplacement des terrasses éphémères sont diffusées sous la forme d'un fichier csv contenant deux colonnes de coordonnées. L'instruction *st_as_sf* et *gpd.points_from_xy* permettent de transformer ces fichiers en fond de de carte respectivement sous R et Python. Attention, ces instructions ne fonctionnent pas en présence de valeurs manquantes.

```
library(sf)
library(tidyverse)

terrasses = read_delim('terrasses-ephemeres.csv', delim=';')
sterrasses = st_as_sf(terrasses, coords = c('Coord X', 'Coord Y'), crs=2154)
```

Code 3 – Transformer des coordonnées géographiques en données spatiales avec R

```
import geopandas as gpd
import pandas as pd

terrasses = pd.read_csv('terrasses-ephemeres.csv', sep=";")
geom = gpd.points_from_xy(terrasses['Coord X'], terrasse['Coord Y'])
sterrasses = gpd.GeoDataFrame(terrasses, geometry=geom, crs=2154)
```

Code 4 – Transformer des coordonnées géographiques en données spatiales avec Python

1.2.2 PostGIS, une base de données spatiales

PostGIS [11] est l'extension spatiale de PostgreSQL (comme Oracle spatial est celle d'Oracle) (PostGIS, 2021). Une table spatiale sous PostGIS est analogue à un tableau de données spatiales sous R ou python : une colonne de géométrie décrit à l'aide de la norme Simple Features l'emprise spatiale des objets de la table. PostGIS permet de manipuler des objets géographiques grâce à ces géométries (points, lignes, polygones...), la géométrie étant alors un type à part entière et non une simple succession de coordonnées. Ce type est alors manipulable au même titre que les autres, avec un grand nombre d'opérations possibles. Par rapport aux fichiers cartograhiques, l'apport de PostGIS est de pouvoir stocker et traiter de grandes quantités d'informations. Ceci est rendu possible par l'indexation spatiale basée sur deux principes :

— chaque objet est indexé par sa *bounding box*, soit les 4 couples de coordonnées qui définissent le plus petit rectangle qui le contient ;
— ensuite, un arbre spatial permet de classer et regrouper les objets selon leur proximité et rompre avec une lecture séquentielle des objets contenus dans une table pour prendre en compte leur positionnement relatif.

Le recours à l'indexation spatiale permet d'éviter dans les requêtes spatiales, telles que les jointures spatiales, des calculs inutiles entre des objets très éloignés sans intersection.

L'accès aux données peut se faire à l'aide d'un client SQL (PGAdmin voir encadré, DBeaver...), à travers un logiciel tiers comme QGIS, en ligne de commande (en utilisant les utilitaires de la bibliothèque GDAL), ou à l'aide d'un serveur cartographique (geoserver...). En particulier, deux approches permettent de communiquer avec PostGIS depuis R ou Python. La première consiste à lancer des requêtes SQL qui s'exécutent sous PostGIS. Les résultats restent sous PostGIS, il n'y a pas de retour sous R ou Python. Cette approche permet de configurer ou de créer de nouvelles vues ou tables dans la base de données.

En R, l'accès à une base de données s'effectue à l'aide de la librairie *DBI*. De plus amples informations sur cette librairie et le requêtage d'une base de données depuis R sont disponibles sur le site utilitR.

```
library(DBI)
library(RPostgres)

conn <- DBI::dbConnect(drv = RPostgres::Postgres(),
                       host = "mypostgis",
```

11. Voir http ://postgis.net le site de référence avec la documentation de la dernière version http ://postgis.net/docs/manual-dev/postgis-fr.html voir aussi un tutoriel pour débuter avec PostGIS https ://www.sigterritoires.fr/index.php/debuter-avec-postgresql-postgis-introduction-a-pgadmin4/ Le site de référence

```
                          port = 5432,
                          dbname = "defaultdb",
                          user = "myuser",
                          password = "mypassword")

dbSendQuery(conn, "CREATE EXTENSION postgis;")
```

Code 5 – Rajouter l'extension PostGIS à une base PostgreSQL depuis R

En Python, la communication avec une base de données, telle que Postgres, est dévolue à la librairie *psycopg2*.

```
import psycopg2
conn = psycopg2.connect(host="mypostgis",
                         port = 5432,
                         database="defaultdb",
                         user="myuser",
                         password="mypassword")
cursor = conn.cursor()
cursor.execute("""CREATE EXTENSION postgis;""")
conn.commit()
cursor.close()
conn.close()
```

Code 6 – Rajouter l'extension PostGIS à une base PostgreSQL depuis Python

La connexion à une base de données PostGIS nécessite de spécifier cinq informations :

- Le *port* logiciel permet, sur un ordinateur donné, de distinguer différents interlocuteurs. Ces interlocuteurs sont des programmes informatiques qui, selon les cas, écoutent ou émettent des informations sur ces ports. Un port est distingué par son numéro (Wikipédia, 2022). Par défaut le port de PostGIS est 5432 ;
- l'hôte (*host*) est le nom du serveur ;
- *database* est le nom de la base de données ;
- *user* est le nom de l'utilisateur. Une même base peut avoir plusieurs utilisateurs différents ;
- Enfin *password* est le mot de passe associé à l'utilisateur.

La deuxième approche vise à envoyer ou récupérer de l'information entre une base de données PostGIS et R ou Python. Les données pouvant être très volumineuses, les transferts peuvent être très longs, de sorte, qu'il est préférable de concentrer les calculs sur la base de données de PostGIS et de récupérer les bases finales en fin de traitement.

En R, les commandes *sf*, *st_read* et *st_write* servent également à communiquer des informations avec PostGIS. Dans l'exemple ci-dessous, une table spatiale *sf* de

points aléatoires dans l'emprise spatiale de Paris est créée puis exportée dans PostGIS. Cette dernière table est enfin requêtée sous R.

```
library(DBI)
library(RPostgres)
library(sf)

# Creation d'une table de points sf et export vers PostGIS (dans l
    'emprise spatiale de Paris)
lon = runif(20, 2.25, 2.42)
lat = runif(20, 48.8, 48.9)
df = data.frame(lon, lat)
df = st_as_sf(df, coords = c("lon", "lat"),
                  crs = 4326, agr = "constant")
st_write(df, dsn = conn, layer = "points",
         delete_layer = TRUE)

# Import de la table points sous sf
sql = 'SELECT * FROM points'
points <- st_read(con, query = sql)
```

Code 7 – Envoyer et récupérer une table sous PostGIS depuis R

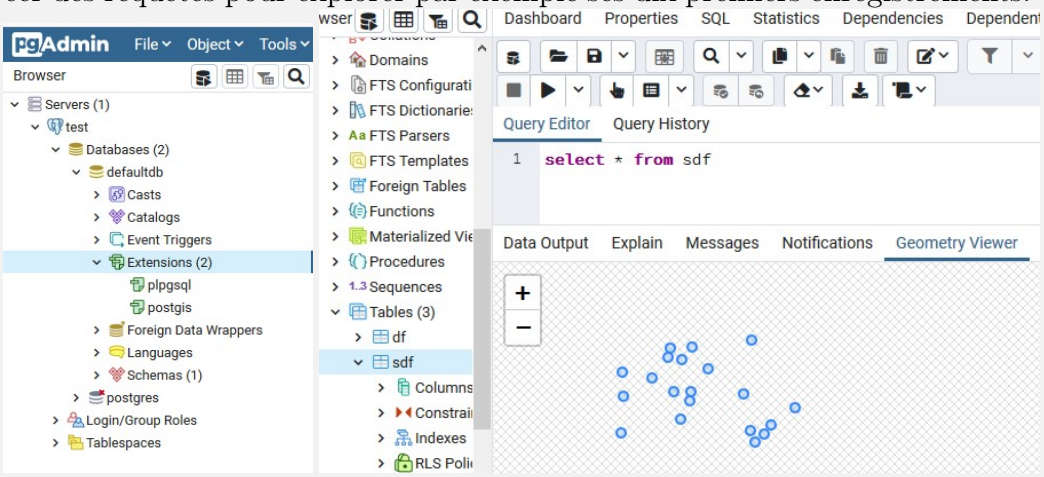
En Python, la librairie *sqlalchemy* permet à la librairie *geopandas* de communiquer avec les bases de données spatiales PostGIS. En revanche, contrairement à R, les transferts s'effectuent à l'aide de deux nouvelles commandes *to_postgis* et *read_postgis*.

```
import pandas as pd
import geopandas as gpd
from sqlalchemy import create_engine

db_connection_url = "postgresql://myusername:mypassword@myhost
    :5432/mydatabase"
engine = create_engine(db_connection_url)

# Creation d'une table de points Geopandas et export vers PostGIS
    (dans l'emprise spatiale de Paris)
df = pd.DataFrame({'lon' : np.random.uniform(2.25, 2.42,20),
                   'lat' : np.random.uniform(48.8, 48.9, 20)})
df = gpd.GeoDataFrame(df,
                      geometry=gpd.points_from_xy(df.lon, df.lat),
                      crs='epsg:4328')
df.to_postgis('points', engine, if_exists='replace')

# Import de la table points sous Python
sql = 'SELECT * FROM points'
points = gpd.read_postgis(sql, engine, geom_col='geometry')
```

Code 8 – Envoyer et récupérer une table sous PostGIS depuis Python

Attention, pour les très grosses tables, la fonction *to_postgis* est très lente. Il est dans ce cas plus judicieux de passer par la commande *COPY* qui autorise l'importation d'un fond de carte directement dans une base *PostGIS*, voir O'Brien (2020). Ceci fonctionne également pour R.

PGAdmin est un des outils les plus utilisés pour paramétrer et requêter une base de données dans un environnement graphique. L'intégration d'une nouvelle base de données s'effectue à l'aide du menu *Create/Server* accessible à l'aide d'un clic droit sur le menu *Servers*. L'onglet *Connection* permet de configurer l'*host*, le *port* et les autres paramètres d'accès à la base de données. Une base PostGIS est une base Postgresql avec une extension *postgis*. Si ce n'est pas le cas, l'extension se rajoute à l'aide du menu *Create/Extension* accessible également à l'aide d'un clic droit sur l'onglet *Extensions*. Dans **PGAdmin**, les tables sont accessibles depuis l'onglet *Schemas/Tables*. Un clic droit sur le nom d'une table permet de lancer des requêtes pour explorer par exemple ses dix premiers enregistrements.



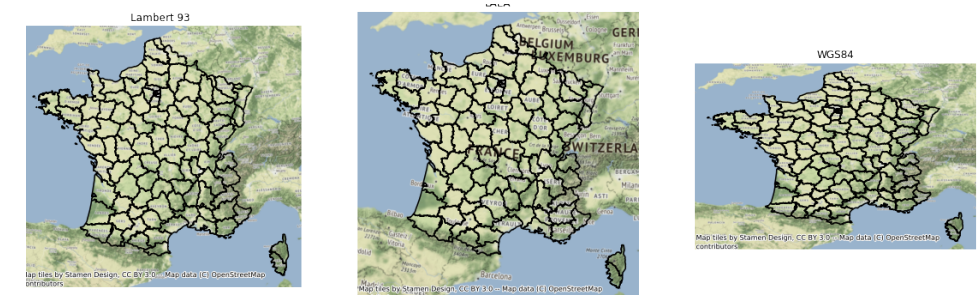
A gauche, présentation de l'arborescence et à droite résultat d'une requête simple.

1.2.3 Système de projection

Tout fond de carte doit être associé à un système de projection. Un système de projection est une méthode permettant de transformer une surface sphérique en une surface plane. Autrement dit, son rôle est de permettre de représenter sur un plan en deux dimensions un phénomène observé à la surface de la Terre. Chaque projection répond à un besoin particulier et est définie par un ensemble de normes [12]. Par exemple, le Lambert 93 est la projection officielle pour les cartes

12. Le site `epsg.io` permet d'explorer les systèmes projections.

de France métropolitaine. Elle a vocation à déformer le moins possible les entités spatiales contenues en France Métropolitaine. La distance entre Lille et Marseille calculée avec la formule de Pythagore sur les coordonnées en Lambert 93 est de 833.812 km. La distance géodésique, tenant compte de la courbure de la Terre, est de 834.005 km. L'approximation de Pythagore est très bonne en France métropolitaine. C'est d'ailleurs cette distance qui est utilisée en statistique spatiale. En revanche, une distance vol d'oiseau entre Lille et l'Australie génère une erreur de plus de 1500 km. La projection LAEA est le système en vigueur en Europe, utilisé entre autres par Eurostat pour diffuser de l'information spatiale. Ce système vise à déformer le moins possible la géométrie des pays européens.



(a) France métropolitaine

(b) Le carreau de 200 m recouvrant l'ancienne Direction Générale de l'Insee

FIGURE 3 – Les trois principales projections utilisées par les statisticiens : Lambert 93, LAEA (projection européenne) et WGS 84 (utilisée par le système de positionnement par satellite GPS)

En fonction de la projection retenue, les représentations des territoires de la France métropolitaine peuvent être très différentes (fig 3). En particulier, la carte en WGS 84 se différencie fortement des deux autres. La projection WGS 84 est très utilisée pour la cartographie dynamique et les systèmes de positionnement par satellite. Contrairement aux projections Lambert 93 et LAEA, la projection

WGS 84 n'est pas une projection en tant que telle, mais une modélisation du globe terrestre. Les coordonnées sont des angles qui permettent de situer tout point sur la surface de la terre à partir d'une référence. Par exemple, le centre de Lille a pour coordonnées $(3.063, 50.636)$ et celui de Marseille $(5.369, 43.296)$. La projection WGS84 ne permet pas de calculer directement des distances et est donc inadaptée à l'analyse spatiale. Une erreur courante est d'intervertir les deux angles.

