ESPON

EUROPEAN UNION

Co-financed by the European Regional Development Fund

Inspire policy making by territorial evidence

ESPON Big Data for Territorial Analysis and Housing Dynamics

Wellbeing of European citizens regarding the affordability of housing.

Monitoring and tools

Technical Guidance Document

Final Report

This monitoring and tools activity is conducted within the framework of the ESPON 2020 Cooperation Programme.

The ESPON EGTC is the Single Beneficiary of the ESPON 2020 Cooperation Programme. The Single Operation within the programme is implemented by the ESPON EGTC and co-financed by the European Regional Development Fund, the EU Member States and the Partner States, Iceland, Liechtenstein, Norway and Switzerland.

This delivery does not necessarily reflect the opinion of the members of the ESPON 2020 Monitoring Committee.

Authors
Renaud Le Goix, Université de Paris - UMS RIATE – CNRS Paris 75013 (France) – Project coordinator
Ronan Ysebaert, Université de Paris - UMS RIATE – CNRS (France) – Project manager
Timothée Giraud, CNRS, UMS RIATE – Université de Paris (France)
Marc Lieury, UMS RIATE – CNRS, University Paris 1 Panthéon-Sorbonne (France)
Guilhem Boulay, University of Avignon (France)
Thomas Louail, Géographie-cités (France)
José Ravier Ramasco, FISC-CSIC (Spain)
Mattia Mazzoli, FISC-CSIC (Spain)
Pere Colet, FISC-CSIC (Spain)
Thierry Teurillat, Haute-Ecole Arc (Switzerland)
Alain Segessemann, Haute-Ecole Arc (Switzerland)
Szymon Marcinczak, University Lodz (Poland)
Bartosz Bartosiewicz, Univeristy Lodz (Poland)
Elisabete Silva, Cambridge University (UK)
Sølve Baerug, Norwegian University of Life Sciences (Norway)
Terje Holsen, Norwegian University of Life Sciences (Norway)

Advisory Group
ESPON EGTC: Marjan Van Herwijnen (project expert), Caroline Clause (financial expert)

Information on ESPON and its projects can be found on www.espon.eu.

The web site provides the possibility to download and examine the most recent documents produced by finalised and ongoing ESPON projects.

Contact: info@espon.eu

ISBN: 978-99959-55-99-1

Final Report

ESPON Big Data for Territorial Analysis and Housing Dynamics

Wellbeing of European citizens regarding the affordability of housing.

Technical Guidance Document
12/11/2019

Table of contents

List of Maps

List of Figures

List of Tables

Abbreviations

ESPON	European Territorial Observatory Network
ESPON EGTC	ESPON European Grouping of Territorial Cooperation
EU	European Union
LAU	Local Administrative Unit
FUA	Functional Urban Area
IDS	Internet Data Sources
OECD	Organisation for Economic Co-operation and Development
TOR	Terms of reference

1 Introduction

This *technical guidance document* describes and demonstrates the methodological framework applied to produce the ESPON EGTC *Big Data for Territorial Analysis of Housing Dynamics 2018-19* study, delivered as the *Wellbeing of European citizens regarding the affordability of housing* report.

While in European larger cities, decent and affordable housing is increasingly hard to get access to, the main goals of the study were:

(1) to set up the framework towards the production of neighborhood and local spatial data;

(2) to implement the framework with harmonized indicators, to examine the unequal spatial patterns of housing affordability in Europe;

(3) to do it in a way that allows to compare between cities and within cities.

Its policy-oriented broader thinking is to analyze the spatial patterns of unequal local affordability, as framed by the *Action Plan of the Partnership on Housing of the EU Urban Agenda* that pushes for improved knowledge regarding affordability of housing.

The report addresses **the housing elements of European policies through one major issue: affordability,** a concept defined as a gap between housing prices and households' income (Friggit, 2017), and this gap has widened during the last decades. Since the 1990s, housing prices have on average increased faster than the income of residents and buyers in major post-industrial city-regions, but this is not ubiquitous. The scientific and policy goals of the study aim at informing and locally mapping the increased affordability gap, a critical issue for social cohesion and sustainability in metropolitan areas in Europe that impacts the well-being of residents in European cities. To do so, **the guidance document aims at discussing the data collection and data models used for the production of the delivered maps and data:**

- The "*Wellbeing of European citizens regarding the affordability of housing*" datasets have been produced to analyse and map affordability in a selection of European cities.
- The report combines institutional data, and data harvested on real-estate advertisement websites. The issues with collecting and harmonizing such heterogeneous data sources (conventional and unconventional data) are discussed, as well as the methodology proposed to bridge such datasets.
- The datasets delivered are structured as spatial data with harmonized indicators that allow **to compare between cities and within cities, to examine the unequal spatial patterns of housing affordability**.

- From the harmonized database, the study focuses on 9 case studies that cover a range of cities: from global and capital cities, down to medium-sized cities. Case studies offer a variegated sample, with several dynamics regarding housing market (gentrification process, tourism presence, housing crisis, etc.). Highlighting these various and complementary situations is relevant to carry out a first international and comparative study on housing dynamics in Europe based on local indicators. This guidance document will highlight **the technical choices made to compare the indicators between cities in different countries**.

The *guidance document* describes the data selection, harvesting and analysis process. It is structured as follows:

- Section 2 overviews the conceptual and theoretical model used for standardisation and delivery of aggregated datasets, as well as data sources gathered, and described in the *wellbeing report*.
- Section 3 describes the data model and provides a description of the data collection and harmonisation processes. It discusses the technical choices and procedures for data harvesting, as well as the procedures for data harmonisation. This section gives insights on how results are technically produced and delivered, by the means of a **workable example**, elaborated with the Paris case study. This section of the guidance document is structured according to the workflow of the analysis, and narratively describes methods and R code used to implement the case study, in order to provide ESPON, stakeholders and policymakers with the conditions of **reproducibility** and **transferability** of the methodology used.[1]
- Section 4 summarizes methodological and conceptual outputs and elaborate on next steps for further studies.
- Several appendices complement the *Technical Guidance Document*, and are made available with the final delivery, for the sake of reproducibility and transferability: they include (1) **an annex on harvesting data**, with Python language libraries; (2) **full code** for the preparation of case-studies (R language), (3) a workable smaller example to demonstrate the diverse procedures used (RMarkdown *html* document) developed as a **prototype demonstrator** of the libraries and packages used for the project.[2]

[1] Most programs used to prepare this report have been written in the R language, using a series of packages that are documented in the Guidance document. R language is now a widely adopted open-sourced standard programming language in spatial analysis, big data analysis, and statistics. Harvesting websites also required the use of Python language (see Annex 1)

[2] Full code in the R language is provided as used in data and spatial analysis (http://bit.ly/2XGEhiD). Readers may also refer to an .html file, a RMarkdown document consisting in an archive of preliminary stages of the project developments (D1_Draft_outline_guidance_document.html). This file was constructed as a workable prototype example of the main packages used to produce analyse the datasets and harvest data.

2 Outlines of the conceptual and theoretical model

Starting with an explicit theoretical framework of affordability, the design of a harmonised data structure is not an easy process and must follow specific methods and procedures. One issue consists in defining an adapted methodology for combining conventional and unconventional data sources. Another is to describe harmonisation procedures, which are not only technical, but also conceptual. This section summarises some conceptual and theoretical elements that are further elaborated in the *Wellbeing Report*, so as to introduce the technical choices of the final data design.

2.1 Theoretical model

The theoretical model underlying the study starts with the problem of home values and market values, and the widening gap compared to income (Aalbers, 2016; Friggit, 2017). The theoretical framework from which we elaborate on stems an overarching conceptualisation of **affordability as part of a feedback loop between residential markets, value, assets, wealth and vulnerability,** thoroughly developed in Le Goi*x et al*. (2019a); Le Goi*x et al*. (2019b). This theoretical framework is detailed in the *Wellbeing report*, section 1.2 and 1.3. The report describes to effort towards collecting, documenting (metadata) local spatial datasets to spatially analyse some critical issues:

- The increased affordability gap, a critical issue for social cohesion sustainability in metropolitan areas in Europe.
- The unequal access to housing markets.
- The increased inequalities stemming from declining affordability (i.e. higher price to income ratio)

2.2 Harmonizing conventional and unconventional data to analyze the well-being

One major issue is the lack of harmonised spatial data to map and monitor affordability in Europe. There are plenty of institutional (tax, census), private (real-estate agents and websites) and national or local data (parcels, local tax rolls). These are not harmonised and interoperable. As demonstrated in the *Wellbeing report (*section 2.1), data by OECD and Eurostat are disseminated respectively at the national and at the city levels, but the dataset are far from complete in terms of thematic and geographical objects available to accurately analyze housing dynamics. To bridge this data gap at the local level (LAU2 and FUA), we collected and combined different spatial datasets and surveys which have so far been employed separately. One issue consisted in defining an adapted methodology for **combining conventional and unconventional data sources.**

Definition of conventional and unconventional data

(quoted from the *Wellbeing report*, section 3).

Conventional data are provided by traditional statistical offices. This information, usually collected at the individual scale and disseminated at several geographical aggregates, is subject to robust processes of harmonisation and validation, by means of explicit conceptualisation of the future usage of the data collected. Conventional data are usually realised through vintages (like censuses), but rely on robust survey, samples and inferential statistics methodologies. Such data are disseminated with explicit description of the fields and variable construction, definitions, sampling procedures, and statistical robustness.

Unconventional data are extracted from various platforms and sources, and are often named "big data". Some might come from institutions, and are datasets collected for various administrative, fiscal reasons, but that were not originally designed for socio-economic or demographic research. In many ways, we incorporate them in research whereas such datasets have not been designed and/or documented to do so (lack of metadata). Although originating from institutions, their robustness, as well as inference on how such data describe the general population can often be questioned.

Many unconventional datasets are also derived from harvested data, made available by ISP (Internet Service Providers) by the means of API, or scraped. Such unconventional data are often viewed as interesting proxies to measure, and better understand spatial behaviors and territorial dynamics (Gallotti *et al*.; Kitchin, 2013), and also as a means of providing higher spatio-temporal resolution data when compared to institutional data sources (FP7 EUNOIA final report, 2015).

3 Technical section: case-study analysis

For the purpose of the *Wellbeing of European citizens regarding the affordability of housing* report we go beyond the aggregated territorial levels to understand intra-urban inequalities between the cities. In term of data creation processing, three challenges have been addressed in this research:

(1) Ensuring a data process which can be reproducible and transferable. It was mainly done through and important documentation and programming codes.

(2) Delivering comparable and harmonised indicators for the selected cities and case studies, 2 geographical levels are used to aggregate the collected data: the LAU2 level, and the 1km European reference grid.

(3) Covering the entire Functional Urban Areas, despite missing data and incomplete datasets (mainly due to the absence of real-estate transactions or offers in the periphery of FUA, which are mainly rural areas).

This section describes and demonstrates how we address each of these challenges, the technical solutions implemented, and the data analysis design. All code for the study is made available with the final delivery of the report. See “Programs” folder, containing R files, delivered as a report appendix[3]. The “Appendices” section of this document contains Python code used for scraping purposes in France.

3.1 Transferability and reproducibility of the study: a narrative of a case-study analysis (Paris)

We elaborate on the case-study of Paris as an ideal case study because of the availability of institutional data[4]. We consider a variety of datasets:

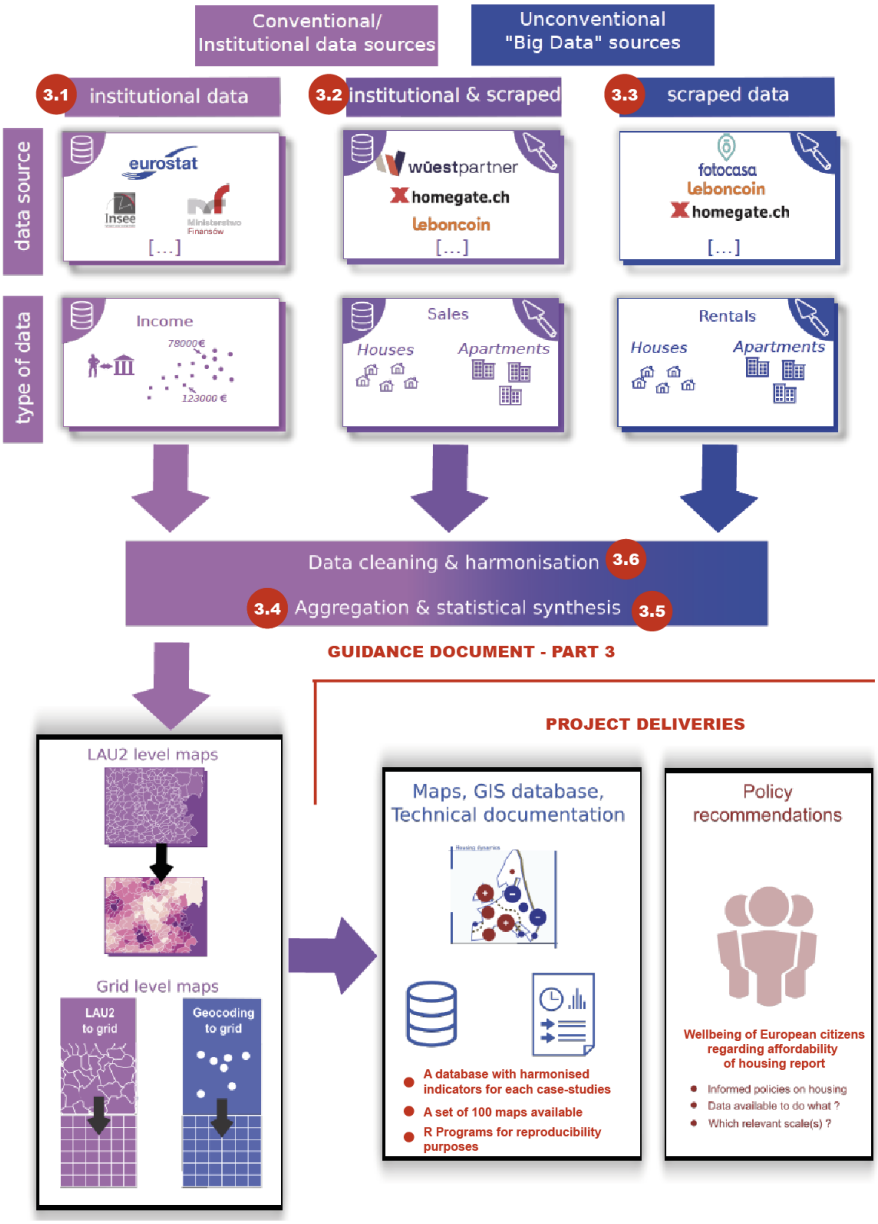
- public conventional census data,
- unconventional institutional datasets, property-level data from the Paris Chamber of Notaries (1996-2012, a sample of 1 million rows),
- unconventional harvested big data sources (real estate websites).

[3] https://sharedocs.huma-num.fr/wl/?id=LPBWZm39VsZpXWnqIi9Q9HiTu0p5j9cF

[4] The *final guidance document* describes the *full methodology* as a narrative summarizing the data process and R code used from a *complete case study*. It differs from interim deliveries, that were based on a *prototype analysis* designed as a RMarkdown document (Interim Delivery on Yvelines), provided to describe and reproduce the overall workflow of analysis targeted. This interim document is available online: https://www.espon.eu/sites/default/files/attachments/Guidance_Document_201900426.pdf

Harmonised and standardised variables are proposed, down to the local level (1 km grid). Methods are applied to a set of geospatial data available. Section 3.3 documents the R language code written for the purpose of the analysis: the goal is to **demonstrate the transferability and reproducibility of the methodology**. The open-source R statistical software uses open-source packages, that are well documented and maintained, and is considered a standard environment in massive geospatial data analysis. By documenting the R code with this narrative section, we document how the methodological framework has been made transferable. It has been implemented with data from other case-studies to prepare maps and visualisations for the *Wellbeing report*, availability of datasets permitting (transaction data namely). In other words, all the indicators produced for each case-studies stem from the same methodology, and are therefore made fully comparable within cities and across cities.

Figure 3.1 - Overview of the workflow.



In this section, we deliver a methodological narrative, in order:

- To describe the **data collection process**, both for conventional and unconventional data sources.
- To describe and document the methodology employed to harvest datasets, using APIs, and R packages *s.a.* `Cartography`, `SpatialPostion`, `rvest`, and `httr`, so as to ensure **reproducibility and transferability of the protocols.**
- To describe a set of **harmonised variables**. Harmonised variables should be made comparable between European cities, within cities and when data are available over time. Ratios and standardized indices, such as affordability ratio are considered as valuable alternatives to rough stock variables (s.a. price, surface), that are structurally contingent to each country, city and local market contexts.
- To correct spatial and temporal data gap. Spatio-temporal information is sensitive to two types of sampling issues: in space, and in time, therefore requiring the use of interpolation procedures to ensure the quality and representativeness of the spatial information produced. Spatial interpolation procedures are described, using for instance the `Spatialposition` R Package.