Une projection est identifiée à l'aide d'un numéro EPSG (European Petroleum Survey Group). Par exemple, 2154 pour le lambert 93, 3035 pour le LAEA (projection européenne) et 4326 pour le WGS 84. On passe d'une projection à une autre, en effectuant une opération de reprojection. Cette opération peut être très longue en fonction de la taille de la couche cartographique. Il est donc recommandé d'adopter une projection unique dès le début d'un projet cartographique.

En R, l'opération de reprojection s'effectue à l'aide de la commande *st_transform* :

```
library(sf)
departement = read_sf('dep_francemetro_2020.geojson')
departement = st_transform(departement, 2154)
```

Code 9 – Reprojection d'un fond départemental en R

En Python, cette opération s'obtient avec la commande *.to_crs* :

```
import geopandas as gpd
departement =  gpd.read_file('dep_francemetro_2020.geojson')
departement = departement.to_crs("EPSG:2154")
```

Code 10 – Reprojection d'un fond départemental en Python

En PostGIS, cette opération s'effectue à l'aide de la commande *ST_TRANSFORM* :

```
ALTER TABLE departement
 ALTER COLUMN geometry TYPE geometry(MultiPolygon,2154)
  USING ST_Transform(geometry,2154)
```

Code 11 – Reprojection d'un fond départemental en PostGIS

1.3 Exemple : création d'une couche carroyée à partir de données ponctuelles

La transformation d'une donnée ponctuelle en une donnée surfacique ne nécessite pas toujours l'emploi d'opérations géomatiques spécifiques. La transformation du fond de carte des terrasses éphémères en des données carroyées de 200m de côté

proposée ici repose sur l'application de fonctions mathématiques élémentaires sur les coordonnées géographiques [13].

La première étape consiste à déterminer les coordonnées du coin sud-est du carreau d'appartenance de chaque terrasse en recourant à l'aide de la fonction valeur entière : $X_{car} = floor(X/s) * s$ et $Y_{car} = floor(Y/s) * s$. Où $size$ est la taille désirée des carreaux. Le regroupement des données selon ces deux nouvelles coordonnées permet de créer la partie attributaire de la future table de données carroyées. Enfin, on déduit la géométrie des carreaux à partir des coordonnées de leur coin Sud-Est :

$Polygon(X_{car}, Y_{car}, X_{car}, Y_{car}+size, X_{car}+size, Y_{car}+size, X_{car}+size, Y_{car}, X_{car}, Y_{car})$

```r
library(sf)
library(tidyverse)

size = 200

terrasses = read_delim('terrasses-ephemeres.csv', delim=';')

terrasses['x'] = floor(terrasses$'Coord X' / size) * size
terrasses['y'] = floor(terrasses$'Coord Y' / size) * size

terrasses$densite = 1 / (size / 1000)^2

lterrasses = terrasses %>%
             group_by(x, y) %>%
             summarize(densite = sum(densite)
                       , .groups = 'drop')
lterrasses$geometry <- sprintf("POLYGON ((%f %f, %f %f, %f %f, %f %f, %f %f))",
                               lterrasses$x, lterrasses$y,
                               lterrasses$x+size, lterrasses$y,
                               lterrasses$x+size, lterrasses$y+size,
                               lterrasses$x, lterrasses$y+size,
                               lterrasses$x, lterrasses$y)
lterrasses <- st_as_sf(lterrasses, wkt = "geometry", crs = 2154)
plot(lterrasses)
```

Code 12 – Transformation de données ponctuelles en données surfaciques en R

```python
import geopandas as gpd
import pandas as pd
import numpy as np
from shapely.geometry import Polygon
```

13. Cette transformation fonctionne uniquement si les coordonnées sont exprimées dans une projection métrique, voir la sous-section précédente pour plus de renseignements sur les projections.

```
5
6  terrasses = pd.read_csv('terrasses-ephemeres.csv', sep=";")
7  size = 200
8
9  terrasses['x'] = np.floor(terrasses['Coord X'] / 200) * 200
10 terrasses['y'] = np.floor(terrasses['Coord Y'] / 200) * 200
11 terrasses['densite'] = 1 / (size / 1000) **2
12 lterrasses = terrasses.groupby(['x', 'y'],
13                                 as_index=False)['densite'].sum()
14
15 geometry = [Polygon([(x, y),
16                       (x + size, y),
17                       (x + size, y + size),
18                       (x, y + size)])
19                       for x,y in zip(lterrasses.x,lterrasses.y)]
20 lterrasses =  gpd.GeoDataFrame(lterrasses, geometry=geometry, crs=
       'EPSG:2154')
```

Code 13 – Transformation de données ponctuelles en données surfaciques en Python

```
1  CREATE table lterrasses as
2  select temp.x, temp.y, sum(temp.densite) as "densite",
3  ST_Polygon( CONCAT('LINESTRING(', temp.x,' ', temp.y, ',',
4                        temp.x + 200,' ', temp.y, ',',
5                        temp.x + 200,' ', temp.y +200 , ',',
6                        temp.x,' ', temp.y + 200, ',',
7                        temp.x,' ', temp.y, ')'), 2154 ) as geometry
8  from
9  (select floor(t."Coord X" / 200) * 200 as "x",
10        floor(t."Coord Y" / 200) * 200 as "y",
11        1 / (200.00/1000.00) as "densite"
12 from terrasses as t) as "temp"
13 group by x,y;
```

Code 14 – Transformation données ponctuelles en données surfaciques en PostGIS

Cette approche simple fonctionne uniquement parce que les carreaux sont agencés de façon régulière dans l'espace. Pour des objets plus complexes il est indispensable de recourir à des traitements géomatiques.

2 Traitement des données spatiales

Un même phénomène tel que l'organisation des terrasses éphémères peut être représenté de diverses manières. L'enjeu pour le statisticien est de savoir transformer la représentation géométrique d'un phénomène en fonction des objectifs statistiques et esthétiques visés. Ces transformations peuvent être simples et ne pas nécessiter de fonctions spatiales complexes comme vu dans l'exemple de la figure 2b) mais plus généralement, les transformations sont le résultat d'un enchaînement de traitements géomatiques plus complexes.

De nombreuses fonctions géomatiques existent, seules les plus utiles par les statisticiens sont discutées dans cette partie. Pour plus de renseignements, le lecteur peut consulter l'aide mémoire en R des fonctions géomatiques (Pebesma, 2021), les fonctions géomatiques en Python et la liste des fonctions géomatiques en PostGIS.

2.1 Fonctions et transformations géométriques élémentaires

Les fonctions et transformations élémentaires visent à transformer la nature ou le type de géométrie d'un fond de carte spatiale sans avoir à recourir à une information spatiale tierce. Elles sont classées dans cette section en fonction de la géométrie initiale de la donnée et de la géométrie visée.

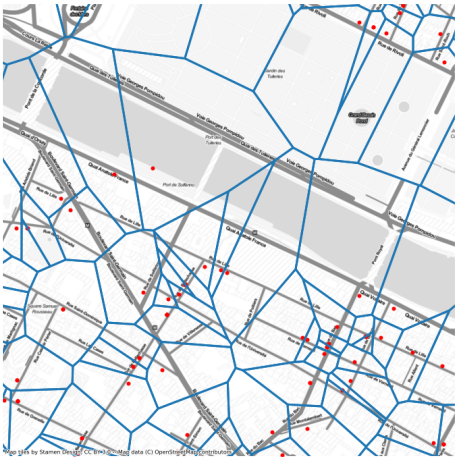
2.1.1 Transformation de données ponctuelles en données surfaciques

Les zones tampons (*buffers*) et les diagrammes de Voronoi sont utilisés pour associer à chaque point d'une couche ponctuelle une surface.

Zones tampons (*buffers*) Les zones tampons associent à chaque point de la couche considérée un disque centré sur celui-ci de rayon défini par l'utilisateur (voir figure 4 a). Les zones tampons servent souvent à déterminer des zones d'accessibilité selon une logique vol d'oiseau autour d'un équipement, d'un point d'intérêt ou à définir l'environnement d'un ménage ou d'un individu. Sous R et PostGIS, les zones tampons sont obtenues à l'aide de la fonction *st_ buffer* et sous Python avec *buffer*. Un exemple d'application des zones tampons est présenté dans la section 2.3.1.

Diagramme de Voronoi Le diagramme de Voronoi est un pavage de l'espace qui associe à chaque point de la couche considérée l'ensemble des points du plan plus proches de ce point que d'aucun autre (voir figure 4 b). Cette modélisation de l'espace permet de définir des zones de chalandise sans recouvrement, contrairement aux espaces définis à l'aide d'une zone tampon.

(a) Zones tampons (100 m)



(b) Diagramme de Voronoi

FIGURE 4 – Associer une surface à un point : exemple avec les terrasses éphémères

En R, le calcul des diagrammes de Voronoi se réalise à l'aide de la fonction *st_voronoi* disponible directement dans la librairie sf.

```r
library(sf)

terrasses = st_read('sterrasses.gpkg')

voronoi <- terrasses %>%
  st_union() %>%
  st_voronoi() %>%
  st_collection_extract()
```

Code 15 – Diagramme de Voronoi des terrasses éphémères en R

En Python, le calcul des diagrammes de Voronoi s'effectue à l'aide du *package geovoronoi*. La première étape consiste à récupérer les coordonnées (x, y) à partir de la colonne géométrie.

```python
import geopandas as gpd
import numpy as np
from  geovoronoi import voronoi_regions_from_coords

terrasses = gpd.read_file('sterrasses.gpkg')

coord = np.vstack([terrasses.geometry.x.T, terrasses.geometry.y.T
    ]).T
pol, _ = voronoi_regions_from_coords(coord, terrasses.unary_union.
    convex_hull)
Voronoi = gpd.GeoSeries(list(pol.values()))
```

Code 16 – Diagramme de Voronoi des terrasses éphémères en Python

En PostGIS,

```
create table voronoi_terrasses as
SELECT
 ST_VoronoiPolygons(st_union(geometry)) As "geometry"
FROM terrasses;
```

Code 17 – Diagramme de Voronoi des terrasses éphémères en PostGIS

Emprise géométrique minimale Déterminer l'emprise géométrique minimale d'un ensemble de points, c'est-à-dire déterminer la surface englobant cet ensemble de points, est une opération courante en géomatique. Cette opération est utile par exemple pour affecter une surface à un ensemble de points afin par exemple de calculer une densité. Parmi l'infinité de méthodes permettant de trouver une telle surface, la recherche du polygone convexe minimal (enveloppe convexe ou *convex hull*) englobant tous les points est l'une des méthodes les plus utilisées. Une autre solution simple consiste à déterminer le plus petit rectangle qui contient tous les points et dont les côtés sont parallèles aux axes de la projection (rectangle d'emprise minimale ou *envelope*), voir figure 5 pour une illustration concernant la ville de Paris.

(a) Polygone convexe d'emprise minimale (*convex hull*)

(b) Rectangle d'emprise minimale (*envelope*)

FIGURE 5 – Emprises géométriques minimales de la ville de Paris

En R, le polygone convexe minimal (*convex hull*) s'obtient en regroupant tous les points en un unique objet à l'aide de la fonction *st_union* puis en appliquant la commande *st_convex_hull*. Il n'existe pas de fonction pour calculer directement le rectangle d'emprise minimale (*envelope*), celle-ci s'obtient en combinant la commande *st_bbox* qui retourne les coordonnées de l'enveloppe et la commande *st_as_sfc* qui permet de les transformer en un polygone.

```
library(sf)
```

```
2
3 terrasses = st_read('sterrasses.gpkg')
4 convex_hull = terrasses %>% st_union() %>% st_convex_hull()
5 envelope = terrasses %>% st_union() %>% st_bbox() %>% st_as_sfc
```

Code 18 – Polygones d'emprise spatiale minimale en R

En Python, le polygone convexe minimal et le rectangle d'emprise minimale, s'obtiennent en regroupant tous les points en un unique objet à l'aide de la fonction *unary_union* puis en appliquant la fonction *convex_hull* ou la fonction *envelope*.

```
1 import geopandas as gpd
2 terrasses = gpd.read_file('sterrasses.gpkg')
3 convex_hull = gpd.GeoSeries(terrasses.unary_union.convex_hull, crs
    ='EPSG:2154')
4 envelope = gpd.GeoSeries(terrasses.unary_union.envelope, crs='EPSG
    :2154')
```

Code 19 – Polygones d'emprise spatiale minimale en Python

En PostGIS, le polygone convexe minimal et le rectangle d'emprise minimale s'obtiennent à l'aide des fonctions *st_convexhull* et *st_envelope*.

```
1 create table convex_hull as
2 SELECT
3  ST_convexhull(st_union(geometry)) As "geometry"
4 FROM terrasses;
5
6 create table envelope as
7 SELECT
8  ST_envelope(st_union(geometry)) As "geometry"
9 FROM terrasses;
```

Code 20 – Polygones d'emprise spatiale minimale en PostGIS

2.1.2 Simplification de données surfaciques ou transformation de données surfaciques en données ponctuelles

Centres de gravité ou *centroids* Les centres de gravité sont utilisés pour résumer l'emprise spatiale d'une couche de polygones par des points. En R sf, et PostGIS, ils s'obtiennent à l'aide de la fonction *st_centroid* et en Python, la fonction *centroid*. Ce passage par une donnée ponctuelle est parfois nécessaire pour obtenir une transformation d'une donnée surfacique en un autre type de donnée surfacique. Par exemple, un diagramme de Voronoi sur les centres de gravité d'une couche de polygones permet d'obtenir une version simplifiée de cette couche plus adaptée à une diffusion web (voir figure 6).

Simplifier la géométrie d'une couche avec mapshaper. La simplification de la géométrie d'une couche de polygones peut être également obtenue à l'aide de la solution *open source* **mapshaper** [14]. **mapshaper** a l'avantage de simplifier la géométrie sans altérer son organisation globale : les contigüités entre les objets sont conservées. L'installation de **mapshaper** est détaillée sur la page github associée. Un exemple d'utilisation en ligne de commande est proposée ci-dessous :

```
curl https://minio.lab.sspcloud.fr/h529p3/diffusion/
    armf_bdtopo_dep_75_2021.zip --output armf.zip
unzip armf.zip
mapshaper armf_bdtopo_dep_75_2021.shp  -simplify dp 20% -o format=
    shapefile armf_simplify.shp
```

Code 21 – Compression du fond de carte des arrondissements parisiens à l'aide de mapshaper

Le *package* *Rmapshaper* permet en R d'accéder à *mapshaper* sans avoir à écrire des lignes de commande dans un terminal. `https://mapshaper.org/` permet d'accéder directement aux fonctions de *mapshaper* en ligne dans une interface graphique, les transformations sont effectuées en local et les fichiers ne sont pas envoyés sur le serveur de l'application.

(a) Fond de carte original (b) Simplification avec mapshaper (compression 20%) (c) Simplification à l'aide des diagrammes de Voronoi

FIGURE 6 – Simplification d'un fond de carte : simplifier le contour des arrondissements parisiens

Autres modifications de couches surfaciques Il est aussi possible de créer des zones tampons, des polygones d'emprise minimale autour des couches surfaciques, en appliquant les fonctions décrites dans la partie 2.1.1. Un exemple d'application des zones tampons avec un rayon négatif est proposé dans la partie 2.3.1. Dans ce cas, les tampons servent à éroder les polygones.

14. `https://mapshaper.org/`

2.2 Transformations complexes mettant en relation plusieurs éléments d'une couche ou de plusieurs couches

Les transformations présentées dans cette partie visent à produire une nouvelle information spatiale et statistique à partir d'une ou de plusieurs couches. Elles peuvent être bien plus longues à calculer que les transformations précédentes et peuvent nécessiter beaucoup de mémoire. Le recours à PostGIS permet généralement de réaliser des calculs impossibles à terminer dans une session R ou Python.

2.2.1 Agrégation des données spatiales

L'**agrégation spatiale** vise à regrouper les unités spatiales entre elles pour former des unités spatiales plus larges. Cette transformation est utile pour déduire à partir d'une couche spatiale disposant d'une granularité fine une couche à la granularité plus large. Par exemple, il arrive souvent de vouloir construire à partir d'un fond départemental un fond régional, ou à partir d'un fond communal un fond des Établissement public de coopération intercommunale (EPCI).

Pour ce faire, deux opérations sont réalisées conjointement : la géométrie des entités appartenant à une même macro-zone est fusionnée, et les données statistiques associées sont agrégées.

En R, avec le *package dplyr*, l'agrégation des données spatiales repose sur la fonction *group_by* et *st_union* comme celle sur données classiques :

```r
library(sf)
library(dplyr)

departement = read_sf('dep_francemetro_2020.geojson')

region  = departement %>%
  group_by(reg) %>%
  summarize(geometry = st_union(geometry),
            surf = sum(surf))
```

Code 22 – Agrégation d'une couche géographique en R

En Python, l'agrégation s'obtient avec la commande *dissolve*. Le paramètre *aggfunc* détermine la fonction d'agrégation à appliquer aux données numériques associées.

```python
import geopandas as gpd
departement =  gpd.read_file('dep_francemetro_2020.geojson')
region = departement.dissolve(by='reg', aggfunc='sum', as_index=
    False)
```

Code 23 – Agrégation d'une couche géographique en Python

En SQL, tout comme en R, l'agrégation s'obtient à l'aide de la commande *group by* et de la fonction *st_ union* qui regroupent les objets géographiques appartenant à un même groupe.

```
create table region as
SELECT ST_Union(geometry) as "geometry", sum(surf) as "surf"
 FROM departement
 GROUP BY reg;
```

Code 24 – Agrégation d'une couche géographique en SQL

2.2.2 Intersection et jointure des données spatiales

Une **jointure spatiale** vise à joindre deux couches spatiales selon la position relative de leurs objets respectifs. Dans une grande majorité des cas, c'est leur *intersection* qui est déterminée : la jointure relie alors deux objets s'ils ont au moins un point du plan en commun. La jointure spatiale est un outil important pour le statisticien car elle permet de produire les statistiques, comme dénombrer un phénomène selon un découpage spatial. Les exemples ci-dessous comptabilisent les terrasses éphémères contenues dans l'emprise spatiale des carreaux de la source Filosofi. Une fois la jointure réalisée, une agrégation permet de produire les statistiques au carreau.

En R, la jointure s'obtient à l'aide de la fonction *st_join*, par défaut selon l'intersection.

```
library(sf)
library(tidyverse)

carreaux = read_sf('Filosofi2015_carreaux_200m_Paris.gpkg')
terrasses = read_sf('terrasses.gpkg')

temp = st_join(carreaux, terrasses, left=F)
#left=F indique une jointure    droite
temp = temp %>%
       st_drop_geometry %>%
       group_by(IdINSPIRE) %>%
       summarize(densite=n())
```

Code 25 – Intersection spatiale en R

En Python, la jointure s'obtient à l'aide de la fonction *sjoin*, par défaut également selon l'intersection.

```
import geopandas as gpd
import pandas as pd

carreaux = gpd.read_file('Filosofi2015_carreaux_200m_Paris.gpkg',
   driver="GPKG")
```

```
terrasses = gpd.read_csv('terrasses-ephemeres.csv', sep=";")

temp = gpd.sjoin(carreaux, terrasses)
temp['densite'] = 1
temp = temp.groupby(['IdINSPIRE'], as_index=False)['densite'].sum
    ()
```

Code 26 – Intersection spatiale en Python

En PostGIS, la jointure spatiale et l'agrégation sont réalisées au sein d'une même requête.

```
create table carreaux_terrasses as
SELECT
  carreaux."IdINSPIRE",
  carreaux.geometry,
  count(*) / power(200.0/1000.0,2)  as "densite"
FROM carreaux
JOIN terrasses
ON ST_intersects(carreaux.geometry, terrasses.geometry)
group by carreaux."IdINSPIRE", carreaux.geometry
```

Code 27 – Intersection spatiale en PostGIS

Avec *intersects*, deux objets sont reliés dès qu'ils partagent au moins un point en commun. D'autres types de jointure, tels que *contains* ou *within* peuvent être mobilisés. Avec *contains* les objets sont reliés si aucun point du deuxième objet se trouve en dehors du premier. *within* est identique à *contains* sauf que les deux objets sont permutés dans l'appel de la fonction [15] Dans le cas d'une jointure d'un fond de carte surfacique avec un fond ponctuel, *intersects* et *contains* et *within* donnent des résultats similaires.

2.3 Exemples de manipulation de données spatiales

Cette partie illustre les manipulations de fonds de carte sur trois exemples classiques en statistique publique : la détermination d'une emprise spatiale, ici de zones bâties, un calcul d'accessibilité, ici aux terrasses éphémères parisiennes supposées ponctuelles, ou aux parcs et jardins dont l'emprise spatiale ne se résume pas à un point, et enfin l'agrégation de données carroyées à l'échelle communale.

2.3.1 Définir l'emprise spatiale des zones bâties

Définir les limites morphologiques d'une ville est une opération complexe qui repose sur une série d'abstractions géographiques [16]. La principale étant de pro-

15. Des explications détaillées peuvent être trouvées dans Entin (2019).

16. Les bases de données de la BD Topo disponibles sur le site de l'IGN (`ign.fr`) contiennent l'ensemble des bâtiments de France et permettent de s'exercer à la constitution des zones bâties.

duire une continuité à partir de bâtiments à l'emprise spatiale finie et isolés les uns des autres. Pour définir les unités urbaines par exemple, l'Insee retient un seuil de distance de 200 m pour délimiter les zones agglomérées (tâches bâties). Deux bâtiments situés à moins de 200 m appartiennent à la même zone. Les zones agglomérées finales sont obtenues en appliquant cette règle de proche en proche aux agrégats déjà constitués. En géomatique, la production de telles taches bâties repose sur un enchaînement d'opérations élémentaires :

1. Calcul des zones tampons de rayon 100 m autour de chaque bâtiment. Le résultat est une liste de *POLYGON* correspondant à chaque tampon ;
2. Regroupement des zones tampons qui se chevauchent pour former une unique zone soit un *MULTIPOLYGON* ;
3. Application d'un tampon négatif de 100 m sur la zone précédente pour éroder celle-ci et obtenir une tache dont les frontières se rapprochent des bâtiments ;
4. Séparation des différentes sous-parties composant la tache érodée pour obtenir une liste de *POLYGON* représentant chaque tache indépendante.

En R, l'union des tampons (étape 2) est réalisée avec la commande *st_ union* La transformation (étape 4) d'un *MULTIPOLYGON* en une liste de *POLYGON* s'obtient à l'aide de l'instruction *st_ cast* avec le paramètre *'POLYGON'*.

```
library(sf)
batiment = read_sf("BATIMENT_extrait.geojson")
batiment = st_transform(batiment, 2154)
taches_urbaines = st_buffer(batiment,100) %>%
                  st_union() %>%
                  st_buffer(-100) %>%
                  st_cast('POLYGON')
```

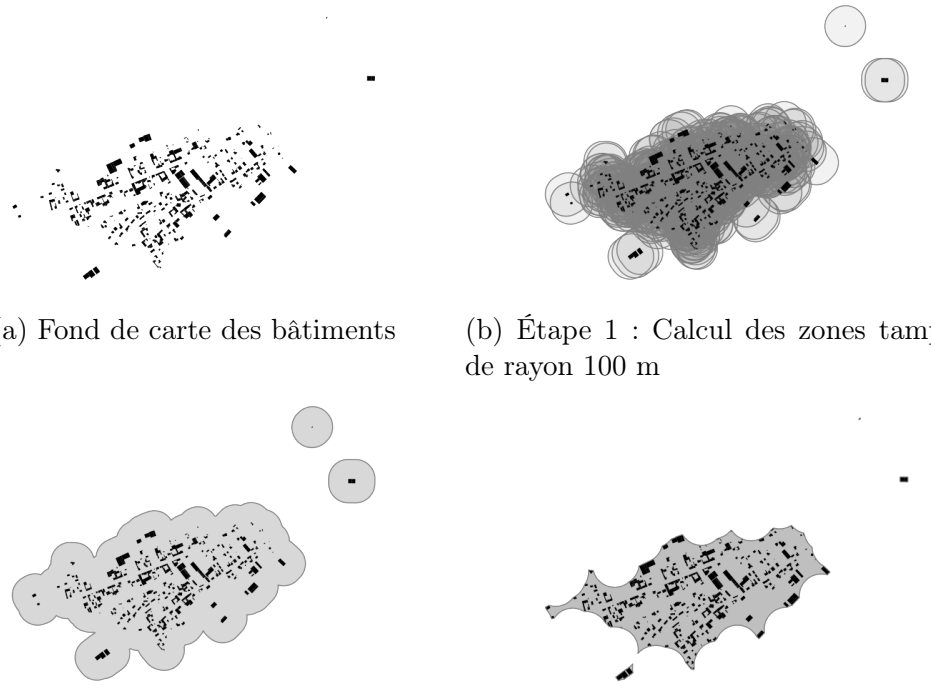
Code 28 – Délimitation des taches en R

En Python, la commande *unary_ union* regroupe les tampons (étape 2) et la fonction *explode* réalise la désagrégation d'un *MULTIPOLYGON* en *POLYGON* (étape 4).

```
import geopandas as gpd
batiment = gpd.read_file('BATIMENT_extrait.geojson').to_crs(2154)

taches_urbaines = batiment.geometry.buffer(100).unary_union.buffer
    (-100)
taches_urbaines = gpd.GeoSeries(taches_urbaines).explode()
```

Code 29 – Délimitation des taches en Python

En PostGIS, la désagrégation d'un *MULTIPOLYGON* en *POLYGON* (étape 4) s'effectue à l'aide de l'instruction *st_ split*.



(a) Fond de carte des bâtiments

(b) Étape 1 : Calcul des zones tampons de rayon 100 m

(c) Étape 2 : Regroupement des zones tampons

(d) Étapes 3 et 4 : Application d'un tampon négatif et séparation des différentes sous-parties

FIGURE 7 – Les étapes de construction des taches bâties

```
CREATE table tache_urbaine as
select st_split(st_buffer(ST_Union(st_buffer(geometry, 100)),
    -100)) as "geometry"
from batiment;
```

Code 30 – Délimitation des taches en PostGIS

2.3.2 Déterminer l'accessibilité à un équipement

Un deuxième exemple classique consiste à déterminer l'accessibilité d'une population à des équipements. C'est une opération apparemment simple mais qui se révèle difficile à mener lorsque les volumétries des objets considérés sont importantes. Naïvement, l'accessibilité d'une population carroyée dont on a résumé la spatialité au point centroïde de chaque carreau à des équipements eux aussi réduits à un point consiste à calculer deux à deux tous les couples de distances et conserver uniquement ceux dont la distance est inférieure à un certain seuil. Lorsque les ensembles de carreaux et d'équipements sont grands, le produit cartésien entre les

deux ensembles peut être gigantesque et ces calculs impossibles à réaliser dans leur globalité. Pour autant, le calcul d'une grande partie des couples n'a pas d'intérêt, et ce problème peut largement être simplifié. En effet, si le seuil de distance est de 200 m, l'appréciation des distances entre les carreaux de Paris et des équipements à Marseille n'est pas pertinente.

Les structures en arbre, telles que les **k-d trees** ont été développées pour répondre à ce type de question (Bentley, 1975). La structure en arbre permet de stocker des points et de requêter plus rapidement qu'en parcourant le tableau de points. Elle évite de calculer explicitement les distances pour des couples de points éloignés, comme entre Paris et Marseille pour un seuil de distance de 200 m par exemple. Les structures en arbre permettent d'appliquer des algorithmes gourmands en mémoire et en temps de calcul sur des ensembles de données de très grandes tailles comme les algorithmes de *clustering* spatial qui relèvent du *Machine Learning* notamment la *DBSCAN* (voir encadré). Ci-dessous, on cherche à déterminer pour chaque terrasse éphémère ses terrasses voisines dans un rayon de 200 m (voir Figure 8 pour le résultat).