To illustrate the project workflow, the narrative displayed in Chapter 3 aims at covering the most important methodological aspects of harmonised data creation on real-estate market (Figure 3.1) :

1. **Data sources and data cleaning** – A first issue highlighted is data collection from the relevant information/data providers to target harmonized indicators. Conventional census data are required to extract information at EU level on socio-economic characteristics of case-studies and collect data on income, fundamental for estimating real-estate affordability (Section 3.1). Institutional unconventional data are required to describe residential property markets (Section 3.2). Data harvesting and web-scraping (Section 3.3) has been critical in overcoming the situation where transaction data simply does not exist. Where official transaction data exists, harvested data allows to document the real-estate market offer (both property and rental market) vs. actual transactions.
2. **Harmonised indicators provision** – Section 3.4 covers the design of harmonised variables in a multiscalar perspective, produced at the local fine-grain geographical city level (grid and LAU2) and the level of the entire FUA. Many are produced by bridging institutional and census data.
3. **Spatial harmonisation** – Most of the indicators have been delivered at the LAU2 level. Section 3.5 explains the methodology and the interest to go beyond this territorial level relevant for policy making by aggregating and interpolating the results

in a 1km INSPIRE grid. This methodology offers a lot of advantages: by the means of interpolation/smoothing techniques, we control for outliers and errors in the input datasets. It also allows to better control for the MAUP effects (take into account the number of real-estate offers/transactions in the calculation of price/square meters), it estimates missing values using the assumption of spatial autocorrelation of real-estate values. It finally allows to go beyond the LAU2 level which is basically too large for observing existing inequalities for some cities defined by large territorial units, such as Barcelona or Warsaw.

4. **Data sources combination**. Section 3.6 displays promising results for Paris and Barcelona where some **extra-combinations** between unconventional data sources have been tested for the sake of **data exploration**: real-estate offers (transaction data in Paris, real-estate offer in Barcelona) and other big data source, like Airbnb. Maps of statistical residuals produced clearly show the effect of Airbnb presence on real-estate market (less offers than expected, all things being equal to the density of real-estate offers or transactions) in touristic quarter (Center or Montmartre in Paris, La Rambla in Barcelona).

This guidance document is concluded with highlights on how such data, maps and interpretation realised have been aggregated together to produce harmonized indicators and analysis at the three geographical levels of interest for this study, *depending of data availability*: local grids, LAU2, central cities, FUAs.

3.2 Conventional institutional data: Eurostat and National Statistical Institutes indicators

Two main categories of institutional data providers ("official statistics") have been used: harmonised European statistics (Eurostat) and national statistics (data coming from National Statistical Institutes and Finance ministries).

EU statistics (Urban statistics, Eurostat) have been used to globally characterise selected case-studies as regards to the other cities of Europe. Taking into account the availability of data, 20 indicators at Core City and 14 at FUA level (less available information) have been identified. These datasets include demographic indicators (age-structure), main households characteristics, information related to the employment (economy tertiary oriented or not) and other relevant factors to understand *who* lives in the cities. However, few information are available on housing. It is only possible to extract one item of the EU perception survey: "is it easy to find good housing in your city?", which gives only a very rough qualitative assessment of affordability by European citizens.

Table 3-1 – Listing of Eurostat available indicators relevant for characterising the housing market

ID	Indicator Name	Reference
POP_2015	Total population	Table "urb_cpop1"
POP024_2015	Share of population aged 0-24 years (%)	Table "urb_cpop1"
POP2544_2015	Share of population aged 25-44 years (%)	Table "urb_cpop1"
POP4564_2015	Share of population aged 45-64 years (%)	Table "urb_cpop1"
POP65_2015	Share of population aged above 65 years (%),	Table "urb_cpop1"
HOUSEHOLD_AREA_	Household area (sq. meters)	Table "urb_clivcon"
HOUSEHOLD_SIZE_2	Household size (persons)	Table "urb_clivcon"
SINGLE_HOUSEHOLI	Share of single households (%)	Table "urb_clivcon"
OWNED_DWELLING	Share of owned dwellings (%)	Table "urb_clivcon"
UNEMP_2014	Unemployment rate (%)	Table "urb_clma"
EMP_INDS_2014	Share of employment in industry (%)	Table "urb_clma"
EMP_HOTELS_2014	Share of employment in restauration, hotels and transports (%)	Table "urb_clma"
EMP_RESTATE_2014	Share of employment in real estate activities (%)	Table "urb_clma"
ST_HIGH_EDU_2011	Share of students in higher education (ISCED 5-6) (per 1000 persons	Table "urb_ceduc"
WF_HIGH_EDU_2011	Proportion of popuation aged 25-64 years qualified at level 5 to 8 ISCED	Table "urb_ceduc"
NIGHTS_2011	Total nights spent in tourist accommodation establishments per resident population	Table "urb_ctour"
BEDS_2011	Number of available beds per 1000 residents	Table "urb_ctour"
HOUSING_EASY_201	Share of persons answering "Strongly agree" to the question of the Urban Audit : is it easy to find good housing in your city ?	Table "urb_leduc"
HOUSING_DIFFICULT	Share of persons answering "Strongly disagree" to the question of the Urban Audit : is it easy to find good housing in your city ?	Table "urb_percep"

For measuring further comparable affordability indexes, the results of the EU-SILC survey have also been used. This is the reference source for comparative statistics on income distribution in the European Union. For this study, the first, the fifth (median) and the ninth decile of income distribution have been gathered. This kind of information can be further (cf. Section 3.4) used to analyse a time-normalised indicator of affordability (price to income ratio), *i.e.* answering the following question: "*How long the 10% poorest/median/10% richest of the population have to work to buy/rent 1sq. kilometer in this city"?* The choice of thresholds has been made with regards to the needs for data standardisation to compare national affordability between cities of several countries, which is not the case with income statistics provided at local level by National Statistical Institutes.

In fact, LAU2 income data have also been gathered. It is especially useful for discussing on affordability in a local context. A disclaimer shall however be disclosed regarding income data. *It is not recommended to compare local situation of affordability between cities of several countries*. Indeed, the methodologies for income computation varies from one country to another. The ways income is imputed to persons or households differ between countries: per capita or per households; before/after tax; with or without social benefits, etc. The statistical parameters for aggregated income also differs in institutional data, some a median, other are average income at LAU2 level. Nevertheless, local affordability indexes are still highly relevant for comparing local affordability between cities of the same country (Avignon-Paris / Madrid, Barcelona-Palma de Mallorca / or Lodz-Warsaw-Krakow separately), the methodology of income calculation being generally harmonised at national level.

3.3 Using unconventional institutional data to analyze the dynamics on property markets: data, methods, sample results

3.3.1 Unconventional institutional data: data from Paris Chamber of the Notaries

We use property-level data from the Paris Chamber of Notaries (1996-2012), provided to the lead researcher by the Paris Notaries Services, a subsidiary of the Chamber of the Notaries, under a research agreement with the Paris Chamber of the Notaries[5]. This sample contains transactions for the region and its suburbs, within the administrative limits of Ile-de-France (roughly 1 million rows). All records contain information on the property amenities and pricing, and series of understudied interesting variable on sellers and buyers, such as age, sex, socio-economic status, national origin, place of residence, and some credit history related to the transaction (95 indicators at transaction scale, cf R CODE 1).

########################## R CODE 1 ##########################

```
#Import BIEN database (Chamber of Notaries) and analyse
dfdata <- read.csv("BIEN_LABEX_2016_consolidated_all_years.txt", stringsAsFactors=FALSE)
```

```
> str(dfdata)
'data.frame':           968695 obs. of  96 variables:
 $ X.1         : int  1 2 3 4 5 6 7 8 9 10 ...
 $ ID          : int  1 2 3 4 5 6 7 8 9 10 ...
 $ ACONST      : int  NA NA NA 1996 1996 1996 1996 1996 1996 1996 ...
 $ ANNAIS_AC   : int  1966 1967 1969 1961 1971 1945 1970 1937 1970 1973 ...
 $ ANNAIS_VE   : int  NA NA NA NA NA NA NA NA NA NA ...
 $ annee       : int  1996 1996 1996 1996 1996 1996 1996 1996 1996 1996 ...
 $ ANNEXE      : chr  "" "" "" "" ...
 $ BATEAU      : chr  "" "" "" "" ...
 $ BIARRON     : chr  "" "" "" "" ...
 $ BICOMPADR   : chr  "ET 12" "ET 12" "ET 12" "zac des 2 golfs" ...
 $ BICOMPNRVO  : chr  "" "" "" "" ...
 $ BIDEPT      : chr  "77" "77" "77" "77" ...
 $ BILIBVOIEO  : chr  "PAUL VALENTIN" "PAUL VALENTIN" "PAUL VALENTIN" "CHAMP DE LAGNY" ...
 $ BINRQUARAD  : chr  "" "" "" "" ...
 $ BINRVOIE    : chr  "10" "10" "10" NA ...
 $ BINUCOM     : chr  "5" "5" "5" "18" ...
 $ BITYPVOIE   : chr  "RUE" "RUE" "RUE" "LD" ...
 $ CAVE        : chr  "1" "0" "1" NA ...
 $ CODNAT_AC   : chr  "F" "F" "F" "F" ...
 $ CODNAT_VE   : chr  "F" "F" "F" "F" ...
 $ CSP_AC      : chr  "60" "60" "60" "51" ...
 $ CSP_VE      : chr  "" "" "" "" ...
 $ DATDEBBAIL  : chr  "" "" "" "" ...
 $ DATMUTPREC  : chr  "26/01/1995 00:00" "26/01/1995 00:00" "26/01/1995 00:00" "02/11/1995 00:00" ...
 $ DEPENDANCE  : chr  "" "" "" "" ...
 $ DURBAIL     : chr  NA NA NA NA ...
 $ ENCOMBRE    : chr  NA NA NA NA ...
 $ ETAGE       : chr  "2" "2" "0" "2" ...
 $ IMSURFTOTB  : chr  NA NA NA NA ...
 $ INDIVI_AC   : chr  "N" "I" "N" "N" ...
 $ INDIVI_VE   : chr  "N" "N" "N" "N" ...
 $ insee       : chr  "77005" "77005" "77005" "77018" ...
 $ IRIS        : chr  "770050000" "770050000" "770050000" NA ...
 $ LARGFAC     : chr  NA NA NA NA ...
 $ LOYANNU     : chr  NA NA NA NA ...
 $ mois        : int  6 5 7 3 3 3 3 3 3 3 ...
 $ MTCRED      : chr  "59455" "63022" "51070" "74395" ...
 $ NATNEGOC    : chr  "PR" "PR" "PR" "" ...
 $ NBRBAT      : chr  NA NA NA NA ...
 $ NBRCHSERV   : chr  "0" "0" "0" NA ...
 $ NBRGARAGE   : chr  "0" "0" "0" "1" ...
 $ NBRPIECE    : chr  "3" "3" "" "3" ...
 $ NBRSALDB    : int  1 NA NA 1 1 1 1 1 1 1 ...
 $ NIVEAU      : chr  NA NA NA NA ...
 $ Nom_commune: chr  "Annet-sur-Marne" "Annet-sur-Marne" "Annet-sur-Marne" "Bailly-Romainvilliers" ...
 $ NRPLAN1     : chr  "426" "426" "426" "106" ...
 $ NUMCOM_AC   : chr  "5" "294" "438" "81" ...
 $ NUMCOM_VE   : chr  "372" "372" "372" "512" ...
 $ PADEPT_AC   : chr  "77" "77" "77" "94" ...
 $ PADEPT_VE   : chr  "77" "77" "77" "59" ...
 $ PISCINE     : chr  "" "" "" "" ...
 $ PRESCREDIT  : chr  "0" "0" "0" "0" ...
 $ PXMUTPREC   : chr  "" "" "" "" ...
 $ QUALITE_AC  : chr  "" "" "" "" ...
 $ QUALITE_VE  : chr  "PR" "PR" "PR" "SC" ...
 $ REFSECTION  : chr  "B" "B" "B" "AD" ...
 $ REQ_AF_AC   : chr  "RP" "" "" "" ...
 $ REQ_AF_VE   : chr  "" "" "" "" ...
 $ REQ_ANC     : chr  "1" "1" "1" "2" ...
 $ REQ_ASCENC  : chr  "" "" "" "N" ...
 $ REQ_CHAUFC  : chr  "" "" "" "" ...
```

5 The transactions BIEN proprietary database was made available by Paris Notaire Service, on the behalf of the Chamber of the Notaries, under an agreement contracted by the LabEx DynamiTe (ANR-11-LABX-0046) and the Univ. Paris 1 Pantheon-Sorbonne.

```
 $ REQ_COS    : chr  "0" "0" "0" "0" ...
 $ REQ_DUREE  : chr  "17" "16" "18" "4" ...
 $ REQ_EPOQU  : chr  "B" "B" "B" "G" ...
 $ REQ_JARDIN : chr  "" "" "N" "" ...
 $ REQ_MUT    : chr  "1" "1" "1" "1" ...
 $ REQ_NIVGAR : chr  NA NA NA NA ...
 $ REQ_OCC    : chr  "1" "4" "3" "3" ...
 $ REQ_PM2    : num  945 NA NA 1593 1383 ...
 $ REQ_POS    : chr  "" "" "" "" ...
 $ REQ_PRIX   : chr  "47259" "50308" "51070" "89182.49" ...
 $ REQ_SURFT  : chr  "0" "0" "0" "0" ...
 $ REQ_VALUE  : chr  NA NA NA NA ...
 $ REQTYPBIEN : chr  "AP" "AP" "AP" "AP" ...
 $ SDHOP      : chr  NA NA NA NA ...
 $ SEXE_AC    : chr  "M" "M" "M" "M" ...
 $ SEXE_VE    : chr  "" "" "" "" ...
 $ SHON       : chr  NA NA NA NA ...
 $ SITMAT_AC  : chr  "M" "C" "M" "M" ...
 $ SITMAT_VE  : chr  "" "" "" "" ...
 $ SURFHABDEC : chr  "50" NA NA "56" ...
 $ TAUXTVA    : chr  "A" "A" "A" "H" ...
 $ TAXPF      : chr  "N" "N" "N" "O" ...
 $ TENNIS     : chr  "" "" "" "" ...
 $ TERRASSE   : chr  "N" "N" "N" "N" ...
 $ TXDRMUT1   : chr  "3" "3" "3" "0" ...
 $ TYPAP      : chr  "AS" "AS" "DU" "AS" ...
 $ TYPBAIL    : chr  "" "AU" "" "" ...
 $ TYPGAR     : chr  NA NA NA NA ...
 $ TYPMAI     : chr  "" "" "" "" ...
 $ TYPMUTPREC : chr  "A" "" "A" "A" ...
 $ TYPPRO     : chr  "P" "P" "P" "P" ...
 $ USAGE      : chr  "HA" "HA" "HA" "HA" ...
 $ VIABILISAT : chr  NA NA NA NA ...
 $ X          : chr  "628337" "628337" "628337" "0" ...
 $ Y          : chr  "2436285" "2436285" "2436285" "0" ...
```

##########

The dataset is then filtered and a subset is prepared (ordinary transactions, residential only, apartments only, without null geographical coordinates, for 2011 and 2012), that represents 90000 observations. Nevertheless, some outliers may appear in the sample, due to data entry mistakes, which is a manual process for notaries in France). A first data cleaning consists in excluding to the sample exceptional values (1% lower and highest price/square meter transaction values). After data cleaning and filtering, the sample is reduced down to 76000 observations (cf. R CODE 2).

########## R CODE 2 ##########

```
library(dplyr)
# Convert required fields in numeric
dfdata <- BIEN_LABEX
dfdata$annee <- as.numeric(dfdata$annee)
dfdata$REQ_PRIX <- as.numeric(dfdata$REQ_PM2)

# Convert required fields in numeric
dfdata[,c("REQ_PRIX")] <- as.numeric(dfdata[,c("REQ_PRIX")])
dfdata <- dfdata %>%
  filter(REQ_MUT==1) %>% #  ordinary transactions (OTC) btw sellers and buyers "gré à gré"
  filter(USAGE=="HA") %>% #  residential only
  filter(REQ_PRIX > 1) %>% #  price > 1 eur
  filter(REQTYPBIEN=="AP" | REQTYPBIEN=="A" ) %>% #Appartements uniquement REQTYPBIEN=="AP" |
REQTYPBIEN=="A"
  filter(X!=0)  %>% filter(Y!=0) %>% #  no 0 or 1 coordinates
  filter(X!=1)  %>% filter(Y!=1) %>% #  no 0 or 1 coordinates
  filter(annee == 2011 | annee == 2012)  %>%
  filter(!is.na(X))  %>% filter(!is.na(Y)) #  no NA coordinates

# Prepare the data
dfdata$REQ_PRIX <- as.numeric(dfdata$prix_ttc)
dfdata$REQ_SURFT <- as.numeric(dfdata$srf_hab_est)
dfdata$NBRPIECE <- as.numeric(dfdata$nbr_pieces)
dfdata$MTCRED <- as.numeric(dfdata$mnt_cred)
dfdata$X <- as.numeric(dfdata$x)
dfdata$Y <- as.numeric(dfdata$y)
dfdata$PRIX_SURF <- dfdata$REQ_PRIX / dfdata$REQ_SURFT
```

```
# Delete outliers
bornesQuantiles_prix <- quantile(dfdata$PRIX_SURF, probs = seq(0,1, 0.01), na.rm=TRUE)
> bornesQuantiles_prix
      0%       1%       2%       3%       4%       5%       6%       7%       8%       9%      10%
 200.000 1562.500 1800.000 1950.617 2062.500 2156.863 2245.895 2317.073 2385.285 2449.275 2500.000
     91%      92%      93%      94%      95%      96%      97%      98%      99%     100%
9309.791 9528.302 9782.609 10000.000 10370.296 10789.474 11315.789 12071.193 13444.368 45000.000

dfdata <- subset(dfdata, PRIX_SURF >= bornesQuantiles_prix[2] & PRIX_SURF <= bornesQuantiles_prix[100])

##################################################################################################
```

3.3.2 Data aggregation in grid and LAU2 level

The focus with *transaction data* being to analyse the geography of affordability through home-ownership inequalities with transactions, several issues have to be dealt with regarding **the spatial level of aggregation**. The information displayed below is aggregated at a very low level of spatial granularity (1km grid, LAU2). On the one hand, it provides information in an "official" territorial division, as such (LAU2). But on the other hand, these datasets are subject to important outliers affecting the quality of the results, especially for territorial units described by a small number of real estate transactions, requiring the use of a grid to perform interpolation and estimation spatial statistics (1km grid).

- For spatial analysis purpose, the 1km grid allows to integrate datasets with various spatial definitions;
- Data secrecy, privacy control and legal and/or ethic requirements regarding the confidentiality of individual transactions ;
- The MAUP (Modifiable Areal Unit Problem), related to the spatial distribution of transactions and aggregation ;
- The weakness of the sample and missing data issues.
- Grid interpolation (cf Section 3.6)

In concrete terms, this step (cf R CODE 3) consists in aggregating data at transaction level[6] to the LAU2 and EU 1km grid, for a set of **basic targeted indicators** (price, surface, rooms, debt contracted[7]). To make possible further analysis, these indicators are aggregated using several statistical parameters: first quartile (Q25), median (Q50), third quartile (Q75), interquartile range (IQR), sum). Sum is calculated to make possible the computation of weighted ratios from the transaction level to aggregated LAU2 and 1km grid level (price per square meters for instance). For confidentiality issues and contract restrictions imposed by the data provider (Paris Notaries Service, Chamber of the Notaries), grid/LAU2 cells including less than 5 records are discarded and set to "NA" (Not Available). Finally, resulting

[6] As agreed in the study contract with ESPON, it is not possible to disseminate this proprietary disaggregated database.

[7] This selection of indicators corresponds to the most basic information it is possible to harmonize at European level. Even if the Chamber of Notaries database delivers data on socio-economic categories of sellers/buyers, few data providers deliver this kind of data in Europe.

aggregated data are merged to EU reference layers thanks to their respective ID and exported to an Excel file. Transaction data are at this stage ready to be used for further analysis.

```
########################          R CODE 3          ########################
Library(xlsx)
# Reference import (LAU2 and grids)
FUA <- st_read(dsn = paste0("Mapkits/",CS,"/CommunesCS.shp"), stringsAsFactors = F)
Grid <- st_read(dsn = paste0("Mapkits/",CS,"/GridCS.shp"), stringsAsFactors = F)


################## Aggregation in Grid layer
# Points location of transactions in grid cells
# X-Y BIEN DB (CRS = Lambert II étendu)
dfgeom <- st_as_sf (dfdata, coords = c ("X","Y"), crs = 27572)
dfgeom <- st_transform(dfgeom,3035)
dfgeom <- st_join(dfgeom, Grid)

# Aggregate targeted indicators by point grid
temp <- as.data.frame(dfgeom %>% count(GRD_ID))
colnames(temp) <- c("GRD_ID","TRANS_NUMBER")

temp1 <- as.data.frame(group_by(dfgeom, GRD_ID) %>%  summarise(PRICE_PAID_SUM = sum(REQ_PRIX, na.rm
= TRUE)))
temp2 <- as.data.frame(group_by(dfgeom, GRD_ID) %>%
                summarise(PRICE_PAID_Q25 = quantile(REQ_PRIX, probs = 0.25, na.rm  = TRUE),
                        PRICE_PAID_Q50 = quantile(REQ_PRIX, probs = 0.5, na.rm  = TRUE),
                        PRICE_PAID_Q75 = quantile(REQ_PRIX, probs = 0.75, na.rm  = TRUE)))
temp2$PRICE_PAID_IQR <- temp2$PRICE_PAID_Q75 - temp2$PRICE_PAID_Q25
temp3 <- as.data.frame(group_by(dfgeom, GRD_ID) %>% summarise(SURFACE_SUM = sum(REQ_SURFT, na.rm
= TRUE)))
temp4 <- as.data.frame(group_by(dfgeom, GRD_ID) %>%
                summarise(SURFACE_Q25 = quantile(REQ_SURFT, probs = 0.25, na.rm  = TRUE),
                        SURFACE_Q50 = quantile(REQ_SURFT, probs = 0.5, na.rm  = TRUE),
                        SURFACE_Q75 = quantile(REQ_SURFT, probs = 0.75, na.rm  = TRUE)))
temp4$SURFACE_IQR <- temp2$PRICE_PAID_Q75 - temp2$PRICE_PAID_Q25
temp5 <- as.data.frame(group_by(dfgeom, GRD_ID) %>% summarise(ROOMS_SUM = sum(NBRPIECE, na.rm  =
TRUE)))
temp6 <- as.data.frame(group_by(dfgeom, GRD_ID) %>%
                summarise(ROOMS_Q25 = quantile(NBRPIECE, probs = 0.25, na.rm  = TRUE),
                        ROOMS_Q50 = quantile(NBRPIECE, probs = 0.5, na.rm  = TRUE),
                        ROOMS_Q75 = quantile(NBRPIECE, probs = 0.75, na.rm  = TRUE)))
temp6$ROOMS_IQR <- temp6$ROOMS_Q75 - temp6$ROOMS_Q25
temp7 <- as.data.frame(group_by(dfgeom, GRD_ID) %>% summarise(DEPT_SUM = sum(MTCRED, na.rm  =
TRUE)))
temp8 <- as.data.frame(group_by(dfgeom, GRD_ID) %>% summarise(DEPT_Q50 = quantile(MTCRED, probs = 0.5,
na.rm  = TRUE)))
temp9 <- as.data.frame(group_by(dfgeom, GRD_ID) %>%
                summarise(PRICESURF_Q25 = quantile(PRIX_SURF, probs = 0.25, na.rm  = TRUE),
                        PRICESURF_Q50 = quantile(PRIX_SURF, probs = 0.5, na.rm  = TRUE),
                        PRICESURF_Q75 = quantile(PRIX_SURF, probs = 0.75, na.rm  = TRUE)))
temp9$PRICESURF_IQR <- temp9$PRICESURF_Q75 - temp9$PRICESURF_Q25
transactions <- cbind(temp,temp1, temp2,temp3,temp4,temp5, temp6, temp7, temp8, temp9)

GRIDdata <- Grid[,c("GRD_ID")]

# Merge with EU reference layer
GRIDdata <- merge(GRIDdata,
            transactions[,c("GRD_ID", "TRANS_NUMBER", "PRICE_PAID_SUM",
                        "PRICE_PAID_Q25","PRICE_PAID_Q50","PRICE_PAID_Q75",
                        "PRICE_PAID_IQR","SURFACE_SUM","SURFACE_Q25","SURFACE_Q50",
                        "SURFACE_Q75","SURFACE_IQR","PRICESURF_Q25","PRICESURF_Q50",
                        "PRICESURF_Q75","ROOMS_SUM","ROOMS_Q25","ROOMS_Q50","ROOMS_Q75",
                        "ROOMS_IQR","DEPT_SUM","DEPT_Q50")], by.x = "GRD_ID", by.y = "GRD_ID", all.x =
TRUE)

st_geometry(GRIDdata) <- NULL

# Export in XLS
write.xlsx(as.data.frame(GRIDdata), file = "TRANSACTIONS_PARIS.xls", sheetName = "grid",
```

```
        col.names = TRUE, append = TRUE, row.names = FALSE, showNA = FALSE)


################## Aggregation LAU2 level
temp <- as.data.frame(dfdata %>% count(insee))
colnames(temp) <- c("insee","TRANS_NUMBER")

temp1 <- as.data.frame(group_by(dfdata, insee) %>% summarise(PRICE_PAID_SUM = sum(REQ_PRIX, na.rm  =
TRUE)))
temp2 <- as.data.frame(group_by(dfdata, insee) %>%
                       summarise(PRICE_PAID_Q25 = quantile(REQ_PRIX, probs = 0.25, na.rm  = TRUE),
                                 PRICE_PAID_Q50 = quantile(REQ_PRIX, probs = 0.5, na.rm  = TRUE),
                                 PRICE_PAID_Q75 = quantile(REQ_PRIX, probs = 0.75, na.rm  = TRUE)))
temp2$PRICE_PAID_IQR <- temp2$PRICE_PAID_Q75 - temp2$PRICE_PAID_Q25
temp3 <- as.data.frame(group_by(dfdata, insee) %>% summarise(SURFACE_SUM = sum(REQ_SURFT, na.rm  =
TRUE)))
temp4 <- as.data.frame(group_by(dfdata, insee) %>%
                       summarise(SURFACE_Q25 = quantile(REQ_SURFT, probs = 0.25, na.rm  = TRUE),
                                 SURFACE_Q50 = quantile(REQ_SURFT, probs = 0.5, na.rm  = TRUE),
                                 SURFACE_Q75 = quantile(REQ_SURFT, probs = 0.75, na.rm  = TRUE)))
temp4$SURFACE_IQR <- temp2$PRICE_PAID_Q75 - temp2$PRICE_PAID_Q25
temp5 <- as.data.frame(group_by(dfdata, insee) %>% summarise(ROOMS_SUM = sum(NBRPIECE, na.rm  =
TRUE)))
temp6 <- as.data.frame(group_by(dfdata, insee) %>%
                       summarise(ROOMS_Q25 = quantile(NBRPIECE, probs = 0.25, na.rm  = TRUE),
                                 ROOMS_Q50 = quantile(NBRPIECE, probs = 0.5, na.rm  = TRUE),
                                 ROOMS_Q75 = quantile(NBRPIECE, probs = 0.75, na.rm  = TRUE)))
temp6$ROOMS_IQR <- temp6$ROOMS_Q75 - temp6$ROOMS_Q25
temp7 <- as.data.frame(group_by(dfdata, insee) %>% summarise(DEPT_SUM = sum(MTCRED, na.rm  = TRUE)))
temp8 <- as.data.frame(group_by(dfdata, insee) %>% summarise(DEPT_Q50 = quantile(MTCRED, probs = 0.5,
na.rm  = TRUE)))
temp9 <- as.data.frame(group_by(dfdata, insee) %>%
                       summarise(PRICESURF_Q25 = quantile(PRIX_SURF, probs = 0.25, na.rm  = TRUE),
                                 PRICESURF_Q50 = quantile(PRIX_SURF, probs = 0.5, na.rm  = TRUE),
                                 PRICESURF_Q75 = quantile(PRIX_SURF, probs = 0.75, na.rm  = TRUE)))
temp9$PRICESURF_IQR <- temp9$PRICESURF_Q75 - temp9$PRICESURF_Q25

transactions <- cbind(temp,temp1, temp2,temp3,temp4,temp5, temp6, temp7, temp8, temp9)

# Delete transactions number below 5 on the given period
transactions <- transactions[ which(transactions$TRANS_NUMBER > 5), ]

# Merge income and transactions with LAU2 reference
LAU2data <- FUA[,c("CENSU","LAU2_","NAME_")]

LAU2data <- merge(LAU2data,
                  transactions[,c("insee", "TRANS_NUMBER",
                                  "PRICE_PAID_SUM","PRICE_PAID_Q25","PRICE_PAID_Q50",
                                  "PRICE_PAID_Q75","PRICE_PAID_IQR","SURFACE_SUM",
                                  "SURFACE_Q25","SURFACE_Q50","SURFACE_Q75",
                                  "SURFACE_IQR","PRICESURF_Q25","PRICESURF_Q50",
                                  "PRICESURF_Q75", "PRICESURF_IQR", "ROOMS_SUM",
                                  "ROOMS_Q25","ROOMS_Q50", "ROOMS_Q75","ROOMS_IQR",
                                  "DEPT_SUM", "DEPT_Q50")],
                  by.y = "insee", by.x = "LAU2_", all.x = TRUE)


st_geometry(LAU2data) <- NULL

# Export in XLS
write.xlsx(as.data.frame(LAU2data), file = "TRANSACTIONS_PARIS.xls", sheetName = "LAU2",
        col.names = TRUE, append = TRUE, row.names = FALSE, showNA = FALSE)

##########################################################################################################
```

Figure 3.2 - Data output : aggregation of relevant indicators at LAU2 and grid level for 10 case-studies (extract).

GRD_ID	TRANS_NI	PRICE_PA	PRICE_PA	PRICE_PA	PRICE_PA	PRICE_PA	SURFACE	SURFACE	SURFACE	SURFACE	SURFACE	PRICESUF	PRICESUF	PRICESUF	ROOMS_S	ROOMS_C	ROOMS_C	ROOMS_C	ROOMS_I	DEPT_SUI	DEPT_Q50
1kmN2893	772	2,35E+08	180607,5	255000	377250	196642,5	31405	27	36	50	196642,5	6566,803	7403,846	8187,279	1605	1	2	3	2	22949439	0
1kmN2892	650	2,5E+08	201775	295450	487000	285225	30234	25,25	37	57	285225	7152,623	8235,294	9325,701	1528	1	2	3	2	30842850	0
1kmN2893	587	1,91E+08	189300	275000	418000	228700	25034	27	37	54	228700	6779,221	7592,593	8327,851	1258	1	2	3	2	16729275	0
1kmN2890	545	1,72E+08	174500	252000	396000	221500	21866	23	33	50	221500	7030,075	7866,429	8714,286	1095	1	2	3	2	22371170	0
1kmN2889	498	1,75E+08	200000	280000	451125	251125	21291	25	35	54,75	251125	7448,898	8242,954	8929,928	1065	1	2	3	2	18594976	0
1kmN2892	491	2,13E+08	185000	330000	575280	390280	24949	24	40	66	390280	7480,432	8488,372	9495,283	1217	1	2	3	2	22460907	0
1kmN2893	483	1,66E+08	179500	279000	448500	269000	20265	25,5	36	52	269000	6845,299	7927,083	8967,262	1095	2	2	3	1	25791335	0
1kmN2890	469	2,04E+08	215000	350000	579808	364808	23050	25	41	65	364808	7727,273	8823,529	10108,33	1093	1	2	3	2	23951327	0
1kmN2888	442	1,89E+08	237562,5	345900	551112,5	313550	20821	28	41	59,75	313550	8144,18	8857,143	9746,651	1013	1,75	2	3	1,25	19114472	0
1kmN2888	413	1,51E+08	200220	300000	475000	274780	18563	26	40	56	274780	7250	8008,8	8830,196	909	1	2	3	2	22536043	0
1kmN2887	410	1,58E+08	230000	329500	470000	240000	18986	29	41	58,75	240000	7590,973	8373,547	9136,905	904	1	2	3	2	10908679	0
1kmN2891	402	1,42E+08	185000	282500	481000	296000	18912	25	38	62,75	296000	6894,726	7627,083	8347,076	932	1	2	3	2	16612160	0
1kmN2891	399	1,88E+08	219000	365000	650000	431000	22546	27	44	76,5	431000	7532,252	8414,286	9379,419	1012	1	2	3	2	23039506	0
1kmN2886	391	1,62E+08	233500	346000	532500	299000	18710	28,5	41	62	299000	7674,797	8573,77	9395,122	910	1	2	3	2	14363640	0
1kmN2892	380	2,09E+08	264750	419850	672722,3	407972,3	23039	31	47	73	407972,3	7995,098	8973,143	9820,742	1024	2	2	3	1	18094475	0
1kmN2887	379	1,51E+08	210040	344925	510357,3	300317,3	20831	32	47	72	300317,3	6269,136	7026,316	8024,555	970	2	2	3	1	27292030	0
1kmN2891	374	1,14E+08	165915	255000	398000	232085	16575	24,25	36,5	56	232085	6147,841	6998,931	7726,705	803	1	2	3	2	16393035	0
1kmN2890	373	1,24E+08	205410	300000	413700	208290	17708	29	43	61	208290	6318,182	6956,522	7642,857	843	1	2	3	2	11055522	0
1kmN2892	372	2,19E+08	280000	437500	810453,2	530453,2	23920	30	50	85,25	530453,2	8322,818	9398,413	10389,48	1007	1	2	4	3	21574739	0
1kmN2893	359	78525009	142000	192830	258000	116000	12983	25	32	44	116000	5419,201	6064,516	6835,72	695	1	2	2	1	11308206	0
1kmN2888	358	1,46E+08	200000	305537,5	546300	346300	17104	25	38	60,75	346300	7614,268	8524,948	9252,155	802	1	2	3	2	16295297	0
1kmN2890	357	1,5E+08	200250	330000	511231	310981	16200	23	36	57	310981	8153,846	9455,139	10564,1	747	1	2	3	2	18296142	0
1kmN2888	356	1,75E+08	250000	355775	599637,5	349637,5	18572	29,75	40	66,25	349637,5	8252,795	9214,841	10210,84	849	2	2	3	1	11108089	0
1kmN2890	354	3,12E+08	338176,8	674500	1209713	871535,8	30884	36,25	68	115	871535,8	8942,071	10000	11185,88	1159	2	3	5	3	38737902	0
1kmN2892	354	99008662	158250	230000	350000	191750	15155	26	38	54	191750	5669,118	6422,52	7219,293	771	1	2	3	2	12531987	0
1kmN2893	350	1,59E+08	240000	354775	579500	339500	20020	33,25	47	73	339500	7041,976	7748,077	8628,472	949	2	2	3	1	19726403	0
1kmN2894	348	1,57E+08	241750	390220	587100	345350	20470	36	53	74	345350	6616,806	7465,35	8247,429	952	2	3	4	2	18212186	0
1kmN2891	345	1,03E+08	177000	250000	370000	193000	14621	26	37	54	193000	6333,333	7066,667	7666,667	726	1	2	3	2	11667916	0
1kmN2888	343	1,45E+08	210000	355000	540250	330250	18754	31	47	70	330250	6404,981	7396,552	8334,621	854	2	2	3	1	18123589	0
1kmN2889	340	1,14E+08	192750	285000	433914	241164	14112	25	35	53	241164	7139,194	7951,632	8750	709	1	2	3	2	18670032	0
1kmN2896	325	80161202	148000	225500	320000	172000	15821	32	46	63	172000	4422,899	5027,083	5528,587	785	2	2	3	1	10361146	0
1kmN2888	322	1,35E+08	249250	360000	529088,5	279838,5	16152	30	46	64	279838,5	7696,342	8406,3	9174,287	766	1	2	3	2	12750987	0
1kmN2895	320	1,02E+08	214170	304165	407750	193580	18363	40	56	72,25	193580	5057,636	5554,921	6099,128	859	2	3	3	1	11001885	0
1kmN2887	313	1,12E+08	225000	307000	430000	205000	14510	30	41	57	205000	7030,303	7666,667	8395,062	725	2	2	3	1	10204118	0
1kmN2892	313	99741133	185000	285000	390000	205000	15107	29	45	63	205000	5843,75	6616,333	7325,581	716	1	2	3	2	14507993	0
1kmN2895	307	91813695	180273	280000	365500	185227	16708	35,5	52	69,5	185227	4942,912	5357,143	5839,554	799	2	3	3	1	9883244	0
1kmN2889	306	1,54E+08	264250	420000	660000	395750	14876	26	39	63	395750	9400	10751,34	11770,61	681	1	2	3	2	19692374	0
1kmN2888	303	91237272	175350	250000	382602,5	207252,5	13091	26	36	54	207252,5	6302,174	6956,522	7919,297	626	1	2	3	2	11864602	0
1kmN2890	303	82119173	165000	242000	340125	175125	11865	24,5	35	48	175125	6195,238	7083,333	7837,509	616	1	2	2	1	10291813	0
1kmN2887	295	1,25E+08	225250	330000	530000	304750	13307	24,5	37	59	304750	8461,174	9500	10550,05	676	1	2	3	2	10188751	0
1kmN2889	281	1,92E+08	337250	631000	928000	590750	20790	41	69	102	590750	8309,859	9230,769	10210,53	850	2	3	4	2	15240069	0
1kmN2887	277	1E+08	215500	324000	460000	244500	12269	27	42	56	244500	7462,963	8214,286	8870,968	611	1	2	3	2	7355552	0
1kmN2885	272	1,06E+08	249499,9	349999,8	466124,9	216624,9	15644	38	53	71,25	216624,9	6186,523	6875,731	7478,194	708	2	3	3	1	26787462	206500
1kmN2888	267	1,37E+08	280000	405000	743569,5	463569,5	17173	37	56	90	463569,5	7055,075	8013,107	8832,168	685	2	2	4	2	17595162	0
1kmN2894	261	77797863	195000	265050	379650	184650	14188	37	50	70	184650	4712,644	5538,462	6291,667	632	2	2	3	1	9458524	0