En R, la structure **k-d tree** est implémentée dans la librairie *dbscan*, au sein de laquelle La fonction *frNN* recherche les points voisins dans un rayon donné (*nearest neighbors*).

```
library(dbscan)
library(tidyverse)

terrasses = read_delim('https://opendata.paris.fr/explore/dataset/
    terrasses-ephemeres/download/?format=csv&timezone=Europe/Berlin
    &lang=fr&use_labels_for_header=true&csv_separator=%3B', delim
    =';')
voisins = frNN(terrasses[,c('Coord X', 'Coord Y')] , 200,
    terrasses[,c('Coord X', 'Coord Y')] )
voisins = lapply(voisins$id[1:100], length))
# la fonction frNN renvoie une liste de points voisins pour chaque
    point. L'agregation s'effectue dans un deuxieme temps.
```

Code 31 – Requête spatiale reposant sur les structures **k-d tree** en R

En Python, la librairie *scikit-learn* contient une implémentation de la structure **k-d tree**.

```
import pandas as pd
from sklearn.neighbors import KDTree

#terrasses = pd.read_csv('=Europe/Berlin&lang=fr&
    use_labels_for_header=true&csv_separator=%3B', sep=";")
terrasses = terrasses[terrasses['Coord X']>0]
tree = KDTree(terrasses[['x', 'y']])

```

```
voisins = tree.query_radius(terrasses[[ 'x', 'y']], 200,
    count_only=True)
#count_only permet de calculer directement le nombre de voisins
```

Code 32 – Requête spatiale reposant sur les structures **k-d tree** en Python

FIGURE 8 – Localisation des terrasses éphémères disposant de plus de 100 terrasses voisines dans un rayon de 200m

Le recours à une approche géomatique permet des calculs d'accessibilité plus complexes, par exemple l'accessibilité à des équipements dont l'emprise spatiale ne se résume pas à un point. Des zones tampons dont le rayon correspond au seuil de distance permettent alors de déterminer l'emprise spatiale d'une accessibilité à vol d'oiseau. Les codes ci-dessous déterminent la population à moins de 500 m d'un jardin dans la ville de Paris. Les parcs et jardins proviennent d'une base en *open data* proposée par la ville de Paris et accessible ici (fig 9).

```
library(sf)
population = st_read('Filosofi2015_carreaux_200m_metropole.shp')

parc = st_read('jardin.geojson')
parc = st_transform(parc, 2154)
parc = parc[as.numeric(st_area(parc)) > 10000,]
buffer = st_buffer(parc, 500)

```

```r
res_intersection = st_join(population, buffer, left=FALSE)
plot(res_intersection$geometry)
```

Code 33 – Déterminer les carreaux à moins de 500 m d'un parc ou d'un jardin en R

```python
import geopandas as gpd
population = gpd.read_file('Filosofi2015_carreaux_200m_metropole.shp')
parc = gpd.read_file('jardin.geojson').to_crs('EPSG:2154')
parc = parc[parc.area>10000]
buffer = parc.copy()
buffer.geometry = buffer.buffer(500)

res_intersection = gpd.sjoin(population, buffer, how="inner", op='intersects')
```

Code 34 – Déterminer les carreaux à moins de 500 m d'un parc ou d'un jardin en Python

(a) Les parcs et leur tampon de 500 m (b) En vert les carreaux qui intersectent un tampon et en rouge les carreaux isolés

FIGURE 9 – Carreaux à moins de 500m d'un parc

Un exemple de *clustering* spatial : DBSCAN

DBSCAN pour *density-based spatial clustering of applications with noise* est un algorithme de *clustering* proposé par Ester, Kriegel, Sander, et Xu (1996) qui vise à repérer les agrégats spatiaux de points pour former des *clusters* soient les zones de forte densité de points. Avant cet algorithme, ces zones étaient délimitées en calculant par exemple une densité à l'aide d'une estimation non paramétrique à noyau. Les *clusters* étaient obtenus en conservant comme agglomérées les zones de densité supérieure à un seuil défini par l'utilisateur. La DBSCAN évite de recourir à une estimation non paramétrique, ce qui en fait un algorithme très rapide. Les implantations sous R (*package* DBSCAN), Python (librairie scikit-learn) et PostGIS (fonction ST_ClusterDBSCAN) repose sur l'exploitation des arbres *K-D trees*. Les deux paramètres *eps* et $Minpts$ déterminent la taille des clusters. *eps* est analogue au rayon de lissage (*bandwidth*) de l'estimation de densité non paramétrique et $Minpts$ correspond au seuil de troncature des densités. De façon simplifiée, l'algorithme se décompose en trois étapes :

- pour chaque point, son nombre de points voisins dans un rayon *eps* est calculé ;
- seuls les points (points cœurs) dont le nombre de voisins est supérieur à $Minpts$ sont conservés pour la suite de l'analyse ;
- un graphe non orienté est constitué en reliant tous les points cœurs situés à une distance inférieure à *eps*. Les agrégats sont les composantes connexes de ce graphe (tous les points d'une même composante peuvent être reliés par un ou plusieurs chemins).

La DBSCAN a été utilisée dans des travaux de l'Insee pour déterminer l'emprise des centres villes en recherchant les polarités commerçantes (Baëhr et Courthial, 2020) ou encore l'emprise des villages dans le cadre de la mesure du mitage occasionné par les nouvelles constructions (Himpens, Poulhes, et Sémécurbe, 2021).

Un exemple d'application de la DBSCAN est proposé ci-dessous sur les terrasses éphémères. Les paramètres de la DBSCAN sont ceux de la partie 2.3.2, $eps = 200$ et $Minpts = 100$.

```
library(dbscan)
agregats = dbscan(terrasses[,c('Coord X', 'Coord Y')], 200,
    50)
agregats$cluster # renvoie l'agregat d'appartenance de chaque
    point. 0 caracterise les points isoles.
```

Code 35 – Application de la DBSCAN à la détection des agrégats de terrasses en R

```
from sklearn.cluster import DBSCAN
agregats = DBSCAN(eps=200, min_samples=100).fit(sterrasses[['
    x', 'y']])
agregats.labels_ # renvoie l'agregats d'appartenance. -1
    caracterise les points isoles
```

Code 36 – Application de la DBSCAN à la détection des agrégats de terrasses en Python

Les agrégats contiennent plus de points que ceux obtenus avec la recherche des points voisins (fig 8), ceci s'explique par le fait que l'algorithme intègre, par défaut, dans les agrégats les points à une distance inférieur à *eps*. Ce comportement peut être modifié sous R avec l'option *borderPoints*.

2.3.3 Agréger les données carroyées à l'échelle des communes

L'agrégation de données ponctuelles dans des données surfaciques ne posent pas de difficulté particulière car chaque point appartient à un unique carreau. Il n'en est pas de même lorsque l'on recherche à déduire d'une information statistique contenue dans une couche surfacique (A) dans une autre couche surfacique (B). La stratégie consistant à réaliser l'intersection spatiale puis l'agrégation n'est pas satisfaisante. Une unité spatiale de la couche (A) pouvant chevaucher plusieurs unités spatiales de la couche (B), les statistiques agrégées sur la couche (B) sont

susceptibles de contenir de nombreux doubles comptes. Afin de tenir compte des doubles comptes, il est nécessaire de désagréger l'information au prorata de la surface commune. Dans ce cas, la population communale totale estimée correspond exactement dans ce cas à la population carroyée totale. Cette opération augmente le temps de calcul car il est nécessaire de calculer explicitement les intersections entre les communes et les carreaux et est difficilement réalisable sous Python ou R. Elle est réalisable avec PostGIS.

```
select "INSEE_COM",
        sum("Men"*interarea /area) "Men",
        sum("Ind"*interarea/area) "Ind"
from (
      select c."INSEE_COM", f."Men", f."Ind",
      ST_Area(ST_Intersection(c.geometry, f.geometry)) "interarea"
   ,
      ST_Area(f.geometry) "area"
      from commune as c inner join filosofi as f on
      ST_Intersects(c.geometry, f.geometry)) as cf
group by cf."INSEE_COM"
```

Code 37 – Calcul d'une population communale à partir de données carroyées au prorata de la surface commune

3 Diffusion dynamique de l'information spatiale

La cartographie dynamique est le processus automatisé de construction, de représentation et de manipulation de cartes géographiques. Elle permet de créer des cartes interactives. Elle repose sur l'exploitation de flux mettant à disposition l'information spatiale en fonction des actions de l'utilisateur. Ces flux sont hébergés sur des serveurs cartographiques accessibles à l'aide de requêtes HTML. Les librairies Javascript de cartographies dynamiques telles que Leaflet ou Openlayer permettent de représenter directement ces flux. Il en est de même des logiciels dédiés tels que QGIS. Sous R et Python, l'exploitation des résultats de requêtes HTML permet de charger en mémoire sous forme de fond de carte les données contenues dans les flux.

Dans cette partie seront présentés les principaux flux géographiques et Geoserver l'un des serveurs de flux géographiques le plus utilisé actuellement.

3.1 Les flux géographiques

Parmi les nombreux types de flux existants, deux se distinguent par leur popularité : les **flux WMS (Web Map Service)** et les flux **WFS (Web Feature Service)**.

Les flux **WMS** envoient des images géoréférencées. Ces images ne contiennent pas d'informations en dehors des coordonnées où elles doivent s'afficher. Ce type de flux est adapté à la diffusion d'orthophotographies (images aériennes ou satellites de la surface terrestre) mais également de couches vectorielles, tel que le contour des communes par exemple. Dans ce cas, l'apparence des communes est déterminée par le serveur cartographique et ne peut être modifiée par l'utilisateur.

Les flux **WFS** renvoient des données vectorielles dans différents formats (json, csv, shapefile, gml). Les informations géométriques récupérées peuvent s'accompagner de nombreuses informations complémentaires. L'apparence des objets récupérés est modifiable par l'utilisateur.

3.1.1 Les flux WMS

Un flux WMS est une adresse url qui se termine par généralement par WMS. L'IGN propose par exemple deux adresses pour répondre à la directive Inspire d'accès aux données géographiques [17] : `https://wxs.ign.fr/inspire/inspire/r/WMS` pour accéder aux orthophotographies) et `https://wxs.ign.fr/inspire/`

17. `https://geoservices.ign.fr/services-web-inspire`

`inspire/v/WMS` pour récupérer sous forme d'images les bâtiments et les routes de France.

L'accès aux données s'obtient en complétant les adresses à l'aide d'un opérateur (à définir avec le mot clé *request*) et des paramètres. Le premier opérateur à connaître est *GetCapabilities* qui permet de décrire les données disponibles dans le flux. Le résultat de la requête `https://wxs.ign.fr/inspire/inspire/v/WMS?request=GetCapabilities` est un fichier XML (cf. fig 10).

FIGURE 10 – Contenu du flux IGN WMS

L'opérateur *GetMap* permet ensuite de récupérer les images d'un flux WMS. La requête suivante permet de récupérer la couche des bâtiments autour du batiment White, emplacement de la direction générale de l'Insee :

```
https://wxs.ign.fr/inspire/inspire/v/WMS?layer=BU.Building&
exception=text/xml&format=image/geotiff&
service=WMS&version=1.3.0&request=GetMap&styles=&crs=EPSG:4326&
bbox=48.8155,2.3072,48.8173,2.3099&width=256&height=256
```

Code 38 – Requête WMS pour récupérer une image

Ceci permet d'obtenir les images .png de la figure 11.

En R, la librairie *ows4R* permet d'accéder directement aux flux WFS et WMS sans avoir à écrire une requête et d'enregistrer l'image.

```
library(tiff)
url = 'https://wxs.ign.fr/inspire/inspire/v/WMS?LAYERS=BU.Building
    &
```

(a) 64 pixels (b) 128 pixels (c) 256 pixels

FIGURE 11 – Bâtiments autour de la Direction Générale de l'Insee à Montrouge suivant différentes définitions de l'image (paramètres *width* et *height*)

```
        EXCEPTIONS=text/xml&FORMAT=image/geotiff&SERVICE=WMS&
        VERSION=1.3.0&REQUEST=GetMap&STYLES=&CRS=EPSG:4326&
        BBOX=48.8155,2.3072,48.8173,2.3099&WIDTH=256&HEIGHT=256'
download.file(url, 'white.tiff')
white = readTIFF('white.tiff')
```

Code 39 – Téléchargement d'une image à partir d'une requête WMS en R

Sous Python, le téléchargement des images issues d'un flux WMS peut s'effectuer à l'aide de la librairie owslib (également valable pour les flux WFS). Tout comme sous R, il est également possible de procéder sans librairie *adhoc* en enregistrant directement les résultats d'une requête sous la forme d'une image.

```
import urllib.request

url = 'https://wxs.ign.fr/inspire/inspire/v/WMS?LAYERS=BU.Building
    &'\
       'EXCEPTIONS=text/xml&FORMAT=image/geotiff&'\
       'SERVICE=WMS&VERSION=1.3.0&REQUEST=GetMap&STYLES=&CRS=EPSG
    :4326&'\
       'BBOX=48.8155,2.3072,48.8173,2.3099&WIDTH=256&HEIGHT=256'
urllib.request.urlretrieve(url, 'white.tiff')
I = plt.imread('white.tiff')
```

Code 40 – Téléchargement d'une image à partir d'une requête WMS en Python

3.1.2 Les flux WFS

Tout comme les flux WMS, un flux WFS est une adresse HTML qui se termine par WFS et qui doit être également complétée par des opérateurs et des paramètres. L'opérateur *GetCapabilities* reste valable pour décrire les données disponibles dans le flux. En revanche, l'accès aux données s'effectue à l'aide de

l'opérateur *getfeature*. Par exemple, la requête suivante qui exploite un flux WFS proposé par l'IGN renvoie sous un format vectoriel les immeubles à proximité de la direction générale de l'Insee à Montrouge :

```
https://wxs.ign.fr/beta/geoportail/WFS?SERVICE=WFS&VERSION=2.0.0&
request=getfeature&typename=CADASTRALPARCELS.PARCELLAIRE_EXPRESS:
    batiment&
outputformat=application/json&bbox=48.8155,2.3072,48.8173,2.3099
```

Code 41 – Requête WFS pour récupérer les données vectorielles

Le paramètre *outputformat* définit le format de sortie de la couche vectorielle, soit ici le *json* (cf fig. 12). En dehors du *json*, les principaux formats de sortie sont le GML, une extension du language XML dédié à la description des objets géographiques et le format *shapefile* à l'aide de l'option *outputFormat=shape-zip* [18]. La couche est sélectionnée à l'aide du paramètre *typename*.