3.4 Using unconventional data sources: web-scraping of real-estate online listings

Unconventional data are often viewed as interesting proxies to measure, and better understand spatial behaviours and territorial dynamics, and also as a means of providing higher spatiotemporal resolution data when compared to institutional data sources. Prior to relying upon the unconventional data sources, it is important to assess their reliability, and to assess how accurate the information provided is when compared to the long established conventional data provision, an institutional statistically robust information collection data.

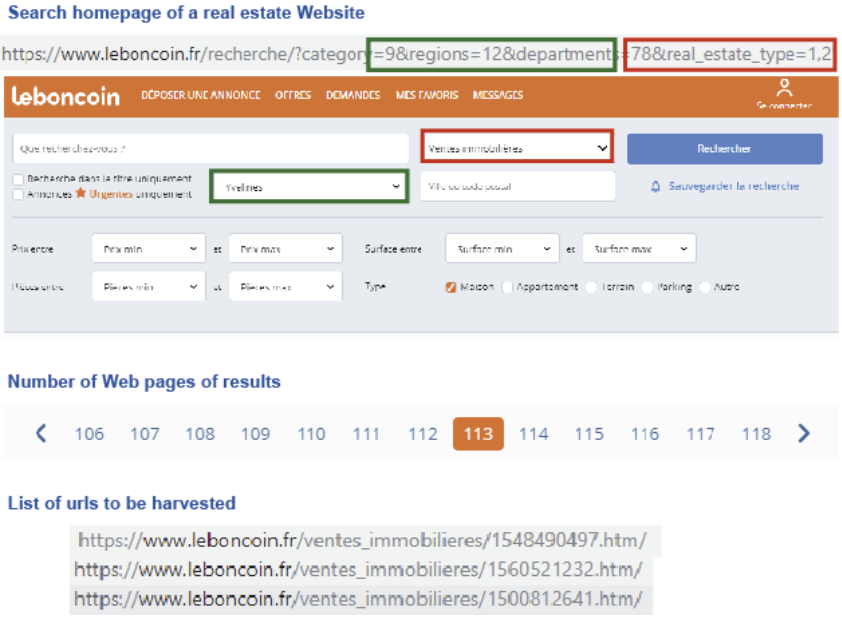
Gathering real-estate data on the internet (real-estate advertisement websites) requires to follow a general procedure, which can be summarized as below:

A. Real-estate website identification – Real-estate websites are aggregators of advertisement originating from real-estate agents, but also individuals, that generally have a national coverage (the ads covers generally a single country). In other terms, it requires for each country to define a listing of real-estate agencies in leadership situation (to obtain a maximum of ads, well-structured and referenced). For France (Avignon, Geneva French part and Paris), *leboncoin.fr* has been deemed of interest for this study, for its coverage and for its data structure that allows a straightforward data harvesting effort, given the timeframe of the study : in May 2019, it included 44886 real-estate offers (apartments only) and 20484 rental offers in Paris.

B. Harvesting real estate online listings (ads) – Step 2 consists in getting the total number of ads and determine the number of pages to scrape after having identified the relevant *tags*

syntax in the URL query: as displayed in Figure 3.3 it corresponds to a geographical *tag* (name of the NUTS3) and the type of offer *tag* (real estate offer / rental ; apartments and-or houses, etc). Then, the method consists in exploring automatically all the adds included in all pages of the query result. The output is a list of URLs to be harvested (one URL by offer).

Figure 3.3 - Get links to all ads of a given real estate Website



C. Identifying all the relevant information for listed properties. The next step consists in preparing the script for each website to automatically fetch the data. It requires to harvest the html webpage, and identify all the interesting attributes/tags to get (price, number of rooms, surface, geographical location[8]…), as presented on Figure 3.4.

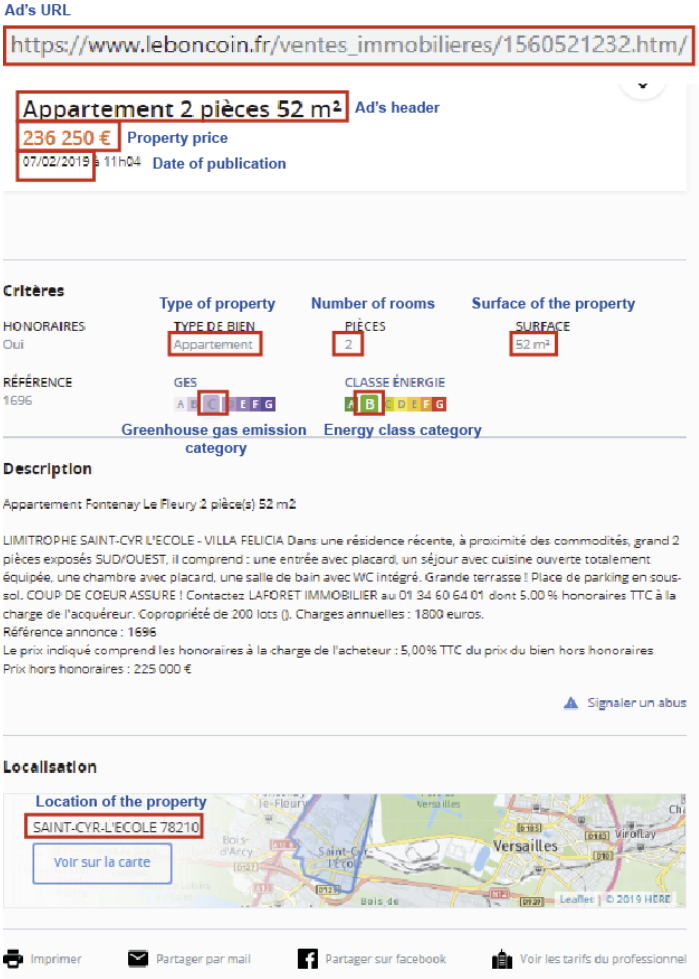
This is **a tedious, very costly and time-consuming process** that requires a lot of retro-engineering. The cost and duration of the project allowed only for test drives and a few months of collection, and some test platforms. We deliver a general methodology: it is obvious that a script is valid for one real estate Website, considering the fact that they are coded differently. Moreover, if the real estate Website change the organisation of the Web page, the tags used in the script must be re-written. **Such an iterative procedure is hand-made, highly artisanal, and highly consuming in qualified worked-force, therefore costly.**

D. Data cleaning. The most common mistakes errors are duplicated ad's (sometimes a real estate ad can be published several times), absence of location coordinates or mistakes when entering the real estate ad (area, price, etc.). Consequently, results obtained through the Web scraping effort must be filtered. As an example for a case-study located in Yvelines in France,

[8] Ideally, X/Y coordinates must be scrapped. For some real-estate website, like leboncoin.fr, it is quite difficult to obtain this information. The LAU2 and the zip code are considered in this case.

the 9934 observations collected resulted in 7460 unique and accurate records, with correct location down to the municipality. The geocoding resulting from this procedure is in many regards of poor quality compared to the locations provided by institutional data (transactions). In the preliminary study, location data is provided by the website either as the city or municipality (78%), and only a few ads are geocoded down to the address. In some other cases location appear to be the location of the agency.

Figure 3.4 - Collect and sparse geospatial information



E. Data aggregation. Cleaned web-scraped data are then aggregated in targeted geographical delineation (LAU2, grid), following the same methodology than the one used for transaction data (Section 3.3. For some countries (Poland, Spain), it is possible to have accurate X/Y location and it is consequently possible to aggregate real-estate offers at grid level; for other countries, like in France with leboncoin.fr, X/Y location of real-estate offers is not directly available, due to the design of the webpage. Harvested data can only be aggregated at LAU2 scale for French real-estate offers.

3.5 Harmonised indicators (FUA and LAU2 scale)

Following the data collection/cleaning procedures (steps 3.3 and 3.4), a dataset is created (as a '**sf'** object in R), which includes all the information required to launch the analysis, respectively:

- Official geometries (LAU2 and grid)
- Income data (municipal and national income data)
- Aggregated transaction data, if available (rooms, surface, price, debt contracted)
- Aggregated web-scraped data for property sales (rooms, surface, price)
- Aggregated web-scraped data for property rental (rooms, surface, price)

With this structured information, it is possible to launch the analysis on the housing market and produce harmonised indicators combining conventional, institutional and unconventional data sources as exemplified in R CODE 4. This section provides examples on how to match data contained in transactions and other web-scraped data sources, and how to combine them to external sources, like municipal income (coming from National censuses) or national income (EU surveys)[9].

The first series of harmonised indicators to be built informs on the main characteristics of the housing market: size (surface, number of rooms) of the properties (transactions, offer, rental), number of offers, price per sq. meters of local markets. This information gives important insights to document **the heterogeneity of real-estate market** segments in case-studies.

The second series of indicators **combine prices**, to compare housing prices between and within case studies. These are variables of interest to get an overarching understanding unequal access to housing: advertised price, income, debt, for instance. To better understand inequalities on housing markets, we start with nominal price, and then produce harmonized variables, based on ratios, s.a. price-to-income, to analyze affordability; and debt-to-value, a proxy for inequalities stemming from equity capital availability of households (data available for French case-studies only). Using **municipal income** allows to highlight the financial **effort that local households have done** to get another property or property rental on **local real-estate market**. On the other hand, the use of **national income** questions on the effort that a standard household **should do** to access the local real-estate market studied in the case study, highlighting how inaccessible a metropolitan market can be.

The section describes and documents the harmonized indicators produced at the FUA level, than at the LAU2 levels, and finally at the 1k Grid level.

[9] The use of national income provided by EU surveys allows namely to overcome the heterogeneity of national income definitions.

3.5.1 FUA indicators

One of the aims of the study consists also in producing statistical synthesis of transaction data at FUA levels (all Paris FUA unit, with available data). This is done with the *HousingStat* function, created for the project and described below (cf R CODE 4), which computes 29 indicators at FUA scale (using the LAU2 raw dataset, and aggregating these indicators using weighted quantiles) relevant for discussing on housing characteristics and affordability:

- Transaction number (number)
- Surface of property (Q25, Q50, Q75, average)
- Number of rooms (Q25, Q50, Q75, average)
- Price paid (Q25, Q50, 75, average)
- Price paid per sq. meter (Q25, Q50, Q75, average)
- Time required to buy one square meter with municipal average income[10] (10% of the poorest-richest population + median income)
- Time required to rent one square meter according to municipal or national income (10% of the poorest-richest population + median income)
- Debt contracted / transaction value.

The interest of this function is also to use it for web-scraped datasets and other case-studies, as displayed below in the synthetic table of the well-being report. It allows consequently to produce/update quite easily, data being available, statistical synthesis at FUA scale.

Table 3-2 – Harmonised indicators created at FUA scale (real estate offer) displaying synthetic indicators at FUA scale

	STATISTICS	Geneva (CH)	Geneva (FR) / Annecy (FR)	Warsaw (PL)	Lodz (PL)	Krakow (PL)	Madrid (ES)	Barcelona (ES)	Palma de Mallorca (ES)	Paris (FR)	Avignon (FR)
Year of reference		2019	2019	2019	2019	2019	2019	2019	2019	2019	2019
Number of offers		1096	10801	39293	1595	9382	79227	147094	22040	44886	5397
Surface	**Q25**	132.1	71	73.4	73.6	66.3	94.55	93.57	103.4	47.4	76.5
	Q50	186	82	104.2	103.2	87.7	143.33	132.61	145.9	62	98.5
	Q75	276.1	123.5	165.7	160.0	130.5	226.25	199.13	230.4	77.5	141.1
	AV.	374.3	105.1	140.9	137.5	116.6	234.57	190.13	401.7	74.8	119.9
Rooms	**Q25**	NA	3.2	2.5	2.55	2.5	NA	2.74	NA	2.3	3.57
	Q50	NA	3.9	3.5	3.53	3.4	NA	3.33	NA	3	4.44
	Q75	NA	4.8	4.5	4.54	4.4	NA	4.16	NA	3.8	5.61
	AV.	7.6	4.1	NA	NA	NA	3.37	3.52	3.38	3	4.65
Price	**Q25**	1 184	282.9	116.1	62.3	89.7	209.3	233.7	242.1	209.1	167.8

[10] Using statistics provided by National Statistical Institute.

(thousands euros)	**Q50**	1 863	373.6	168.5	872.7	117.5	334.4	334.4	360.5	274.9	233.0
	Q75	3 076	500.4	280.8	143.4	178.0	590.7	507.8	587.3	371.3	344.8
	AV	4 460	400.2	234.6	118.65	158.4	516.7	441.8	554.8	307.9	283.8
Price per sq. meters	**Q25**	NA	3584.5	NA	NA	NA	1897.6	2193.6	1940.5	4138	1925
	Q50	NA	4133.6	NA	NA	NA	2550.5	2722.8	2552.3	4764	2404
	Q75	NA	4682.9	NA	NA	NA	3299.7	3388.9	3364.3	5474	2820
	AV	11915	4002.8	1665.3	863.2	1357.8	2202.9	2324	1381.1	4118	2366
Price to income	**LOC**[11]	67.8	15.7	22.0	15.69	19.1	15.05	14.01	19.6	12.8	14.7
	Q10[12]	217.1	35.7	77.1	39.0	52.1	93.17	79.67	100.04	25.2	24.1
	Q50	112.7	19.6	39.5	19.96	26.6	36.38	31.11	39.1	13.9	13.3
	Q90	61.4	10.6	20.7	10.48	14	17.67	15.11	19.0	7.6	7.2
Time required to buy 1sq. meter (month)	**LOC**	2.2	1.8	1.9	1.27	2.0	0.77	0.88	0.59	2.1	1.48
	Q10	7.0	4.1	6.6	3.41	5.4	4.77	5.03	2.99	4	2.41
	Q50	3.6	2.2	3.4	1.74	2.7	1.86	1.96	1.17	2.2	1.33
	Q90	2.0	1.2	1.8	0.92	1.4	0.90	0.95	0.57	1.2	0.72

########## R CODE 4 ##########

```
Library(survey)
Library(sf)
Library (housing) (This package created within the project imports reference layers for all the case studies of the ESPON Housing dynamics project (Paris, Avignon, Barcelona, Madrid, Palma de Majorque, Warsaw, Lodz, Krakow and Geneva). The mapping functions implemented allow to create an ESPON map with all the required styles (colors, labels, etc.)

# This function is used for analysing case-study data at FUA level

# HousingStat Function
HousingStat <- function(x, id, tsum, sq25, sq50, sq75, ssum,
                        rq25, rq50, rq75, rsum, pq25, pq50, pq75, psum,
                        psq25, psq50, psq75, incloc, incnatd1, incnatd5,
                        incnatd9, pop, deb, name){

  # Delete NA and geometries
  try(if(missing("tsum") | missing("x") | missing("id")) stop("Transactions/offers number (tsum), data with geometries (x) is at least required for the analysis", call. = TRUE))
  STAT <- st_set_geometry(x, NULL)
  STAT <- STAT[!(is.na(STAT[,tsum])),]

  # Number of transactions registered
  number <- sum(STAT[,tsum], na.rm = TRUE)

  # Apartment surface
  surface_q25 <- ifelse ((missing("sq25") | missing("tsum") | missing("id")), NA,
                         svymean(STAT[,sq25], svydesign(ids = STAT[,id] ,data = STAT[,sq25], weights = STAT[,tsum]))[1])
  surface_q50 <- ifelse ((missing("sq50") | missing("tsum") | missing("id")), NA,
                         svymean(STAT[,sq50], svydesign(ids = STAT[,id] ,data = STAT[,sq50], weights = STAT[,tsum]))[1])
  surface_q75 <- ifelse ((missing("sq75") | missing("tsum") | missing("id")), NA,
                         svymean(STAT[,sq75], svydesign(ids = STAT[,id] ,data = STAT[,sq75], weights = STAT[,tsum]))[1])
  surface_av <- ifelse ((missing("ssum") | missing("tsum")), NA,
                        sum(STAT[,ssum], na.rm = TRUE) / number)

  # Rooms
  rooms_q25 <- ifelse ((missing("rq25") | missing("tsum") | missing("id")), NA,
```

[11] Price to income at LAU2 level, municipalities of the FUA area (be careful to heterogeneity of income definitions among EU countries).