FIGURE 12 – Résultat de la requête *getfeature* précédente

Le contenu, l'emprise spatiale disponible et les formats de sortie d'une couche sont accessibles à l'aide d'une requête *DescribeFeatureType*. Ainsi la requête suivante décrit en autre chose, les colonnes contenues dans la table des bâtiments obtenus dans la requête précédente :

```
https://wxs.ign.fr/beta/geoportail/WFS?SERVICE=WFS&VERSION=2.0.0&
REQUEST=DescribeFeatureType&
typename=CADASTRALPARCELS.PARCELLAIRE_EXPRESS:batiment
```

```
req = 'https://wxs.ign.fr/beta/geoportail/WFS?SERVICE=WFS&
       VERSION=2.0.0&REQUEST=getfeature&
       typename=CADASTRALPARCELS.PARCELLAIRE_EXPRESS:batiment&
       outputformat=application/json&
```

18. En sortie, les différents fichiers composant le format *shapefile* sont compressés dans un fichier *zip*

```
        bbox=48.8155,2.3072,48.8173,2.3099'
download.file(url, 'white.geojson')
```

Code 42 – Traitement d'une requête WFS en R

Sous Python, tout comme pour les flux WMS, la commande *urllib.request* peut être utilisée pour enregistrer en local les résultats d'une requête WFS :

```
import urllib.request
url = 'https://wxs.ign.fr/beta/geoportail/WFS?SERVICE=WFS&'\
      'VERSION=2.0.0&REQUEST=getfeature&'\
      'typename=CADASTRALPARCELS.PARCELLAIRE_EXPRESS:batiment&'\
      'outputformat=application/json&'\
      'bbox=48.8155,2.3072,48.8173,2.3099'
urllib.request.urlretrieve(url, 'batiment.geojson')
```

Code 43 – Traitement d'une requête WFS en Python

3.1.3 Exemple d'application dynamique

Le code HTML suivant vise à illustrer l'intégration d'un fluw WFS dans une carte dynamique. La librairie de fonctions JavaScript utilisée est OpenLayer [19] qui permet une intégration native de ce type de flux contrairement à la libraire *Leaflet* [20]. De façon classique l'application se décompose en deux parties. Dans la première partie, les instructions servent à charger la librairie OpenLayer et à définir le style de la page. La deuxième partie écrite en JavaScript intègre le flux WFS dans le composant de cartographie dynamique d'OpenLayer.

```
<!DOCTYPE html>
<html>
<head>
    <title>Test WFS</title>
    <script src="https://cdn.jsdelivr.net/gh/openlayers/openlayers
    .github.io@master/en/v6.6.1/build/ol.js"></script>
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/gh/
    openlayers/openlayers.github.io@master/en/v6.6.1/css/ol.css">
  <style>
      .map {
        width: 100%;
        height:400px;
      }
    </style>
 </head>
<body>
  <div id="map" class="map"></div>
    <script>
```

19. https://openlayers.org/
20. En revanche, OpenLayer et Leaflet sont à même de traiter de façon native les flux WMS

```
17          var vector = new ol.layer.Vector({
18              //data sources
19              source: new ol.source.Vector({
20                  format: new ol.format.GeoJSON(),
21                  url: 'https://wxs.ign.fr/beta/geoportail/WFS?
    SERVICE=WFS&'+
22          'VERSION=2.0.0&REQUEST=getfeature&'+
23          'typename=CADASTRALPARCELS.PARCELLAIRE_EXPRESS:batiment&'+
24          'outputformat=application/json&bbox
    =48.8155,2.3072,48.8173,2.3099'
25              }),
26              //layer style
27              style: function(feature, resolution) {
28                  return new ol.style.Style({
29                      stroke: new ol.style.Stroke({
30                          color: 'blue'
31                      })
32                  });
33              }
34          });
35
36          var map = new ol.Map({
37              layers: [new ol.layer.Tile({source: new ol.source.OSM
    ()}), vector],
38              target: 'map',
39              view: new ol.View({center: [2.308,48.816], maxZoom:
    19,
40                  zoom: 16, projection: 'EPSG:4326'})
41          });
42    </script>
43 </body>
44 </html>
```

Code 44 – Cartographie dynamique d'un flux WFS à l'aide de la librairie OpenLayer

Pour s'initier en ligne à **Leaflet**, le lecteur peut se référer à la mini-application web proposée par Fabrice Garnes dont le code est accessible à cette adresse `https://codepen.io/fabcg/pen/wvWGQdW`. Cette application, dont le code (html, css, et JavaScript) est éditable, utilise Leaflet pour afficher des informations cartographiques dans un navigateur et récupère des fonds de carte sur les serveurs d'OpenStreetMap. Une couche image est récupérée à l'aide d'une requête WMS ainsi qu'une couche vectorielle à l'aide d'une requête WFS. Elle mobilise aussi le géocodeur [a] de l'IGN basé sur la Base Adresse Nationale. Il est également possible de dupliquer le code sur un projet personnel par exemple, sous codepen.io ou jsfiddle.net en créant un compte ce qui permet d'enregistrer les modifications.

a. L'objectif d'un géocodeur est d'associer à une adresse une position spatiale

3.2 GeoServer : diffuser des flux géographiques

Un serveur cartographique transforme une source de données spatiales, par exemple, un fichier *shapefile* ou une table spatiale contenue dans une base de données PostGIS en un flux interrogeable (WMS ou WFS) à l'aide d'une requête HTML. De nombreuses applications informatiques ont été développées pour réaliser cette tâche. Parmi celles-ci, GeoServer est l'une des plus plébiscitées par les cartographes du fait de sa polyvalence et de son interface graphique qui permet une administration simple des données sources et des flux en sortie.

La définition d'un flux s'obtient à l'aide de trois étapes :

1. Tout d'abord, il est nécessaire de définir un espace de travail. Un espace de travail permet d'organiser et de regrouper des flux similaires. Cet espace de travail se retrouve dans les requêtes WFS et WMS et correspond à la première partie des couches contenues dans les paramètres *layers* et *typename* des requêtes WMS et WFS. Par exemple, dans la requête WFS sur le bâti, CADASTRALPARCELS.PARCELLAIRE_EXPRESS est l'espace de travail qui contient la couche bâtiment.
2. Ensuite, un entrepôt de données, à savoir une source de données qui contient des données *raster* ou vecteurs, (*workspace*), doit être rattaché à l'espace de travail. Cet entrepôt peut être une base PostGIS ou un répertoire de travail contenant des fichiers *shapefile* par exemple.
3. La dernière étape consiste à sélectionner la couche de données spatiales contenue dans l'entrepôt de données qui doit être diffusé. Pour le format de sortie WMS, un style doit être défini et rajouté à la couche.

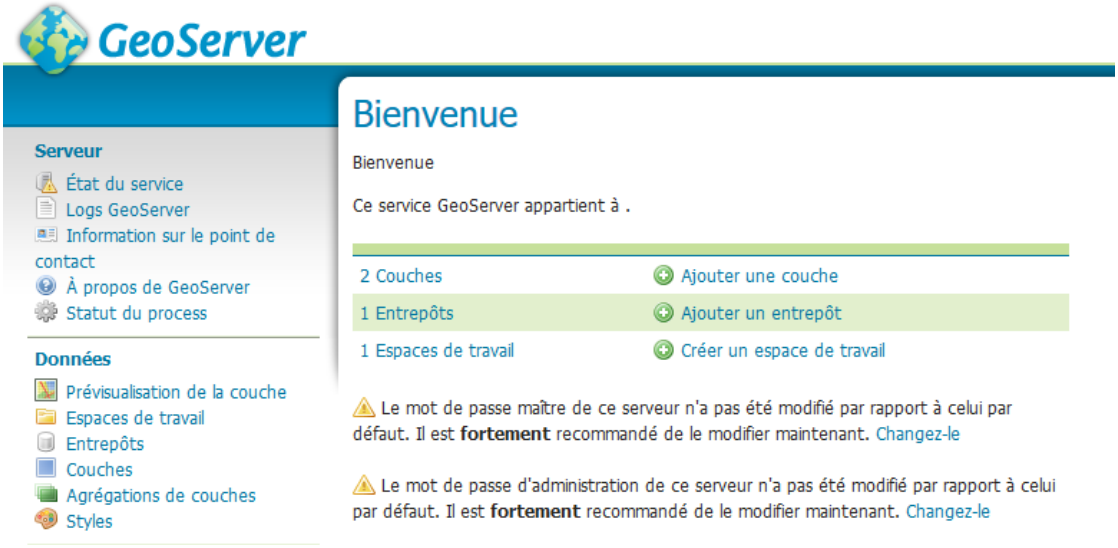


FIGURE 13 – Interface de GeoServer

Le menu Prévisualisation de la couche permet de tester les flux directement mis en ligne sans avoir à écrire une requête HTML.

L'interface de GeoServer permet d'apprendre ses principaux concepts et fonctionnalités. En revanche, le déploiement de nombreux flux y est fastidieux à réaliser à la main. GeoServer est accessible à l'aide d'une API pour automatiser la mise à disposition des flux. La librairie R geosapi permet d'accéder à cette API sans recourir à des lignes de commande. Il n'existe pas d'équivalent de cette librairie en Python. Les requêtes doivent être écrites explicitement et envoyées au GeoServer directement à l'aide de la librairie *request*. Un exemple est proposé dans ce post de blog.

3.3 Pour aller plus loin : les tuiles vectorielles

3.3.1 Le principe des tuiles vectorielles

Les flux WFS ne sont pas adaptés à la diffusion de couches géographiques vectorielles volumineuses. En effet, quelle que soit l'échelle d'affichage retenue, l'intégralité de la couche est téléchargée en local, ceci a pour conséquence de ralentir considérablement la réactivité des cartes dynamiques. Les outils de cartographie en ligne tels que Google Maps ou OpenStreetMap reposent sur une technologie adaptée aux données géographiques massives, à savoir les tuiles spatiales [21]. Les tuiles

21. L'emprise des tuiles s'observe dans Google Maps lorsque l'on change le zoom : l'information géographique s'affiche par bloc carré. Chaque bloc est une tuile.

ont été développées pour représenter des cartes évolutives en fonction de l'échelle cartographique et de l'emprise spatiale choisies. Cette technologie affiche uniquement les détails pertinents pour une échelle d'utilisation donnée : plus l'échelle d'observation est fine, plus les fonds de carte contenus dans les tuiles sont précis.

La figure 14 et le tableau 2 présentent l'organisation d'une tuile vectorielle. L'espace géographique est divisé en tuiles indexées par trois paramètres : l'échelle d'observation, la longitude et la latitude. Les tuiles sont caractérisées par les deux propriétés suivantes :

— pour une échelle donnée, les tuiles forment une partition du plan géographique ;
— les tuiles sont organisées selon un principe d'emboîtement d'échelle : les tuiles les plus fines sont strictement contenues dans les tuiles les plus grossières. Plus précisément, chaque tuile se décompose en quatre sous-tuiles à l'échelle plus fine suivante.

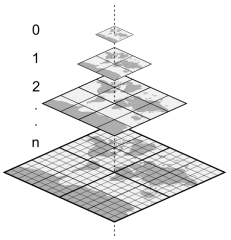


FIGURE 14 – Indexation des tuiles en fonction de l'échelle et de leur localisation. Source : Documentation QGIS

Niveau d'échelle	**Information visible**
0	La terre
3	Un continent
4	Une grande île
6	Une grande rivière
10	Une autoroute
15	Un bâtiment

TABLEAU 2 – Précision géographique en fonction de l'échelle d'observation. Source : Mapbox

En pratique, seule l'information contenue dans les tuiles intersectant l'emprise spatiale de la zone géographique d'observation est téléchargée ce qui réduit le temps d'affichage de la carte [22] (le choix de la zone d'observation définie implicitement l'échelle d'observation). L'accès à des données provenant de tuiles vectorielles dans une application web cartographique s'obtient à l'aide d'une adresse web du type *nom_ domaine/ z/ x/ y* ,où z est l'échelle (le zoom), x et y l'emplacement de la tuile.

La société Mapbox, spécialisée en cartographie en ligne, propose le format de

22. Sous Google Maps, l'échelle d'observation est le dernier chiffre de l'adresse url. L'adresse `www.google.fr/maps/@48.8565926,2.3533182,7z` produit une carte centrée sur Paris et recouvrant le grand bassin parisien. L'échelle de restitution est égale à 7. En augmentant celle-ci, par exemple en passant de 7 à 12, l'emprise spatiale se réduit et la précision de l'information géographique augmente. La carte recouvre alors la commune de Paris et sa proche périphérie

stockage *MBtiles* des tuiles vectorielles. Un fichier *MBtiles*[23] est une base de données SQLite dont les enregistrements sont des tuiles. Il s'obtient en deux étapes :

— pour une plage d'échelles donnée, les couches géographiques qui constitueront la tuile vectorielle sont découpées dans les tuiles indépendamment les unes des autres.