[12] Price to income at national level, first decile (EU-SILC Survey).

```
                svymean(STAT[,rq25], svydesign(ids = STAT[,id] ,data = STAT[,rq25], weights = STAT[,tsum]))[1])
 rooms_q50 <- ifelse ((missing("rq50") | missing("tsum") | missing("id")), NA,
                svymean(STAT[,rq50], svydesign(ids = STAT[,id] ,data = STAT[,rq50], weights = STAT[,tsum]))[1])
 rooms_q75 <- ifelse ((missing("rq75") | missing("tsum") | missing("id")), NA,
                svymean(STAT[,rq75], svydesign(ids = STAT[,id] ,data = STAT[,rq75], weights = STAT[,tsum]))[1])
 rooms_av <- ifelse ((missing("rsum") | missing("tsum")), NA,
                sum(STAT[,rsum], na.rm = TRUE) / number)

 # Prices
 price_q25 <- ifelse ((missing("pq25") | missing("tsum") | missing("id")), NA,
                svymean(STAT[,pq25], svydesign(ids = STAT[,id] ,data = STAT[,pq25], weights = STAT[,tsum]))[1])
 price_q50 <- ifelse ((missing("pq50") | missing("tsum") | missing("id")), NA,
                svymean(STAT[,pq50], svydesign(ids = STAT[,id] ,data = STAT[,pq50], weights = STAT[,tsum]))[1])
 price_q75 <- ifelse ((missing("pq75") | missing("tsum") | missing("id")), NA,
                svymean(STAT[,pq75], svydesign(ids = STAT[,id] ,data = STAT[,pq75], weights = STAT[,tsum]))[1])
 price_av <- ifelse ((missing("psum") | missing("tsum")), NA,
                sum(STAT[,psum], na.rm = TRUE) / number)

 # Prices / sq. meters
 price_met_q25 <- ifelse ((missing("psq25") | missing("tsum") | missing("id")), NA,
                  svymean(STAT[,psq25], svydesign(ids = STAT[,id] ,data = STAT[,psq25], weights =
STAT[,tsum]))[1])
 price_met_q50 <- ifelse ((missing("psq50") | missing("tsum") | missing("id")), NA,
                  svymean(STAT[,psq50], svydesign(ids = STAT[,id] ,data = STAT[,psq50], weights =
STAT[,tsum]))[1])
 price_met_q75 <- ifelse ((missing("psq75") | missing("tsum") | missing("id")), NA,
                  svymean(STAT[,psq75], svydesign(ids = STAT[,id] ,data = STAT[,psq75], weights =
STAT[,tsum]))[1])
 price_met_av <- ifelse ((missing("ssum") | missing("psum")), NA,
                  sum(STAT[,psum], na.rm = TRUE) / sum(STAT[ssum], na.rm = TRUE))

 # Price / income
 incomeloc_av <- ifelse ((missing("incloc") | missing("pop")), NA,
                  sum((STAT[,incloc] * STAT[,pop]), na.rm = TRUE) / sum(STAT[,pop], na.rm = TRUE))
 price_incomeloc <- ifelse ((missing("price_av") | missing("incomeloc_av")), NA,
                    price_av / incomeloc_av)
 price_incomenatD10 <- ifelse ((missing("price_av") | missing("incnatd1")), NA,
                       price_av / STAT[,incnatd1][1])
 price_incomenatD50 <- ifelse ((missing("price_av") | missing("incnatd5")), NA,
                       price_av / STAT[,incnatd5][1])
 price_incomenatD90 <- ifelse ((missing("price_av") | missing("incnatd9")), NA,
                       price_av / STAT[,incnatd9][1])

 # Time required to buy 1sq meter
 time_loc <- ifelse ((missing("price_met_av") | missing("incomeloc_av")), NA,
                price_met_av / (incomeloc_av/12))
 time_natd10 <- ifelse ((missing("price_met_av") | missing("incnatd1")), NA,
                  price_met_av / (STAT[,incnatd1][1]/12))
 time_natd50 <- ifelse ((missing("price_met_av") | missing("incnatd5")), NA,
                  price_met_av / (STAT[,incnatd5][1]/12))
 time_natd90 <- ifelse ((missing("price_met_av") | missing("incnatd9")), NA,
                  price_met_av / (STAT[,incnatd9][1]/12))

 # Time required to rent 1sq meter
 rent_loc <- ifelse ((missing("price_met_av") | missing("incomeloc_av")), NA,
                price_met_av / (incomeloc_av/365))
 rent_natd10 <- ifelse ((missing("price_met_av") | missing("incnatd1")), NA,
                  price_met_av / (STAT[,incnatd1][1]/365))
 rent_natd50 <- ifelse ((missing("price_met_av") | missing("incnatd5")), NA,
                  price_met_av / (STAT[,incnatd5][1]/365))
 rent_natd90 <- ifelse ((missing("price_met_av") | missing("incnatd9")), NA,
                  price_met_av / (STAT[,incnatd9][1]/365))

 # Debt to value
 debt <- ifelse ((missing("deb") | missing("psum")), NA,
             sum(STAT[,deb]) / sum(STAT[,psum]))

 # Synthesis
 CS <- c(number, surface_q25, surface_q50, surface_q75, surface_av,
        rooms_q25, rooms_q50, rooms_q75, rooms_av, price_q25, price_q50,
        price_q75, price_av, price_met_q25, price_met_q50, price_met_q75,
        price_met_av, price_incomeloc, price_incomenatD10, price_incomenatD50,
        price_incomenatD90, time_loc, time_natd10, time_natd50, time_natd90, debt,
        rent_loc, rent_natd10, rent_natd50, rent_natd90)
```

```
  Indicator <- c("number","surface_q25","surface_q50", "surface_q75", "surface_av",
                 "rooms_q25", "rooms_q50", "rooms_q75", "rooms_av", "price_q25",
                 "price_q50", "price_q75", "price_av", "price_met_q25", "price_met_q50",
                 "price_met_q75", "price_met_av", "price_incomeloc", "price_incomenatD10",
                 "price_incomenatD50", "price_incomenatD90", "time_loc", "time_natd10",
                 "time_natd50", "time_natd90", "debt", "rent_loc", "rent_natd10",
                 "rent_natd50", "rent_natd90")
  transactions <- data.frame(Indicator, CS)
  colnames(transactions)[2] <- name
  transactions <- format(transactions, scientific = FALSE, digits = 2)
  return(transactions)
}


# Import LAU2 layer containing income and transaction data
city <- hm_import(folder = "Paris", filepath = "../mapkits")
LAU2 <- city$CommunesCS

# In the parameters of the function, it is required to put in entry the LAU2 layer including all the reference
indicators (generated before) and setting all the relevant indicator labels.
transactions <- HousingStat(x = LAU2, id = "LAU2_", tsum = "TRANS_NUMBER",
                            sq25 = "SURFACE_Q25", sq50 = "SURFACE_Q50",
                            sq75 = "SURFACE_Q75", ssum = "SURFACE_SUM",
                            rq25 = "ROOMS_Q25", rq50 = "ROOMS_Q50",
                            rq75 = "ROOMS_Q75", rsum = "ROOMS_SUM",
                            pq25 = "PRICE_PAID_Q25", pq50 = "PRICE_PAID_Q50",
                            pq75 = "PRICE_PAID_Q75", psum = "PRICE_PAID_SUM",
                            psq25 = "PRICESURF_Q25", psq50 = "PRICESURF_Q50",
                            psq75 = "PRICESURF_Q75", incloc = "INCOME_Q50_1112",
                            incnatd1 = "INCOME_D1_2012", incnatd5 = "INCOME_D5_2012",
                            incnatd9 = "INCOME_D9_2012", deb = "DEPT_SUM",
                            pop = "NBMENFISC_1112", name = "Paris")

##########################################################################################################
```

3.5.2 LAU2 harmonized indicators

Producing harmonized indicators at LAU2 scale is the next step. The most relevant ratios are combined starting from basic indicators. It is important to notice that it could be possible to go far beyond the scope of the study, considering all the amount of data gathered and aggregated. For instance, the data delivered for Paris includes more than 60 basic indicators. It could be consequently possible to create much more indicators for analysing the real-estate market.

Basically, the LAU2 targeted indicators (calculated for each case-study) are calculated as follows (R CODE 5 and Figure 3.5):

- price/sq. meter (price paid, advertised price, rental), a standard indicator, but highly contingent to the local structure of housing ;
- Three affordability indexes are calculated. *Local affordability* and *national affordability* (cf Section 3.3.1), and also the *difference between local and national affordability*. This last indicator provides an understanding weather it is easier for a local household, as compared to a household coming from the rest of the country, to access to property in the designated city (positive values) or not (negative values).
- Another index is the *profitability index*, calculated as a ratio between advertised price for property and advertised price for property rental. A high index means two things: Advertised price are high, as regards to rental offer; or rental offer are low, as regards

to real estate offer. In other terms, a high index can be interpreted as locations where **the development of rental offers may be specifically interesting for real-estate owners**.

Figure 3.5 – From basic indicators (transaction, web-scraped and institutional data) to harmonised indicators at LAU2 scale

Indicator id	CODE	Indicator name	Calculation method	Comment
1	PRICE_TRANS_SQ	Price paid for properties (Euros per square meters), average	PRICE_PAID_SUM / SURFACE_SUM	Sum of price paid / sum of property surface
2	PRICE_ASKED_SQ	Advertized price for properties (Euros per square meters), average	OFFERS_PRICE_SUM / OFFERS_SURFACE_SUM	Sum of advertized price / sum of property surface
3	RENT_ASKED_SQ	Advertized price for rental offers (Euros per square meters), average	RENT_PRICE_SUM / RENT_SURFACE_SUM	Sum of advertized price for property rental / syr of property rental surface
4	PROFIT	Profitability index	PRICE_ASKED_SQ / RENT_ASKED_SQ	Indicator 2 / Indicator 3
5	TRANS_SQ_METERS_LOC	Local affordability (transactions)	PRICE_TRANS_SQ / (INCOME_LOC_2011-12/12)	Price paid / m² per local income (average of median depending of the case-studies)
6	TRANS_SQ_METERS_NATD	National affordability (transactions)	PRICE_TRANS_SQ / (INCOME_NAT_D5_2011/12)	Price paid / m² per median national income
7	TRANS_SQ_METERS_DIFF	Difference between local and national affordability (transactions)	TRANS_SQ_METERS_LOC - TRANS_SQ_METERS_NATD5	Indicator 5 - Indicator 6
8	BUY_SQ_METERS_LOC	Local affordability	PRICE_ASKED_SQ / (INCOME_LOC_2015/12)	Advertized price / m² per local income (average of median depending of the case-studies)
9	BUY_SQ_METERS_NATD5	National affordability	PRICE_ASKED_SQ / (INCOME_NAT_D5_2015/12)	Advertized price / m² per median national income
10	BUY_SQ_METERS_DIFF	Difference between local and national affordability	BUY_SQ_METERS_LOC - BUY_SQ_METERS_NATD5	Indicator 8 - Indicator 9
11	RENT_SQ_METERS_LOC	Local affordability (rental)	RENT_ASKED_SQ / (INCOME_LOC_2015/365)	Advertized price (rental) / m² per local income (average of median depending of the case-studies)
12	RENT_SQ_METERS_NATD5	National affordability (rental)	RENT_ASKED_SQ / (INCOME_NAT_D5_2015/365)	Advertized price (rental) / m² per median national income
13	RENT_SQ_METERS_DIFF	Difference between local and national affordability (rental)	RENT_SQ_METERS_LOC - RENT_SQ_METERS_NATD5	Indicator 11 - Indicator 12

```
########################          R CODE 5          ##########################
# Ratios to be combined
LAU2$PRICE_ASKED_SQ <- LAU2$OFFERS_SQRMETER_MEAN
LAU2$RENT_ASKED_SQ <- LAU2$RENT_SQRMETER_MEAN
LAU2$PRICE_TRANS_SQ <- LAU2$PRICE_PAID_SUM/LAU2$SURFACE_SUM
LAU2$PROFIT <- LAU2$PRICE_ASKED_SQ / LAU2$RENT_ASKED_SQ

LAU2$TRANS_SQ_METERS_LOC <- LAU2$PRICE_TRANS_SQ  / (LAU2$INCOME_Q50_1112/12)
LAU2$TRANS_SQ_METERS_NATD5 <- LAU2$PRICE_TRANS_SQ  / (LAU2$INCOME_D5_2011/12)
LAU2$TRANS_SQ_METERS_DIFF <- LAU2$TRANS_SQ_METERS_LOC - LAU2$TRANS_SQ_METERS_NATD5

LAU2$BUY_SQ_METERS_LOC <- LAU2$PRICE_ASKED_SQ  / (LAU2$MED15/12)
LAU2$BUY_SQ_METERS_NATD5 <- LAU2$PRICE_ASKED_SQ  / (LAU2$INCOME_D5_2015/12)
LAU2$BUY_SQ_METERS_DIFF <- LAU2$BUY_SQ_METERS_LOC - LAU2$BUY_SQ_METERS_NATD5

LAU2$RENT_SQ_METERS_LOC <- LAU2$RENT_ASKED_SQ  / (LAU2$MED15/365)
LAU2$RENT_SQ_METERS_NATD5 <- LAU2$RENT_ASKED_SQ  / (LAU2$INCOME_D5_2015/365)
LAU2$RENT_SQ_METERS_DIFF <- LAU2$RENT_SQ_METERS_LOC - LAU2$RENT_SQ_METERS_NATD5

#############################################################################################################
```

3.5.3 Mapping LAU2 harmonised indicators

Harmonized indicators being available at the LAU2 level, the last section of the process can be launched to create the final maps. This is done using the '*housing'* and *'cartography' R packages.* The *housing* package has been especially developed for the project to import EU reference layers for all the case studies of the ESPON Housing dynamics project (Paris, Avignon, Barcelona, Madrid, Palma de Majorque, Warsaw, Lodz, Krakow and Geneva). The mapping functions implemented allow to create an ESPON map with all the required styles (colors, labels, logos etc.). The *cartography* package offers a series of tools to design thematic cartography such as proportional symbols, choropleth, typology, flows or

discontinuities maps. It also offers several features that improve the graphic presentation of maps, for instance, map palettes, layout elements (scale, north arrow, title...), labels or legends. These packages allow for a certain level of automation of iterative tacks. See R CODE 6.

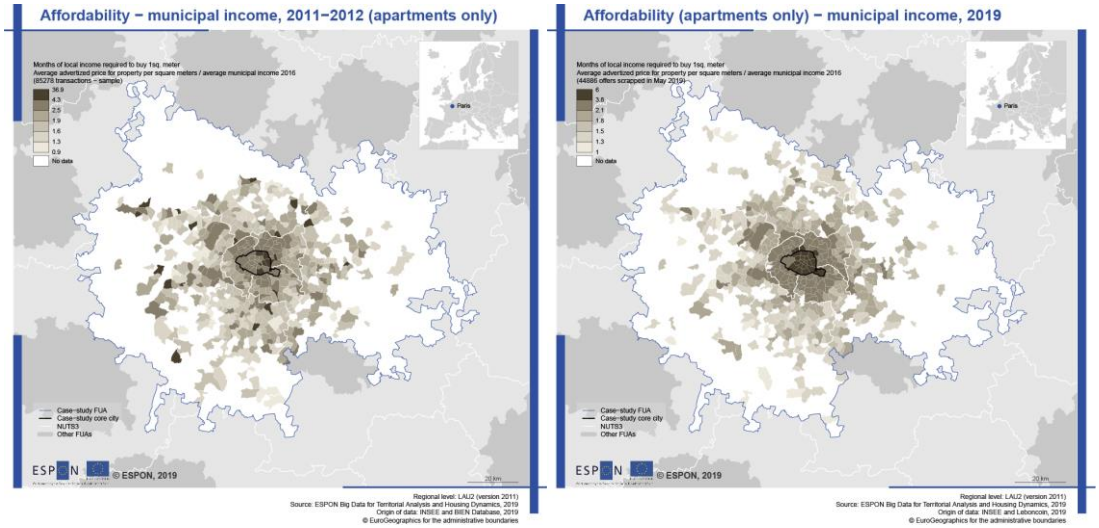
All the maps produced for the *Wellbeing* report have been created using the same methodology discretization methodology: "q6", which is a method using quantile probabilities (0, 0.05, 0.275, 0.5, 0.725, 0.95, 1). From a cartographic perspective, the interest of the thresholds it introduced for cutting the statistical series is twice: first, it allows to introduce a double colour palette (warm colours below the case-study median and cold colours above the median); second the maps created for all the case-studies are comparable: for each case-study it is possible to observe on the map the 5 % of the units with higher/lower values. This choice of thresholds is another way to make the results comparable within and across case-studies.

R CODE 6 is designed to produce 2 types of maps: price to income ratio (local income, price paid) and advertised price to income ratio (with web-scraped data). All the maps produced have been realized using this methodology. As a consequence, it is possible to produce quickly a large set of maps for each case-study for analysing housing market characteristics in 10 case-study cities.

```
#########################          R CODE 6          #########################
Library(cartography)
Library(sf)
Library (housing)
# Import reference geometries used for the map (with indicators)
city <- hm_import(folder = "Paris", filepath = "../mapkits")

# Maps parameters
sizes <- getFigDim(city$stripe, width = 800, mar=c(0,0,0,0))

#  Time required to buy 1 sq. meters locally (indicator to be mapped)
#  Extract the map in vector format, correctly sized (pdf)
pdf(file = "../fig/03_LAU2_PRICE_SQ_METERS_LOC.pdf",width = sizes[1]/72, height = sizes[2]/72,
useDingbats=FALSE, pointsize=15.3568)

#  Plot background layers (template)
hm_bg(city)

#  Map the indicator
choroLayer(LAU2, var = "TRANS_SQ_METERS_LOC",  method = "q6",
         col = carto.pal(pal1 = "taupe.pal", n1 = 6),
         colNA = "white", border = "white", lwd = 0.1, legend.pos = "n",
         add = T)

#  Plot top layers, logos and texts (data sources, title, etc.)
hm_top(x = city,  title = "Affordability - municipal income, 2011-2012 (apartments only)",
      source = "INSEE and BIEN Database, 2019", object = "LAU2")

#  Plot the legend
legendChoro(pos = c(st_bbox(city$stripes[3,])[3] + (st_bbox(city$mainframe)[3] - st_bbox(city$zoomBox)[3]),
                 (st_bbox(city$mainframe)[2] + st_bbox(city$mainframe)[4]) / 1.975),
          title.txt = "Months of local income required to buy 1sq. meter\nAverage advertized price for property per
square meters / average municipal income 2016\n(85278 transactions - sample)",
          title.cex = 0.6, values.cex = 0.5, cex = 0.8,
          breaks = getBreaks(LAU2$TRANS_SQ_METERS_LOC, method = "q6"),
          col = carto.pal(pal1 = "taupe.pal", n1 = 6),
          nodata.col = "white", values.rnd =1)
#  Export the map
dev.off()
```

```
# And the same with web-scraped data - Time required to buy 1 sq. meters locally
pdf(file = "../fig/10_LAU2_BUY_SQ_METERS_LOC.pdf",width = sizes[1]/72, height = sizes[2]/72,
useDingbats=FALSE, pointsize=15.3568)
hm_bg(city)

choroLayer(LAU2, var = "BUY_SQ_METERS_LOC",  method = "q6",
         col = carto.pal(pal1 = "taupe.pal", n1 = 6),
         colNA = "white", border = "white", lwd = 0.1, legend.pos = "n",
         add = T)

hm_top(x = city,  title = "Affordability (apartments only) - municipal income, 2019",
     source = "INSEE and Leboncoin, 2019", object = "LAU2")

legendChoro(pos = c(st_bbox(city$stripes[3,])[3] + (st_bbox(city$mainframe)[3] - st_bbox(city$zoomBox)[3]),
               (st_bbox(city$mainframe)[2] + st_bbox(city$mainframe)[4]) / 1.975),
         title.txt = "Months of local income required to buy 1sq. meter\nAverage advertized price for property per
square meters / average municipal income 2016\n(44886 offers scrapped in May 2019)",
         title.cex = 0.6, values.cex = 0.5, cex = 0.8,
         breaks = getBreaks(LAU2$BUY_SQ_METERS_LOC, method = "q6"),
         col = carto.pal(pal1 = "taupe.pal", n1 = 6),
         nodata.col = "white", values.rnd =1)
dev.off()

##########################################################################################################
```

Map 3-1 Two resulting maps created with the R CODE 6



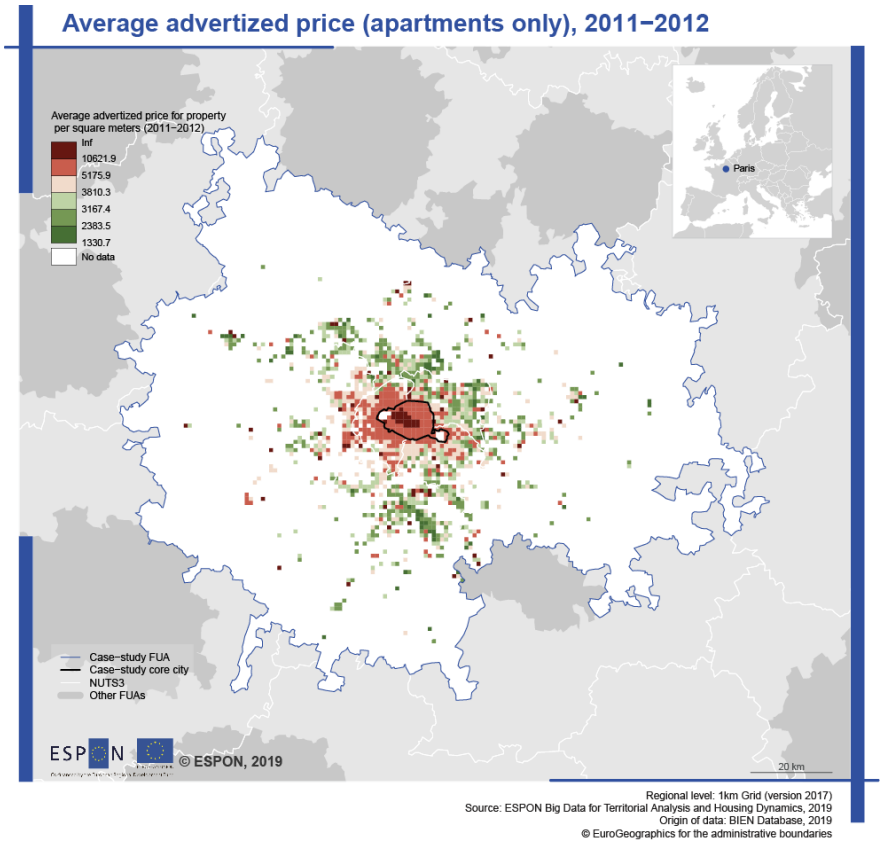
3.6 Grid data and interpolation – spatial harmonisation issue to obtain a global and accurate picture of the real-estate market locally

For each case-study, at least one indicator has been aggregated at 1 km grid level. The resulting raw map (Map 3-2) for Paris reveals three phenomena which may affect the interpretation and the dissemination of the map:

- **High spatial heterogeneity**: despite data cleaning (exceptional values), spatial structures are not clear: in the suburbs, high values are closed to lower ones. In suburban areas especially, because of the fragmented structure of the built environment and lower densities, classical econometric hypothesis regarding spatial autocorrelation of property prices are often unverifiable (Le Goi*x et al*., 2019b).

- **Missing values**. It can be due to a lack of transaction in some grid cells or the impossibility to display the data considering the fact that the number of observed transaction is below the confidentiality threshold allowed by the Chamber of Notaries database: it is not possible to disseminate data (datasets, or data displayed on maps) below a given number of 5 transactions by territorial units (at LAU2 or grid level) coming from the Chamber of Notaries database.
- Impossibility to disseminate the data as such, also due to confidentiality threshold.

Map 3-2 Average advertised price at 1 km grid level – raw map



Grid interpolation allows us to estimate a potential price in adjacent cells, with assumptions regarding the spatial interactions between transactions. To offset these limitations, we use a combination of a 1km grid and techniques of interpolation, following the assumptions of Stewart's potential, using the `SpatialPosition R package (Commenges *et al.*, 2015). For examples and detailed discussion of methodology regarding data processing, gridding, interpolation, and mapping, see (Le Goix *et al.*, 2019b).

The use of **interpolation and estimation procedures** allows to better control the quality and representativeness of the spatial information produced, which is an estimation of the price, i.e. a potential price. To do so, we used 'SpatialPosition', a R package allowing to compute Stewart potential.

The Stewart potentials of population is a spatial interaction modeling approach which aims to compute indicators based on stock values weighted by distance. These indicators have two

main interests: first, they produce understandable maps by smoothing complex spatial patterns; second they enrich the stock variables with contextual spatial information (Giraud, Commenges, 2019). At the European scale, this functional semantic simplification may help to show a smoothed context-aware picture of the localized socio-economic activities. It is also a convenient methodological solution to offset the risk of Modifiable Area Unit Problem (MAUP).

To interpolate and create Stewart potential, several steps are iterated, as displayed in the R CODE 7:

- We create a distance matrix between the grid cells. Several methods for measuring the distance can be considered: functional distances (time-road distance for instance) or mathematical distance (Euclidian distance, Manhattan distance). Here the Manhattan distance is considered, also called “taxi-distance”, especially useful for city networks.
- We compute the potential according to a specific spatial interaction function. The function inputs the matrix distance calculated above, known observations to computes the estimates **(sum of price paid and sum of property surface for Paris),** spatial interaction function (Pareto or Power law), span (distance where the density of probability of the spatial interaction function equals 0.5 – 2000 m) and a beta parameter (impedance factor for the spatial interaction function).

The parameters used for the analysis (**2000m** for the **span**, **Pareto** for the **spatial interaction function**, **2** for the beta) are justified by the resolution of the grid (1km) and a review of the literature on spatial characteristics of real-estate information, and semi-variance tests performed on the datasets.

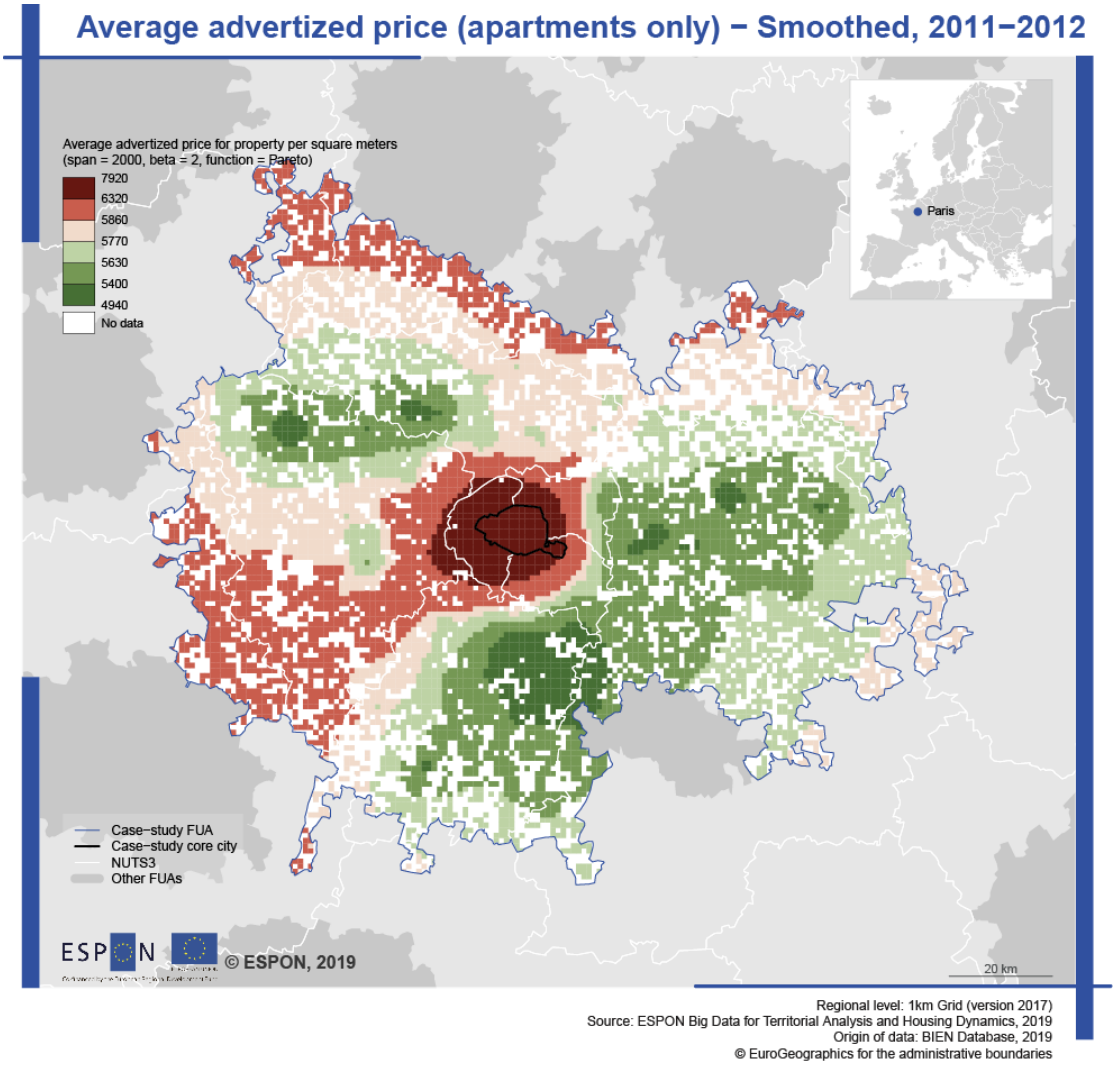
The Map 3-3 is the result of this harmonisation. It allows to go beyond the LAU2 delineation, overcome the MAUP effect, and interpolate values *ceteris paribus* the number of observed transactions and yields a global and accurate picture of the real-estate market on Paris FUA.

########################## R CODE 7 ##########################

```
library(SpatialPosition)
library(sf)

## Create dist matrix (manhattan dist)
cGRID <- st_coordinates(st_centroid(st_geometry(GRID)))
row.names(cGRID) <- GRID$GRD_ID
mat <- as.matrix(dist(cGRID, method = "manhattan"))
row.names(GRID) <- GRID$GRD_ID

# Delete 1% higest and lowest values & observations below 5
Q <- quantile(GRID$PRICE_ASKED_SQ, probs = c(.01,.99), na.rm = T)
sel_o <- GRID$TRANS_NUMBER > 5 & !is.na(GRID$TRANS_NUMBER) &
  GRID$PRICE_ASKED_SQ >= Q[1] & GRID$PRICE_ASKED_SQ <= Q[2]
x <- GRID[sel_o,]

# Span & potential calculation
span <- 2000
beta <- 2
p_prix_o <- stewart(knownpts = GRID[sel_o,], unknownpts = GRID[, "GRD_ID"],
                    varname = "PRICE_PAID_SUM", matdist = mat,
                    typefct = "pareto", beta = beta, span = span,
```

```
                returnclass = "sf")
p_surf_o <- stewart(knownpts = GRID[sel_o,], unknownpts = GRID[, "GRD_ID"],
                varname = "SURFACE_SUM", matdist = mat,
                typefct = "pareto", beta = beta, span = span,
                returnclass = "sf")

# Merge outputs with grid layer
GRID$OFFERS_PRICE_SUM_SM <- p_prix_o$OUTPUT
GRID$OFFERS_SURFACE_SUM_SM <- p_surf_o$OUTPUT
GRID$PRICE_ASKED_SQ_SM <- GRID$OFFERS_PRICE_SUM_SM/GRID$OFFERS_SURFACE_SUM_SM
##################################################################################################################
```

Map 3-3 Average advertised price at 1 km grid level – smoothed map (span = 2000m, Pareto function, beta = 2)



3.7 Comparing real-estate values to other big data sources (Airbnb)

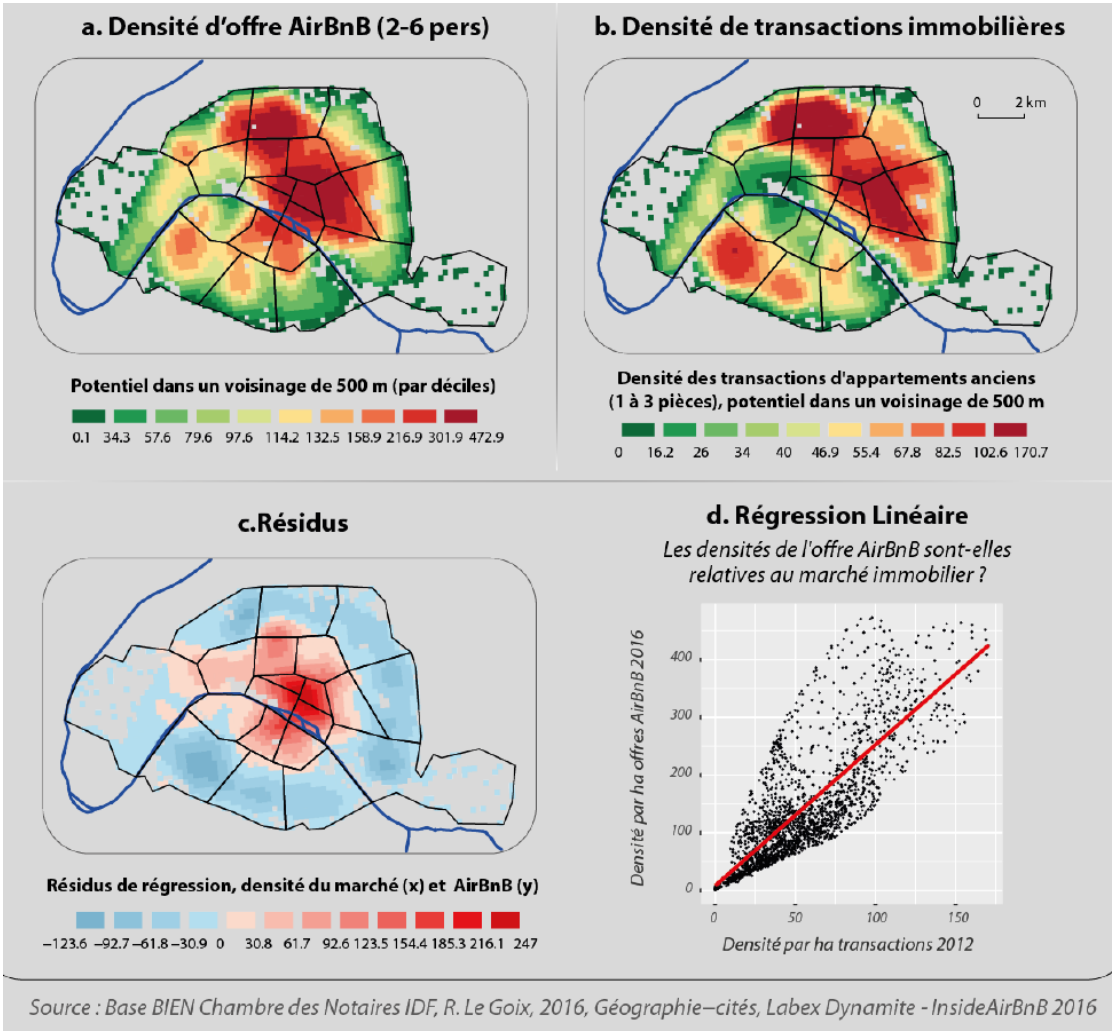
To go beyond the analysis of main data provided with the Wellbeing being report, we also tested the interest to combine these indicators with other big data sources, like Airbnb.