— Les fonds de carte contenus dans des tuiles d'échelle et de localisation identiques sont regroupés ensemble. A la fin du processus, chaque tuile contient un ensemble de couches.

L'information géographique contenue dans les tuiles vectorielles est délivrée dans un flux cartographique à l'aide d'un serveur *adhoc*. Mapbox propose d'héberger gratuitement des tuiles vectorielles pour des projets expérimentaux. Pour les projets plus conséquents, des solutions payantes sont proposées.

3.3.2 Mise en oeuvre avec Tegola

Depuis quelques années des serveurs Open Source compatibles avec les normes de Mapbox ont été développés. Parmi les différentes solutions disponibles, Tegola [24] se distingue par ses performances et sa simplicité d'utilisation [25]. Contrairement à l'approche proposée par Mapbox, cette solution ne nécessite pas de découper *a priori* les fonds de carte pour les intégrer dans les tuiles. L'information est stockée dans une base PostGIS et les tuiles sont constituées à la volée en fonction des demandes des utilisateurs. Ceci est rendu possible dans PostGIS grâce à l'instruction *ST_AsMVT* qui permet de découper une couche dans des tuiles. Les flux sont définis à l'aide d'un fichier de paramètres, **.toml*, dont un exemple est donné ci-dessous. L'objectif ici est de diffuser trois couches contenues dans un serveur PostGIS sur des plages d'échelles ne se recoupant pas (département, commune et iris).

```
[webserver]
port = ":8080"

[[providers]]
name = "defaultdb"  #nom de la base
type = "postgis"
host = *****      #PostGIS database host
port = 5432     #le port de la base de donn e
database = "defaultdb"       #nom de la base de donn es
user = ****     #l'identifiant de l'utilisateur
password = ****     #le mot de passe du serveur PostGIS
srid = 4326 # la projection
```

23. La norme des fichiers *MBtiles* est définie dans le repo github de Mapbox `https://github.com/mapbox/mbtiles-spec`

24. `https://tegola.io/`

25. Il suffit en effet, de télécharger et d'extraire l'exécutable pour utiliser Tegola.

```
13
14   [[providers.layers]]
15   name = "departement"
16   geometry_fieldname = "geometry"
17   id_fieldname = "dep"
18   sql = "SELECT ST_AsBinary(ST_MULTI(geometry)) AS geometry, dep
      FROM departement WHERE geometry && !BBOX!"
19
20   [[providers.layers]]
21   name = "commune"
22   geometry_fieldname = "geometry"
23   id_fieldname = "commune"
24   sql = "SELECT ST_AsBinary(st_multi(geometry)) AS geometry,
      commune FROM commune WHERE geometry && !BBOX!"
25
26   [[providers.layers]]
27   name = "iris"
28   geometry_fieldname = "geometry"
29   id_fieldname = "iris"
30   sql = "SELECT ST_AsBinary(st_multi(geometry)) AS geometry, iris
      FROM iris WHERE geometry && !BBOX!"
31
32 [[maps]]
33 name = "paris"
34 center = [2.3626, 48.8744, 7.0] # carte centr e sur Paris
35
36   [[maps.layers]]
37   provider_layer = "defaultdb.departement"
38   min_zoom = 6
39   max_zoom = 8
40
41   [[maps.layers]]
42   provider_layer = "defaultdb.commune"
43   min_zoom = 9
44   max_zoom = 12
45
46   [[maps.layers]]
47   provider_layer = "defaultdb.iris"
48   min_zoom = 13
49   max_zoom = 18
```

[[providers]] sert à établir la connexion avec le serveur PostGIS. *[[providers.layers]]* définit les couches utilisées. En particulier, l'instruction *sql* est utilisée pour écrire la requête qui permet de sélectionner l'information contenue dans les couches. *[[maps]]* sert à paramétrer l'échelle et la localisation initiale et *[[maps.layers]]* les échelles d'affichage des couches.

Le lancement du serveur Tegola s'effectue avec la ligne de commande suivante, le serveur étant lancé en local :

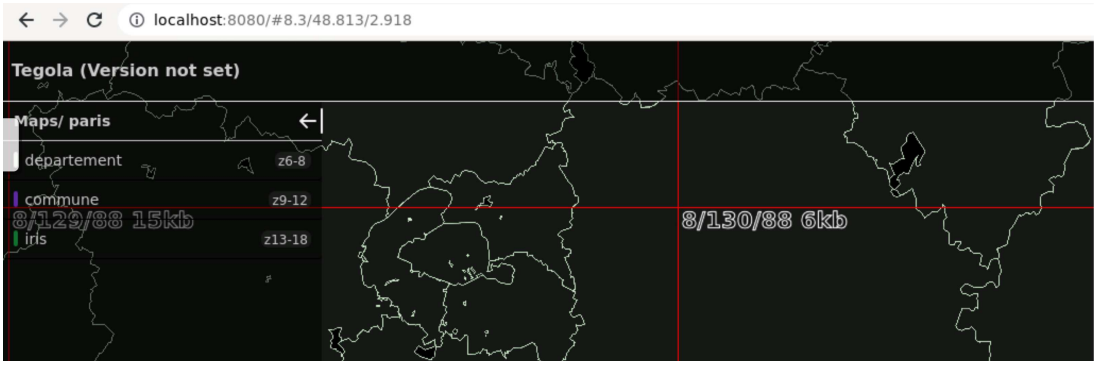
```
./tegola serve --config=config.toml
```

Tegola dispose d'une application de prévisualisation des tuiles dans un navigateur internet à l'adresse *localhost :8080* (fig 15).

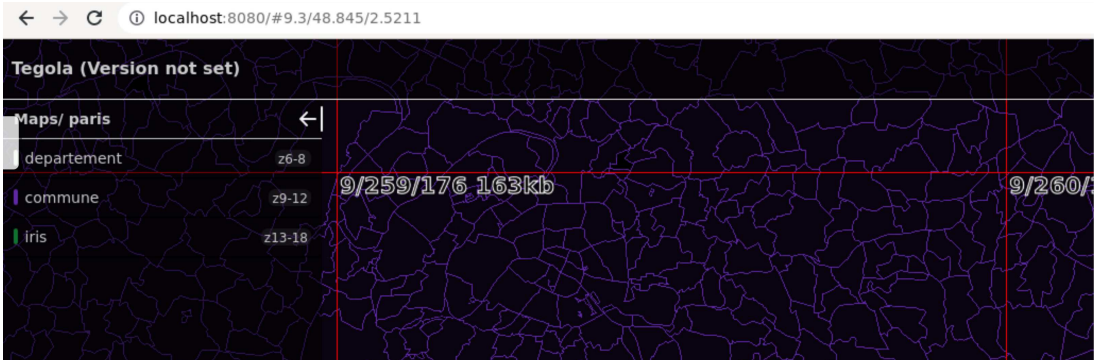
L'affichage d'une carte choroplèthe dynamique à l'aide de tuiles vectorielles peut être réalisée directement à l'aide de la librairie mapbox.gl [26] et OpenLayers.

Sous Leaflet, il faut rajouter les plugins Leaflet.VectorGrid.

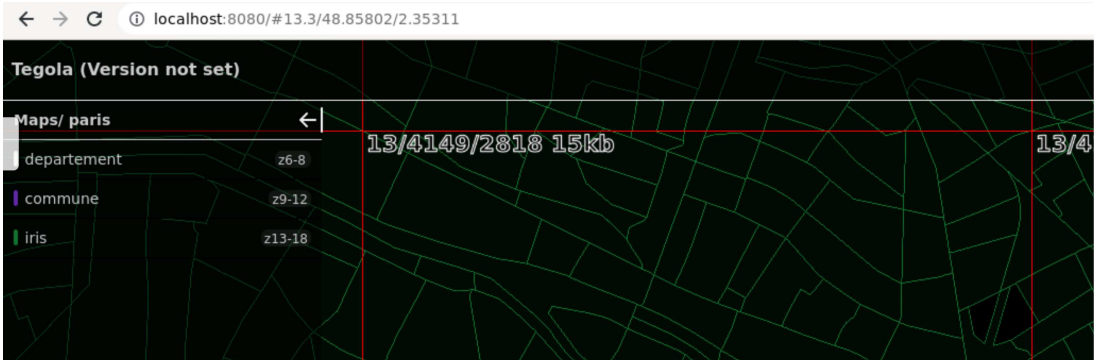
26. https://docs.mapbox.com/mapbox-gl-js/example/updating-choropleth/



(a) Affichage des départements



(b) Affichage des communes



(c) Affichage des iris

Note : les lignes rouges représentent l'emprise des tuiles. Les numéros dans les tuiles désignent leur indices : l'échelle, la longitude, la latitude.

FIGURE 15 – Application de prévisualisation des tuiles vectorielles de Tegola.

Discussion : le territoire des statisticiens

Construire une statistique territoriale revient à associer à un territoire ou à une portion de l'espace, la mesure statistique d'un phénomène. Cette opération est une abstraction géographique qui ne va pas de soit et mérite d'être questionnée. Dans cette dernière partie, en guise de conclusion, on rappelle les questions que les statisticiens doivent se poser avant de faire une carte ainsi que les risques d'erreurs classiques ou de biais perceptifs propres à la cartographie thématique à garder en mémoire dès qu'on utilise des données spatiales.

Quels messages passent mieux par une carte ? Toute quantité numérique peut être représentée sous forme d'une carte. En pratique, une carte n'a d'intérêt que si elle illustre la dimension spatiale d'un phénomène socio-économique et qu'elle aide à sa compréhension. Une carte doit être structurée pour que le message qu'elle diffuse soit lisible (par exemple opposition nord-sud comme sur la carte de Dupin). En géographie, cette structuration de l'espace peut se formaliser sous la forme de « lois »géographiques. Par exemple, pour décrire et expliquer les concentrations et dispersions des implantations humaines, Tannier (2017) propose cinq lois :

- Friction de la distance : des éléments proches ont davantage de chance d'être en relation (pour des processus) ou d'être similaires (états) que des éléments éloignés car la friction de la distance atténue l'influence mutuelle des phénomènes au fur et à mesure.
- Rôle de l'appartenance à un même milieu : des éléments d'un même milieu ont davantage de chances d'être en relation (ou de se ressembler) que des éléments de milieux différents. Un même milieu peut être autant une même maille administrative (commune, région...), une même nation, les mêmes conditions climatiques, une même langue, un même groupe social...
- Influence de la taille des entités considérées : des entités de grande taille, ayant une large emprise spatiale ou contenant une quantité importante d'individus, d'activités, d'information..., ont davantage d'influence sur les éléments de leur environnement que des entités de petite taille.
- Dépendance d'échelles : pour un type donné d'implantations humaines, l'existence d'un processus de concentration considérant un certain voisinage implique un processus de dispersion considérant un voisinage plus large (ou plus retreint).
- Dépendance dans le temps : chaque phénomène influence sa propre variation dans le temps.

Ces lois sont utiles car elles permettent de soutenir les réflexions sur le rôle de l'espace dans la production des comportements sociaux. Les garder en mémoire est utile pour les statisticiens pour questionner les messages transmis par les cartes.

Le « piège territorial » Les territoires (nationaux, régionaux, communaux, locaux ou les carreaux d'un carroyage), appelés également unités spatiales, occupent une place centrale dans l'action publique. Outils de gouvernance, ils sont utilisés pour adapter les politiques publiques aux contextes locaux. Les cartes thématiques réalisées par les statisticiens participent à l'appropriation par les acteurs publics de la connaissance de leurs territoires. Il n'en reste pas moins que le territoire du statisticien n'est pas celui de l'acteur public. John Agnew dénomme « piège territorial »l'erreur consistant à confondre les deux, voir Agnew et Dufoix (2014). Le piège territorial selon John Agnew s'incarne dans trois préjugés géographiques :

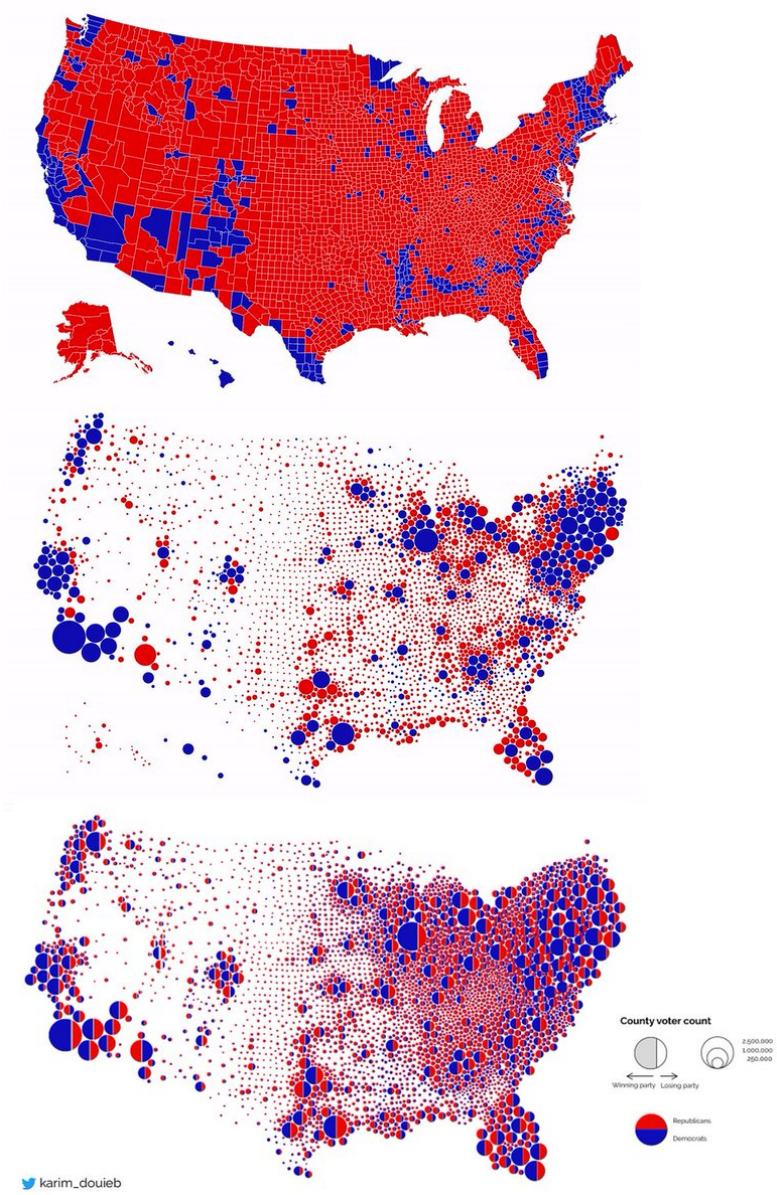
— la réification des espaces territoriaux de l'État en unités figées d'espace souverain sûr ;
— opposition entre l'intérieur et l'extérieur ;
— supposer que l'État territorial préexiste à la société et qu'il en est le conteneur alors même que les populations se déplacent dans l'espace et ont plusieurs lieux d'encrage.