As a first analysis, the aim consists in answering a striking and often fuzzy policy issue: **to what extent Airbnb affects real-estate residential market?** That is to say, to what extent

the withdrawal of rental offers from residential markets puts a pressure on real estate markets that conditions the availability of affordable permanent housing for prospective buyers and renters?

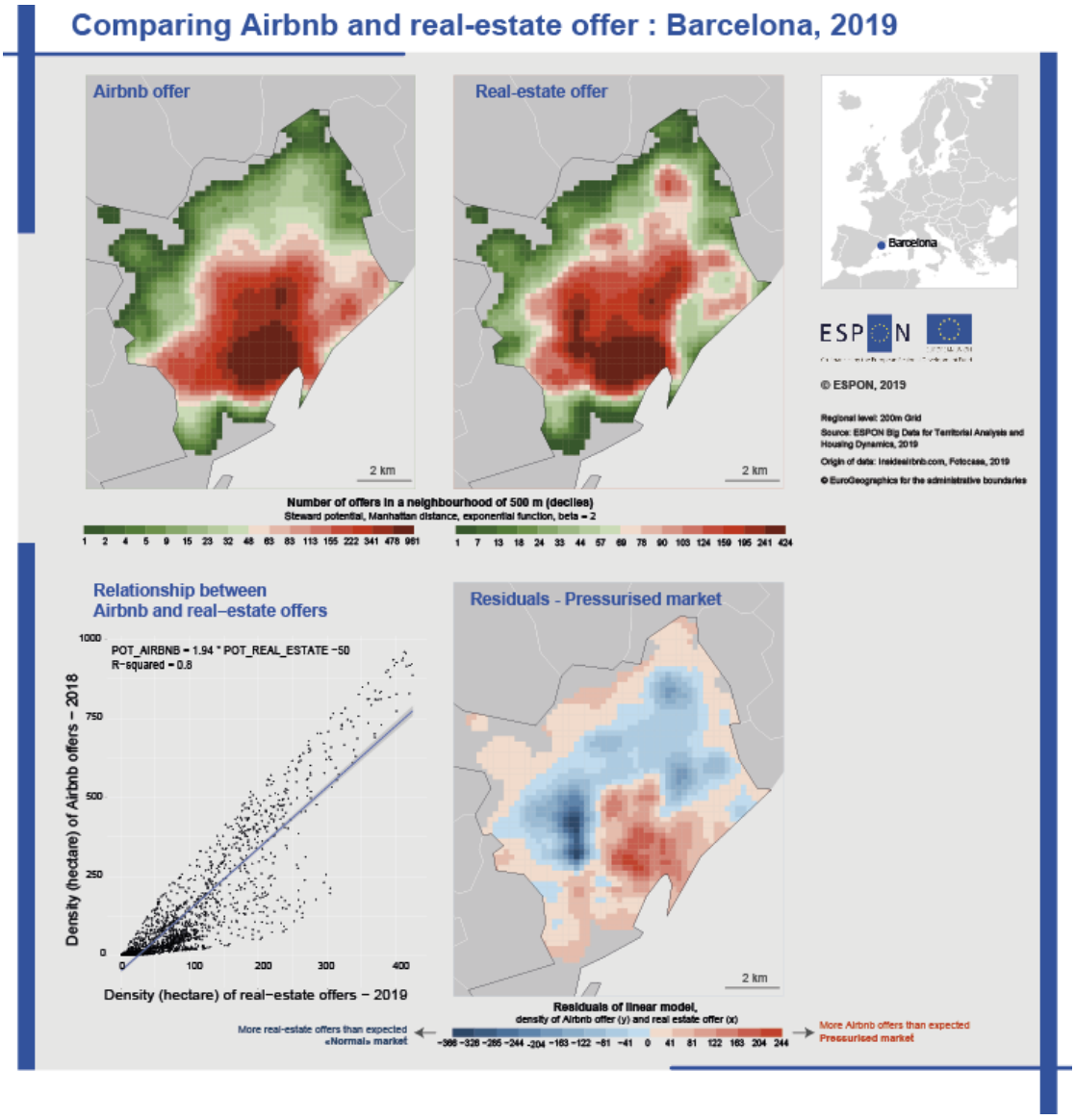
This analysis, initially done for Paris with transactions data, has revealed significant relationship between real-estate transaction and Airbnb offer (density in 200m grid cells, preliminary smoothed) (Pecou*t et al*., 2016). We do not compare price, but only the number of transactions and the number of Airbnb offers (in a regular grid, i.e. *densities*): positive outliers of a linear relationship between the number of transactions and the number of Airbnb rentals show that an *abnormal* number of properties are converted and used for short term rental, and subtracted (extracted for short term profit) from the residential markets. But more interesting is the analysis of the statistical regression and its residuals. When mapping the fine grain geography of these residuals, it appears clearly that for the center part of Paris and touristic neighborhoods (Montmartre), more Airbnb offer appear, all things being equal to the density of real-estate transactions, less real-estate transaction happened. It reveals consequently spatial concentration of real-market pressurisation.

Figure 3.6 – Comparing Airbnb and real-estate transactions (Paris).
Density of Airbnb offer (a.) compared to the density of transactions on apartment property markets (b.).
Residuals (c) of the linear regression (d).



One of the challenges of this project was to **test** the same methodology, developed in a R program (cf R CODE 8) to other case-studies. It has been done for Barcelona in the same spatial resolution (200 m grid preliminary smoothed), comparing real-estate offers (apartments) and Airbnb density offers. The statistical relationship is still significant ($r^2 > 0.8$). Residual maps show high residuals (more Airbnb offer expected, all things being equal to the density of real-estate offers) in the Core city of Barcelona (La Rambla, Sagrada Familia quarter).

Figure 3.7 – Comparing Airbnb and real-estate offer (Barcelona)



########################## R CODE 8 ##########################

```
library(cartography)
library(sf)
library(SpatialPosition)
library(ggplot2)

folder <- "Barcelona"

# 1 - Import the data and relevant layers
# Data
```

```
grid <- st_read(paste0(folder,"/200.shp"), stringsAsFactors = F) # 200 m grid
airbnb <- read.csv(paste0(folder,"/Airbnb200.csv")) # Airbnb offer
offer <- read.csv(paste0(folder,"/FotoBuy200.csv")) # Real estate offer
grid <- merge(grid, airbnb, by.x = "ID", by.y = "id", all.x = TRUE)
grid <- merge(grid, offer, by.x = "ID", by.y = "id", all.x = TRUE)
names(grid)[7:8] <- c("AIRBNB","REAL_ESTATE")

# Layers (carto)
city <- st_read(paste0(folder, "/layers/CityCS.shp"), stringsAsFactors = F)
lau2 <- st_read(paste0(folder, "/layers/CommunesCS.shp"), stringsAsFactors = F)

# 2 - Data smoothing
## Create dist matrix (manhattan dist )
span <- 500
beta <- 2
method <- "manhattan"
fct <- "exponential"

cGRID <- st_coordinates(st_centroid(st_geometry(grid)))
row.names(cGRID) <- grid$ID
mat <- as.matrix(dist(cGRID, method = method))
# Compute potential
row.names(grid) <- grid$ID

# Delete values below 1
sel_a <- grid$AIRBNB > 1 & !is.na(grid$AIRBNB)
sel_o <- grid$REAL_ESTATE > 1 & !is.na(grid$REAL_ESTATE)

# Span & potential calculation
p_dens_a <- stewart(knownpts = grid[sel_a,], unknownpts = grid[, "ID"],
                    varname = "AIRBNB", matdist = mat,
                    typefct = fct, beta = beta, span = span,
                    returnclass = "sf")
p_dens_o <- stewart(knownpts = grid[sel_o,], unknownpts = grid[, "ID"],
                    varname = "REAL_ESTATE", matdist = mat,
                    typefct = fct, beta = beta, span = span,
                    returnclass = "sf")

grid$pot_AIRBNB <- p_dens_a$OUTPUT
grid$pot_REAL_ESTATE <- p_dens_o$OUTPUT

# Density (offer / sq. kilometer)
grid$pot_AIRBNB <- grid$pot_AIRBNB/4
grid$pot_REAL_ESTATE <- grid$pot_REAL_ESTATE/4

# Delete 0 values
sel <- grid$pot_AIRBNB > 1 & grid$pot_REAL_ESTATE > 1
grid <- grid[sel,]


# 3 - Mapping
# Real estate offers
sizes <- getFigDim(city, width = 800, mar=c(0,0,0,0))

pdf(file = paste0(folder,"/fig/REAL_ESTATE.pdf"),width = sizes[1]/72,
    height = sizes[2]/72, useDingbats=FALSE, pointsize=18)

disc <- quantile(grid$pot_REAL_ESTATE, probs = seq(0,1,0.0625), na.rm = TRUE)

plot(st_geometry(grid), col = NA, border = NA)
plot(st_geometry(lau2), col = "#c6c4c4", border = "white", add =TRUE)

choroLayer(x = grid, var = "pot_REAL_ESTATE", breaks = disc, nclass = 16,
           col = carto.pal(pal1 = "green.pal", n1 = 8, pal2 = "red.pal", n2 = 8),
           border = NA, add = TRUE, legend.pos = "bottom", legend.horiz = TRUE,
           legend.title.txt = paste0("Number of offers in a neighbourhood of ",
           span, " m (deciles)\nSteward potential, ",method," distance, ",
           fct, " function, beta = ",beta),
           legend.values.rnd = 1)

plot(st_geometry(city), col = NA, border = "black", lwd = 0.5, add = T)

layoutLayer(title = "Density - Real estate offers (2019)", scale = 2,
            tabtitle = TRUE,  theme = "red.pal")
dev.off()
```

```
# Airbnb offers
pdf(file = paste0(folder,"/fig/AIRBNB.pdf"),width = sizes[1]/72,
    height = sizes[2]/72, useDingbats=FALSE, pointsize=18)

disc <- quantile(grid$pot_AIRBNB, probs = seq(0,1,0.0625), na.rm = TRUE)

plot(st_geometry(grid), col = NA, border = NA)
plot(st_geometry(lau2), col = "#c6c4c4", border = "white", add =TRUE)

choroLayer(x = grid,var = "pot_AIRBNB", breaks = disc, nclass = 16,
          col = carto.pal(pal1 = "green.pal", n1 = 8, pal2 = "red.pal", n2 = 8),
          border = NA, add = TRUE, legend.pos = "bottom", legend.horiz = TRUE,
          legend.title.txt = paste0("Number of offers in a neighbourhood of ",
                                    span, " m (deciles)\nSteward potential, ",method," distance, ",
                                    fct, " function, beta = ",beta),
          legend.values.rnd = 1)

plot(st_geometry(city), col = NA, border = "black", lwd = 0.5, add = T)

layoutLayer(title = "Density - Airbnb offers (2019)",
           scale = 2,
           tabtitle = TRUE,
           theme = "green.pal")
dev.off()


# 4 - Statistical regression
lin<-lm(grid$pot_AIRBNB ~ grid$pot_REAL_ESTATE)

coeff <- lin$coefficients[2]
int <-  lin$coefficients[1]

txt <- paste0("POT_AIRBNB = ", round(coeff,2),
              " * POT_REAL_ESTATE ", round(int,2))
txt2 <- paste0("R-squared = ", round(summary(lm(grid$pot_AIRBNB~grid$pot_REAL_ESTATE))$r.squared,2))

layout(matrix(1:4,2,2)) > plot(lin2)

pdf(file = paste0(folder,"/fig/LM.pdf"),width = sizes[1]/72,
    height = sizes[2]/72, useDingbats=FALSE, pointsize=25)

ggplot(grid, aes(pot_REAL_ESTATE, pot_AIRBNB)) +
  geom_point(size = 1, colour = "black") +
  stat_smooth(method = "lm") +
  #  geom_smooth(aes(group=transect), method="lm") +
  ylab("Density (hectare) of Airbnb offers - 2018") +
  xlab("Density (hectare) of real-estate offers - 2019") +
  annotate("text", x = max(grid$pot_REAL_ESTATE), y = max(grid$pot_AIRBNB), label = txt, hjust = 2) +
  annotate("text", x = max(grid$pot_REAL_ESTATE), y = max(grid$pot_AIRBNB), label = txt2, hjust = 2) +
  ggtitle("Relationship between Airbnb and real-estate offers")
dev.off()

# 5 - Residuals
grid$estimate_lin <- ((coeff*grid$pot_REAL_ESTATE) + int)
grid$residual_lin <- (grid$pot_AIRBNB - grid$estimate_lin)

pdf(file = paste0(folder,"/fig/RESIDUALS.pdf"),width = sizes[1]/72,
    height = sizes[2]/72, useDingbats=FALSE, pointsize=18)

plot(st_geometry(grid), col = NA, border = NA)
plot(st_geometry(lau2), col = "#c6c4c4", border = "white", add =TRUE)

choroLayer(x = grid, var = "residual_lin",  method = "sd",
          col = carto.pal(pal1 = "blue.pal", n1 = 9, pal2 = "red.pal", n2 = 6),
          border = NA,  add = TRUE,legend.pos = "bottom", legend.horiz = TRUE,
          legend.title.txt = "Residuals of linear model,\n density of Airbnb offer (y) and real estate offer (x)",
          legend.values.rnd = 1)
plot(st_geometry(city), col = NA, border = "black", lwd = 0.5, add = T)

layoutLayer(title = "Residuals",
           scale = 2,
           tabtitle = TRUE,
           theme = "blue.pal")

dev.off()
```

4 Summary of methodological and conceptual outputs. Next steps for further studies.

This project, short by its duration (9 months), allowed for some significant scientific and methodological endeavours and advancement of knowledge, in two respects: gathering and combining real-estate heterogeneous statistical information. It demonstrated also some significant inputs for comparing between neighbourhood and between cities, the current state of housing markets on real-estate market. More than 80 indicators have been produced at LAU2 and grid scales related to real-estate market. Looking at the current data availability at European scale (few indicators, only available at FUA or Core city level), it is indubitable that it provides some advances to go far away for measuring wellbeing of European citizens as regards to the real-estate market. From our point of view, the main methodological progresses allowed by this study are:

- **The creation of an operational methodological framework for producing comparable analysis between case-study cities**. As far as we know, this project is the first one which tries to define a common methodological framework for the analysis of real-estate market in an international perspective at this very local scale. The definition of relevant data sources, the data collection process (targeting basic indicators), the data cleaning and the data harmonisation process (harmonised indicator of high policy value and spatial smoothing to manage the MAUP effects) is a solid foundation upon which further studies can elaborate upon. Moreover, the integration of the methodological framework in a R standard code workflow ensures the reproducibility of the inputs provided. It opens also the door to further methodological improvements (statistical analysis, data cleaning and so on).

- **It combined conventional and unconventional data sources, from institution or commercial providers**. This was clearly one important challenge to overcome in a very short amount of time. This was realised both by using real-estate agents data sources and data coming from the Web (Airbnb offers) to put into perspective data coming from institutional data sources (transaction and income data). Maps realised show concrete spatial structures and territorial discontinuities inside and between the studied cities.