Les cartes thématiques reposant sur des découpages territoriaux sont susceptibles de participer à la reproduction de ce piège territorial. La cartographie des résultats de l'élection présidentielle américaine de 2016 illustre la façon dont les cartes peuvent façonner différents discours territoriaux, voir Figure 3.3.2. La première carte du vote majoritaire par comté est une analyse en aplat de couleurs avec uniquement deux teintes séparées par la borne 50% (carte de taux ou carte de ratio). Elle suggère une victoire écrasante du candidat républicain au détriment de la candidate démocrate. Les deux autres cartes nuancent ce constat : les territoires peu denses ont voté majoritairement républicain mais contribuent peu en valeur absolue à la géographie du vote. Ce constat a fait dire à l'auteur de ces cartes, Karim Douieb, que « les territoires ne votent pas. Les gens, oui ».

Faut-il pour autant bannir la carte de ratio des analyses statistiques ? Dans le cas des États-Unis, celle-ci est porteuse d'une information pertinente, car le système électoral privilégie une base territoriale à travers le système des grands électeurs contrairement à un suffrage majoritaire qui ne tient pas compte de l'inégale répartition spatiale des voix. La carte de ratio permet de comprendre les résultats de l'élection, la victoire du candidat républicain. En revanche, les cartes en volume relativisent cette victoire et permettent d'affirmer que l'élection ne fut pas plébiscite pour le vainqueur [27].

En France, la majorité des personnes défavorisées sont concentrées en volume dans les centres urbains (fig 17). Ces sous-populations sont donc inégalement représentées dans l'espace. Très présentes dans les banlieues des grands pôles, elles peuvent aussi être ségrégées. Les politiques de mixité sociale et d'actions sur les

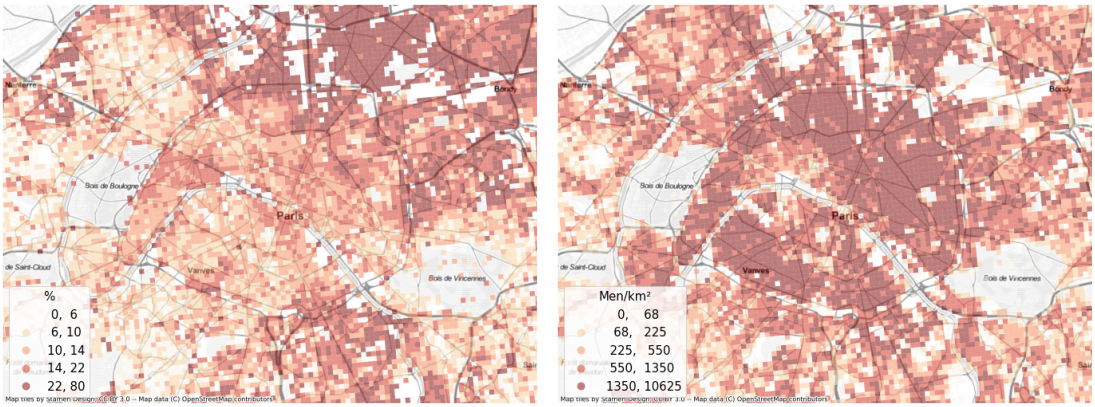
27. Ceci est d'autant plus vrai que la candidate démocrate avait plus de voix en volume que le candidat républicain.



Note : trois manières de visualiser les résultats de l'élection américaine de 2016 (en rouge le vote républicain et en bleu le vote démocrate) : en haut, le gagnant par comté, au milieu rapporté à la population plutôt qu'à la surface du comté et en bas la répartition des votes rapportée à la taille de la population par comté. Source : Karim Douïeb, sur Twitter.

FIGURE 16 – "Land doesn't vote, people do."

quartiers de la politique de la ville (dont le découpage correspond à des zones de concentration de ménages en situation de pauvreté) ont vocation à agir sur les effets de la ségrégation sur les populations concernées. Le piège territorial dans ce cas est de considérer que seuls les territoires de concentration décrivent l'organisation des populations défavorisées. Les cartes de volume nuancent ce constat et sont utiles pour les politiques de droit commun qui visent à dimensionner par exemple des équipements dans les centres-villes à destination des populations défavorisées.



(a) Part des ménages pauvres parmi les ménages (b) Densité de ménages pauvres par km²

FIGURE 17 – Deux visions d'un même phénomène : répartition des ménages pauvres dans l'agglomération parisienne. Source : Filosofi 2015

Opposer les représentations en volume et en part n'est pas pertinent, les deux apportent une connaissance différente sur un même phénomène social. Les outils de visualisation dynamique modernes permettent de changer les points de vue en passant d'une représentation à l'autre et ainsi de dépasser le piège territorial.

L'erreur écologique ou *Modifiable Areal Unit Problem* en géographie
Même si les données ont une précision métrique, la production d'un savoir géographique nécessite de les relier entre elles à l'aide de règles spatiales, autrement dit de spatialiser les données. La spatialisation consiste à associer à chaque objet une portion, une quantité d'espace bien défini. Cette mise en relation spatiale s'apparente à un point de vue que l'on se donne afin de rendre intelligible les distributions spatiales. Ce point de vue opère comme un filtre qui met en valeur certaines propriétés spatiales et en délaisse d'autres. Il en résulte que les analyses géographiques ont donc toujours une part d'arbitraire et sont fonction de la spatialisation choisie.

Les données spatiales sont ainsi sensibles aux **effets d'agrégation** appelés

erreur écologique en sciences sociales (Courgeau, 2000) et ***Modifiable Areal Unit Problem*** en géographie (Openshaw et Pumain, 1981).

L'erreur écologique consiste à inférer des relations statistiques à l'échelle des individus à partir de données agrégées, (Wikipédia, 2021a). [28]

Openshaw (1983) a formalisé sous le concept de *Modifiable Areal Unit Problem* (MAUP) les effets d'agrégation propres à la géographie. Le MAUP se décompose en deux effets interdépendants : l'**effet de zonage** et l'effet d'échelle. Le premier est lié à la forme des unités spatiales élémentaires [29] tandis que le deuxième désigne la dépendance des analyses géographiques et statistiques à l'échelle d'observation.

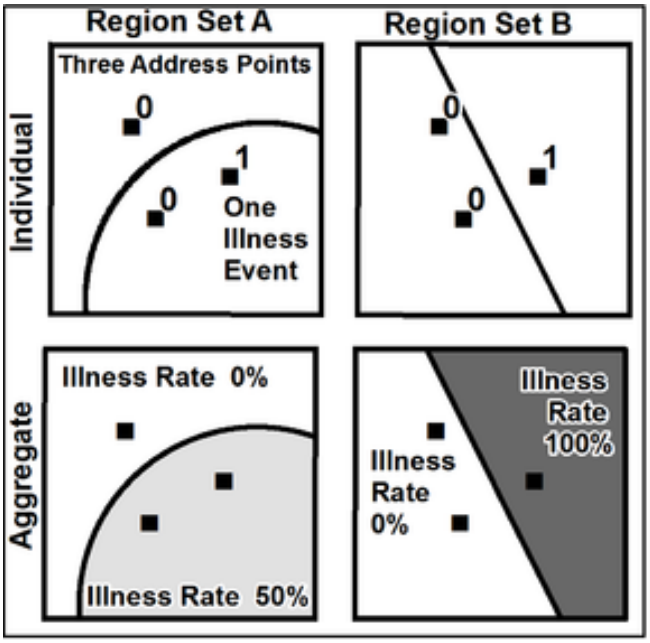
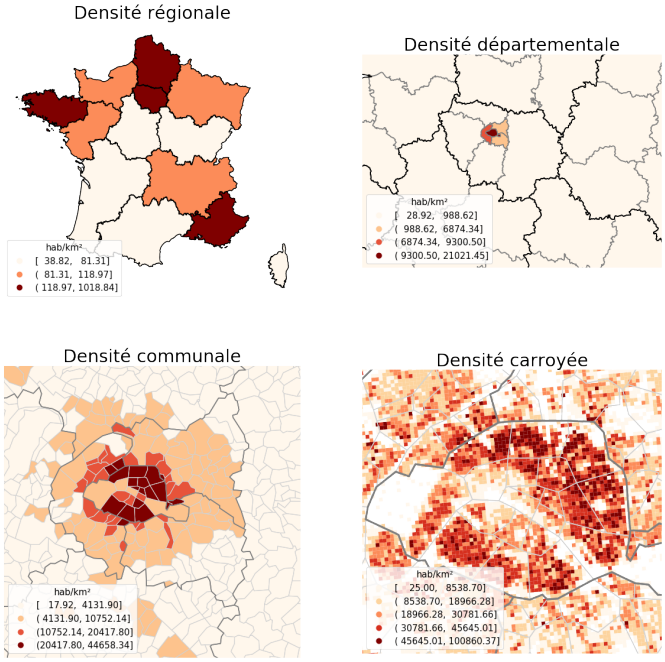


FIGURE 18 – MAUP : effet de zonage

Dans les analyses statistiques, on peut se prémunir en partie des effets de zonage

28. L'analyse de Durkheim sur la prévalence des suicides selon la confession religieuse à l'échelle des provinces de la Prusse est souvent citée comme un exemple manifeste d'erreur écologique, voir Courgeau (2000). Durkeim extrapole un taux de suicide pour les protestants quatre fois supérieur à celui observé sur données individuelles en s'appuyant sur la relation presque linéaire qu'il existe entre taux de suicide et part des protestants dans les provinces de Prusse. Or cette relation reflète surtout des variations de conditions de vie (corrélées positivement avec la part de protestants et explicatives du taux de suicide). Il s'agit là d'un effet de structure, appelé aussi effet de composition, voir Courgeau (2000).

29. Le redécoupage des circonscriptions électorales par les élus est un exemple classique d'effet de zonage (fig 18) . Aux États-Unis cette pratique porte le nom de *gerrymandering* en référence à Elbridge Gerry qui fut accusé d'avoir redécoupé la circonscription d'un comté afin de favoriser son parti (Wikipédia, 2021b).



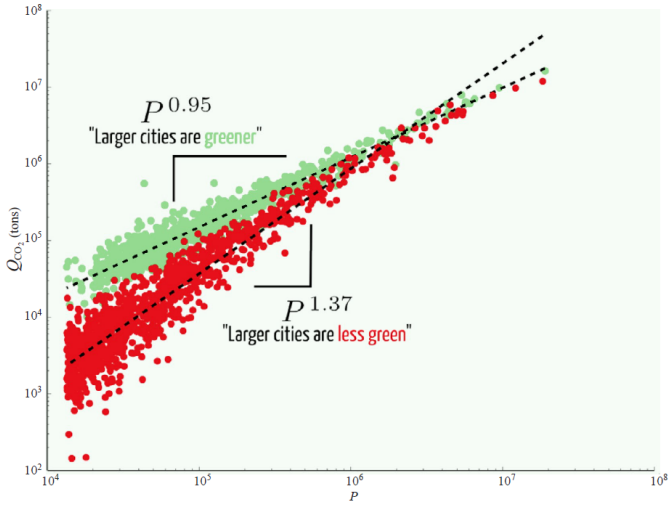
Note : densités de population à 4 échelles différentes. Source : Recensement de la population 2017 et Filosofi 2015.

FIGURE 19 – MAUP : effet d'échelle.

en recourant à des découpages réguliers tels que des carroyages ou bien encore en réalisant des lissages spatiaux. En revanche, on ne peut quasiment rien faire quant aux effets d'échelle - le choix du rayon dans la définition de la concentration d'un ensemble est un effet d'échelle - ils sont inhérents à la prise en compte de l'espace dans les analyses statistiques. La figure 19 témoigne des effets d'échelle sur un indicateur densité. Elle montre comment la variabilité spatiale des densités augmente avec la finesse de l'échelle d'analyse.

Les effets d'échelle du MAUP ne sont pas uniquement cartographiques et concernent également les modélisations statistiques. Louf et Barthelemy (2014) explorent le lien existant entre la population des villes et la production de CO_2 (fig **??**). Ils constatent qu'en fonction de la définition territoriale des villes américaines choisie, les villes les plus vertueuses ne sont plus les mêmes. Avec le découpage en aire urbaine (définition fonctionnelle), plus les villes sont petites et plus elles sont vertes alors que la relation s'inverse avec le découpage en unité urbaine (définition morphologique). Ils concluent qu'en l'absence d'un modèle sous-jacent, il est dangereux d'interpréter les résultats d'une analyse de données.

Jusqu'à récemment, le MAUP était considéré comme une limite à la connais-



En rouge, analyse avec le découpage en unité urbaine et en vert avec le découpage en aire urbaine. Source : Louf et Barthelemy (2014)

FIGURE 20 – *Are larger cities greener or smoggier ?* Relation entre l'émission de CO_2 et la taille des villes aux Etats-Unis.

sance géographique. Dans le meilleur des cas, les statisticiens et les géographes le résolvaient en utilisant l'échelle d'action des acteurs publics (mais est-elle la plus pertinente ?) ou en recherchant une échelle caractéristique au phénomène étudié (mais sur quel critère la déterminer ?). A présent, le MAUP est pensé comme une source d'information sur l'organisation spatiale des distributions étudiées. Plus précisément, la variation des indicateurs à travers les échelles dévoile le caractère multiscalaire des distributions spatiales étudiées (Grasland et Madelin, 2006). Pour comprendre l'organisation d'une distribution spatiale, on doit nécessairement l'observer à différentes échelles. L'usage de tuiles vectorielles permet dans les cartes dynamiques de faire comprendre le rôle des échelles sur l'observation des phénomènes par les utilisateurs.

Références

Agnew, J., et S. Dufoix (2014) : "Le piège territorial," *Raisons politiques*, (2), 23–51.

Baëhr, A., et M. Courthial (2020) : "Le commerce de centre-ville recule dans les villes moyennes," *Insee Flash Hauts-de-France*, (91).

Bentley, J. L. (1975) : "Multidimensional Binary Search Trees Used for Associative Searching," *Commun. ACM*, 18(9), 509–517.

Courgeau, D. (2000) : "Réflexions sur la causalité en sciences sociales," *Revue des politiques sociales et familiales*, 60(1), 49–60.

Entin, M. (2019) : "Contains/Covers/Intersects/Within ?," `https://mentin.medium.com/which-predicate-cb608b470471`, Medium Post. Accessed : 2022-01-19.

Ester, M., H.-P. Kriegel, J. Sander, et X. Xu (1996) : "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, p. 226–231. AAAI Press.

Giraud, T., et H. Pecout (2021) : "Cartographie avec R," `https://rcarto.github.io/carto_avec_r/`, Accessed : 2021-08-16.

Grasland, C., et M. Madelin (2006) : "Espon 3.4. 3 The Modifiable Areas Unit Problem. Final Report, 254 p," .

Himpens, S., M. Poulhes, et F. Sémécurbe (2021) : "Bâti dispersé, bâti concentré, des disparités territoriales persistantes," *Insee Analyses*, (63).

Lambert, N., et C. Zanin (2016) : *Manuel de cartographie : principes, méthodes, applications*. Armand Colin.

Le Bras, H., et E. Todd (2013) : *Le mystère français*. Seuil Paris.

Loonis, V., et M.-P. Bellefon (2018) : *Manuel d'analyse spatiale. Théorie et mise en œuvre pratique avec R* Insee et Eurostat.

Louf, R., et M. Barthelemy (2014) : "Scaling : lost in the smog," *Environment and Planning B : Planning and Design*, 41(5), 767–769.

O'Brien, R. (2020) : "How To Load Your Pandas DataFrame To Your Database 10x Faster," https://towardsdatascience.com/upload-your-pandas-dataframe-to-your-database-10x-faster-eb6dc6609ddf, Accessed : 2021-08-16.

Openshaw, S. (1983) : "The modifiable areal unit problem (Vol. 38)," *Norwich : Geo Books*.

Openshaw, S., et D. Pumain (1981) : "Le problème de l'agrégation spatiale en géographie," *L'Espace géographique*, 10(1), 15–24.

Palsky, G. (1990) : "La cartographie thématique en France : recherches sur ses origines et son évolution jusqu'à la fin du 19è siècle," Ph.D. thesis, Paris 1.

——— (1991) : "La cartographie statistique de la population au XIXe siècle," *Espace Populations Sociétés*, 9(3), 451–458.

——— (1996) : *Des chiffres et des cartes : naissance et développement de la cartographie quantitative française au XIXe siècle*. Comité des travaux historiques et scientifiques Paris.

Pebesma, E. (2021) : "Simple Features for R," https://r-spatial.github.io/sf/, Accessed : 2021-08-16.

Playfair, W. (1805) : *An inquiry into the permanent causes of the decline and fall of powerful and wealthy nations*. Books4x Company.

PostGIS (2021) : "Postgis," https://postgis.net/, Accessed : 2021-08-16.

Tannier, C. (2017) : "Analyse et simulation de la concentration et de la dispersion des implantations humaines de l'échelle micro-locale à l'échelle régionale - Modèles multi-échelles et trans-échelles," Habilitation à diriger des recherches, Université Bourgogne Franche-Comté.

Wikipédia (2021a) : "Erreur écologique — Wikipédia, l'encyclopédie libre," En ligne ; Page disponible le 24-août-2021.

——— (2021b) : "Gerrymandering — Wikipédia, l'encyclopédie libre," [En ligne ; Page disponible le 30-novembre-2021].

——— (2021c) : "Système d'information géographique — Wikipédia, l'encyclopédie libre," .

——— (2022) : "Port (logiciel) — Wikipédia, l'encyclopédie libre," [En ligne ; Page disponible le 1-janvier-2022].

Série des Documents de Travail « Méthodologie Statistique »

9601 : Une méthode synthétique, robuste et efficace pour réaliser des estimations locales de population.
G. DECAUDIN, J.-C. LABAT

9602 : Estimation de la précision d'un solde dans les enquêtes de conjoncture auprès des entreprises.
N. CARON, P. RAVALET, O. SAUTORY

9603 : La procédure FREQ de SAS - Tests d'indépendance et mesures d'association dans un tableau de contingence.
J. CONFAIS, Y. GRELET, M. LE GUEN

9604 : Les principales techniques de correction de la non-réponse et les modèles associés.
N. CARON

9605 : L'estimation du taux d'évolution des dépenses d'équipement dans l'enquête de conjoncture : analyse et voies d'amélioration.
P. RAVALET

9606 : L'économétrie et l'étude des comportements. Présentation et mise en œuvre de modèles de régression qualitatifs. Les modèles univariés à résidus logistiques ou normaux (LOGIT, PROBIT).
S. LOLLIVIER, M. MARPSAT, D. VERGER

9607 : Enquêtes régionales sur les déplacements des ménages : l'expérience de Rhône-Alpes.
N. CARON, D. LE BLANC

9701 : Une bonne petite enquête vaut-elle mieux qu'un mauvais recensement ?
J.-C. DEVILLE

9702 : Modèles univariés et modèles de durée sur données individuelles.
S. LOLLIVIER

9703 : Comparaison de deux estimateurs par le ratio stratifiés et application aux enquêtes auprès des entreprises.
N. CARON, J.-C. DEVILLE

9704 : La faisabilité d'une enquête auprès des ménages.
1. au mois d'août.
2. à un rythme hebdomadaire

C. LAGARENNE, C. THIESSET

9705 : Méthodologie de l'enquête sur les déplacements dans l'agglomération toulousaine.
P. GIRARD.

9801 : Les logiciels de désaisonnalisation TRAMO & SEATS : philosophie, principes et mise en œuvre sous SAS.
K. ATTAL-TOUBERT, D. LADIRAY

9802 : Estimation de variance pour des statistiques complexes : technique des résidus et de linéarisation.
J.-C. DEVILLE

9803 : Pour essayer d'en finir avec l'individu Kish.
J.-C. DEVILLE

9804 : Une nouvelle (encore une !) méthode de tirage à probabilités inégales.
J.-C. DEVILLE

9805 : Variance et estimation de variance en cas d'erreurs de mesure non corrélées ou de l'intrusion d'un individu Kish.
J.-C. DEVILLE

9806 : Estimation de précision de données issues d'enquêtes : document méthodologique sur le logiciel POULPE.
N. CARON, J.-C. DEVILLE, O. SAUTORY

9807 : Estimation de données régionales à l'aide de techniques d'analyse multidimentionnelle.
K. ATTAL-TOUBERT, O. SAUTORY

9808 : Matrices de mobilité et calcul de la précision associée.
N. CARON, C. CHAMBAZ

9809 : Échantillonnage et stratification : une étude empirique des gains de précision.
J. LE GUENNEC

9810 : Le Kish : les problèmes de réalisation du tirage et de son extrapolation.
C. BERTHIER, N. CARON, B. NEROS

9901 : Perte de précision liée au tirage d'un ou plusieurs individus Kish.
N. CARON

9902 : Estimation de variance en présence de données imputées : un exemple à partir de l'enquête Panel Européen.
N. CARON

0001 : L'économétrie et l'étude des comportements. Présentation et mise en œuvre de modèles de régression qualitatifs. Les modèles univariés à résidus logistiques ou normaux (LOGIT, PROBIT) (version actualisée).
S. LOLLIVIER, M. MARPSAT, D. VERGER

0002 : Modèles structurels et variables explicatives endogènes.
J.-M. ROBIN

0003 : L'enquête 1997-1998 sur le devenir des personnes sorties du RMI - Une présentation de son déroulement.
D. ENEAU, D. GUILLEMOT

0004 : Plus d'amis, plus proches ? Essai de comparaison de deux enquêtes peu comparables.
O. GODECHOT

0005 : Estimation dans les enquêtes répétées : application à l'Enquête Emploi en Continu.
N. CARON, P. RAVALET

0006 : Non-parametric approach to the cost-of-living index.
F. MAGNIEN, J. POUGNARD

0101 : Diverses macros SAS : Analyse exploratoire des données, Analyse des séries temporelles.
D. LADIRAY

0102 : Économétrie linéaire des panels : une introduction.
T. MAGNAC

0201 : Application des méthodes de calages à l'enquête EAE-Commerce.
N. CARON

C 0201 : Comportement face au risque et à l'avenir et accumulation patrimoniale - Bilan d'une expérimentation.
L. ARRONDEL, A. MASSON, D. VERGER

C 0202 : Enquête Méthodologique Information et Vie Quotidienne - Tome 1 : bilan du test 1, novembre 2002.
J.-A. VALLET, G. BONNET, J.-C. EMIN, J. LEVASSEUR, T. ROCHER, P. VRIGNAUD, X. D'HAULTFOEUILLE, F. MURAT, D. VERGER, P. ZAMORA

0203 : General principles for data editing in business surveys and how to optimise it.
P. RIVIERE

0301 : Les modèles logit polytomiques non ordonnés : théories et applications.
C. AFSA ESSAFI

0401 : Enquête sur le patrimoine des ménages - Synthèse des entretiens monographiques.
V. COHEN, C. DEMMER

0402 : La macro SAS CUBE d'échantillonnage équilibré
S. ROUSSEAU, F. TARDIEU

0501 : Correction de la non-réponse et calage de l'enquêtes Santé 2002
N. CARON, S. ROUSSEAU

0502 : Correction de la non-réponse par répondération et par imputation
N. CARON

0503 : Introduction à la pratique des indices statistiques - notes de cours
J-P BERTHIER

0601 : La difficile mesure des pratiques dans le domaine du sport et de la culture - bilan d'une opération méthodologique
C. LANDRE, D. VERGER

0801 : Rapport du groupe de réflexion sur la qualité des enquêtes auprès des ménages
D. VERGER

M2013/01 : La régression quantile en pratique
P. GIVORD,
X. D'HAULTFOEUILLE

M2014/01 : La microsimulation dynamique : principes généraux et exemples en langage R
D. BLANCHET

M2015/01 : la collecte multimode et le paradigme de l'erreur d'enquête totale
T. RAZAFINDROVONA

M2015/02 : Les méthodes de Pseudo-Panel
M. GUILLERM

M2015/03 : Les méthodes d'estimation de la précision pour les enquêtes ménages de l'Insee tirées dans Octopusse
E. GROS
K. MOUSSALAM

M2016/01 : Le modèle Logit Théorie et application.
C. AFSA

M2016/02 : Les méthodes d'estimation de la précision de l'Enquête Emploi en Continu
E. GROS
K.MOUSSALAM

M2016/03 : Exploitation de l'enquête expérimentale Vols, violence et sécurité.
T. RAZAFINDROVONA

M2016/04 : Savoir compter, savoir coder. Bonnes pratiques du statisticien en programmation.
E. L'HOUR
R. LE SAOUT
B. ROUPPERT

M2016/05 : Les modèles multiniveaux
P. GIVORD
M. GUILLERM

M2016/06 : Econométrie spatiale : une introduction pratique
P. GIVORD
R. LE SAOUT

M2016/07 : La gestion de la confidentialité pour les données individuelles
M. BERGEAT

M2016/08 : Exploitation de l'enquête expérimentale Logement internet-papier
T. RAZAFINDROVONA

M2017/01: Exploitation de l'enquête expérimentale Qualité de vie au travail
T. RAZAFINDROVONA

M2018/01: Estimation avec le score de propension sous R
S. QUANTIN

M2018/02: Modèles semi-paramétriques de survie en temps continu sous R
S. QUANTIN

M2019/01 : Les méthodes de décomposition appliquées à l'analyse des inégalités
B. BOUTCHENIK
E. COUDIN

S. MAILLARD

M2020/01 : L'économétrie en grande dimension
J. L'HOUR

M2021/01 : R Tools for JDemetra+ - Seasonal adjustment made easier
A. SMYK
A. TCHANG

M2021/02 : Le traitement du biais de sélection endogène dans les enquêtes auprès des ménages par modèle de Heckman
L. CASTELL
P. SILLARD

M2021/03 : Conception de questionnaires auto-administrés
H. KOUMARIANOS
A. SCHREIBER

M2022/01 : Introduction à la géomatique pour le statisticien : quelques concepts et outils innovants de gestion, traitement et diffusion de l'information spatiale
F. SEMECURBE
E. COUDIN