- **Multiscalar perspective for feeding the policy debate.** More than 100 maps have been created for 10 selected case-studies. These maps, based on local observations (the level of the transaction or the real-estate offer) offer **innovative insights on socio-economic dynamics and challenges inside and between European cities**. Most of the studies on European cities are limited, due to statistics availability, at the scale of the city (EU FUA or core cities). These new elements brought here are in capacity to feed the policy debate on inequalities inside the city. For instance at EU level, Paris is often considered as an example: smart, open to the World, creative, etc. The elements brought here also demonstrate, regarding housing market, that

Paris and its Functional Urban Area is also very unequal: at LAU2 level, advertised prices for property per square/meters in 2019 (l*eboncoin.fr*) ranges from 1,430 euros/sq meter in the North-Western part of Paris FUA (Montreuil-sur-Epte), up to 16,000 euros (Paris 6th arrondissement). With reference to the level of wealth of local population (median income), buying 1sq. meter corresponds to 0.8 months of income (minimum value) in Grigny (Southern suburbs), up to 9 months in Paris (6th arrondissement). The situation on the rental market is similar: renting 1sq. meter (advertised price) corresponds to less than 10 euros per month in the periphery (Coulombs, Cerny) to more than 40 euros in the inner city or its immediate suburbs (Clamart, Western part of Paris). From our point of view, it is clearly by combining the scale of analysis (EU level, city level, intra-urban level) that it will be possible to provide policy makers with a better understanding of the challenges raised by real-estate dynamics for accessing to descent housing near to the center of economic and employment basins.

Nevertheless, studies on international comparisons on spatial effects of housing inequalities are just beginning. Given the time-frame of the data collection and preparation of the Wellbeing report, some items should be considered for further studies, covering both data collection and analysis issues:

- **We pathed the way to extend the analysis to other case-studies.** Gathering data for all the European cities would be obviously the ultimate target, but also a dream. Indeed, this study has demonstrated – starting from level 0 of data availability at European level – that it will not be possible without significant resources and funding. From institutional perspective, it would require to **identify the most appropriate data providers** and **negotiate the access** to the transaction data for each country of Europe. In this case, data have been bought with researches agreements outside the project for a city selection (Chamber of Notaries in France, Townhall transactions taxes for Barcelona, real estate price register in Poland). The main difficulty here, outside the cost of the data is also first to manage the territorial level of access to the data (BIEN database covers only Ile-de-France, PERVAL the rest of France. In Spain data are available at municipal level and not for the entire country); and secondly to obtain comparable data between this high heterogeneity of data providers. As demonstrated in this document, the BIEN database covers 96 indicators and makes possible in-depth analysis on the socio-economic characteristics of the seller or the buyer. For other countries, like in the UK, only the point location of the transaction and the price is available. In this context, using web-scraped methods can be considered as an adapted alternative. This study has demonstrated that it can be true to some extent. Some relevant statistics may be produced. But the amount of work it induces is high: defining relevant real-estate agents, creating scraping procedures (1

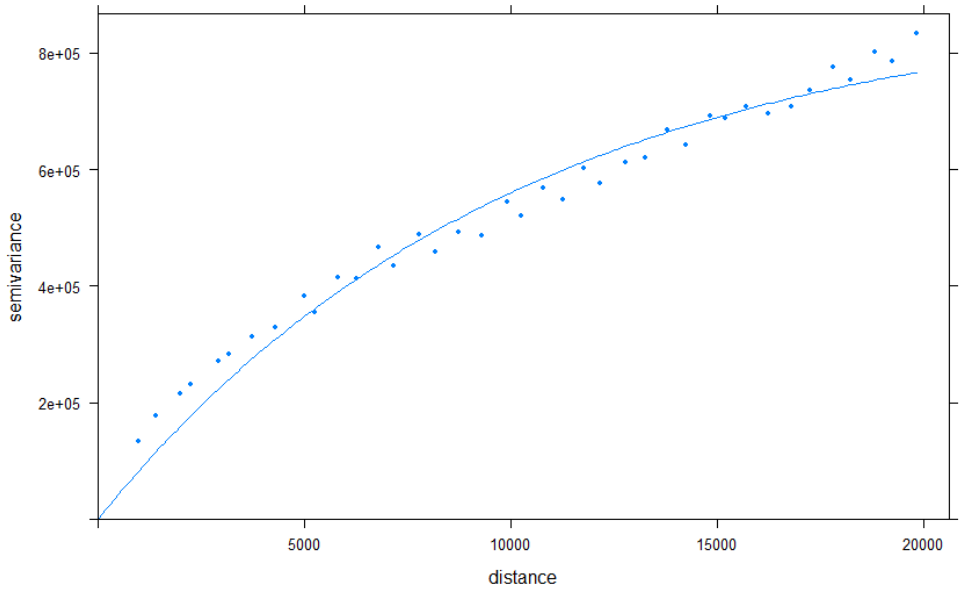
by data provider) and data cleaning is not an easy process. And modeling it at the scale of a research project is not necessary the best solution. However, transferring these procedures of data collection and cleaning to national and/or EU statistical institutions may be a sustainable solution in the long term.

- **The scraping and web-harvesting process has proven to be costly** and does not always insure a cost-effective way to gather significant data, in a non-commercial data aggregation effort. The code further requires constant improvement and technological follow-up to be useable with future implementation of the targeted websites: the web-mining effort involves a constant manning of the process.
- How to integrate **the time dimension and compare institutional and unconventional data sources.** Even if data have been collected, going more in-depth in cross-analysis of these data sources should be promising, as demonstrated by the comparison of Airbnb offer and real-estate offer. In that perspective, comparing the relationship between real-estate transactions and offers should be considered. First of all, it allows to statistically check the validity of real-estate offers, as a proxy of the real-estate market. Several issues may be explored: for instance the analysis of the density of offers as compared to the density of real transactions may reveal some interesting patterns. Some areas may be characterized by an over-representation of offers, as regards to real-estate actual transactions. It may suggest some disinterest for some specific areas. Reversely, over-representation of real-estate transactions as regards to offer may suggest a pressurized real-estate market (lack of offers). Secondly, the analysis of the relationship between advertised price and price paid (or price paid towards the time) may also be explored. This kind of analysis raises among others the issue of the real-estate market structure and evolution: Where advertised prices are growing, all things being equal to price paid some years ago? Are the trends observed towards the time homogeneous in the urban area, or not? Are these real-estate trends correlated to socio-economic characteristics, or not (gentrification issues)?
- **Further studies should systematically differentiate apartments and houses for the analysis.** Except for Paris (only apartments have been considered), analysis are provided for houses and apartments combined in the analysis. Also, the segmentation of residential markets does not systematically compares between different cities. In fact, at the scale of the Functional Urban Areas, the analysis should be more focused on apartments, which represents the largest part of the housing park. When looking into the suburbs of the FUA, the housing park is on average more structured by individual housing, which does not follow the same real-estate market segmentation and rental/buying structure. The fact to take into account houses in the analysis raises also some methodological issues which must be clarified to create consistent datasets: to what extent does it make sense to measure the surface of a

house for an evaluation of the price per sq.meters? Does it requires to take into account the garden or not? Does the data providers (institutional data providers, real-estate agents) differentiate living space and the garden in real-estate transactions database and ads? Not systematically: this is another drawback of using unconventional “big data” sources.

- **Go far away for data smoothing.** Harmonized maps created used basically the price/square meter parameter and homogeneous functions parameters, based on literature review (Pareto function, Manhattan distance, beta = 2, span = 2000m) to interpolate real-estate values in the space. Results are promising but could at least be extended in two different ways. First, elaborating further on the analysis of interpolation methods and how to implement the distance decay function between transactions. It can be explored with the use of variograms (Figure 4.8). In spatial statistics, this function describes the degree of spatial dependence of a spatial random field. In practice, the use of variogram in the context of real-estate analysis may propose interesting perspectives for a better understanding of the local real-estate market structure and could help to better define appropriate parameters for better modeling the Stewart potential used to interpolate price in the space.

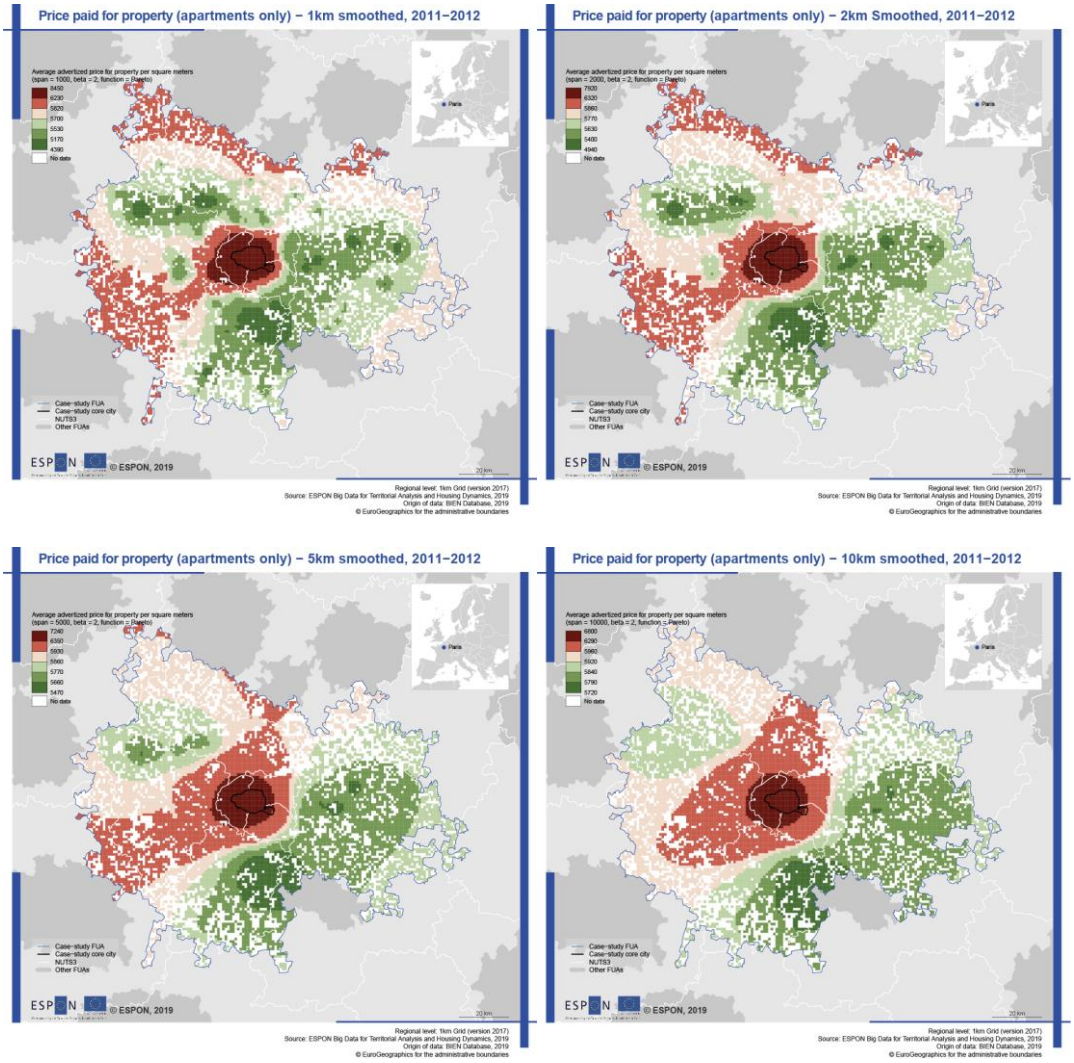
Figure 4.8 – Variogram of advertised price of real-estate property prices (euros/sq. kilometer) in Barcelona according to distance



Secondly, there is no unique scale at which should be defined the spatial extent of “higher prices” or “lower prices”. Each type of economic and social interaction indeed spans specific geographic distances. Each of the scales of analysis may be relevant and closely pertains to urban contexts and also stakeholders. Obviously for real-estate agents or people interested in buying a real-estate property the local context (Figure 4.9, top left, span = 1km) is the most relevant. For urban planners or policy

makers the intermediate level (2-5km) gives important insights on the local structure of the market and gives overall information on intra-urban discontinuities and inequalities. At European level, smoothed map with a more generalized information provides significant insights on the global organization of real-estate market in Functional Urban Areas, and is certainly more adapted for international comparisons. This first attempt may be explored more in depth in the future, and requires sometimes to return on the data structure to identify if exceptional values observed corresponds to the reality or due to statistical artifacts (outliers in specific locations).

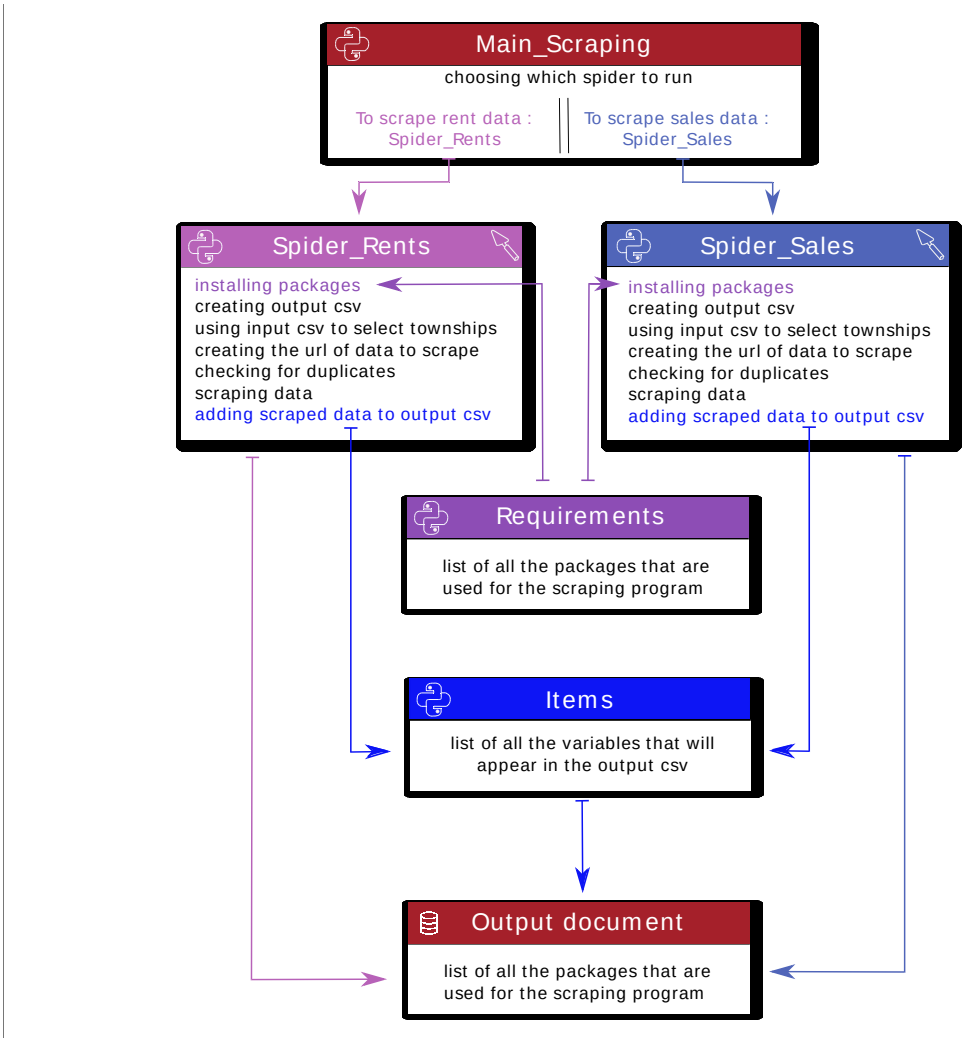
Figure 4.9 – Average real-estate transaction prices (euros per square meters) in Paris interpolated at several geographical scales (span parameter = 1000m, 2000m, 5000m and 10000m)



Annex 1 – Python scripts used to harvest French real-estate website

The python scraping program developed elaborates on **Scrapy**, a python library mostly focused on data harvesting. Scrapy runs using different python files, and a main organisation that the user can change to fit his needs. The user sets "**spiders**" that crawl through selected **webpages** and gather information required by the user, using the html code of said pages (see figure A.1). Each item of the following layout is a python file who either holds information that is usefull for scraping or crawls the webpages to gather information.

Figure A.1 - Scraping webpages using Scrapy



#################### MAIN_SCRAPING ####################

```
import scrapy
# twisted is a python library that is used to run scrapy
from twisted.internet import reactor, defer
from scrapy.crawler import CrawlerRunner
from scrapy.utils.log import configure_logging

#https://doc.scrapy.org/en/latest/topics/practices.html
```

```
from boncoin_project.spiders import Cities
from boncoin_project.spiders import locations
configure_logging()
runner = CrawlerRunner()

# A runner is defined to send spiders to crawl on the webpages
# here you can choose which spider to use (crawl rent or sales offers)
@defer.inlineCallbacks
def crawl():
    yield runner.crawl(Cities.Cities)
    yield runner.crawl(locations.Locations)
    yield runner.crawl(another spyder)
    reactor.stop()
crawl()
reactor.run()
```

```
 ##############################################################################

 ####################                REQUIREMENTS            ####################
```

```
cfscrape
fake-useragent
beautifulsoup4
pendulum
scrapy
csvkit
pandas
```

```
 ##############################################################################

  ####################                ITEMS                ####################
```

```
# Define here the models for your scraped items
# See documentation in:
# http://doc.scrapy.org/en/latest/topics/items.html

import scrapy

# allows us to import in the csv file the information that has been scraped online

# the AnnoncesSales class is used to store the data collected on household sales
class AnnoncesSale(scrapy.Item):
    annoncet = scrapy.Field()
    logprix = scrapy.Field()
    url = scrapy.Field()
    #descr = scrapy.Field()
    loghonoraires = scrapy.Field()
    logtypebien = scrapy.Field()
    lognbpieces = scrapy.Field()
    logsurface = scrapy.Field()
    energieclass = scrapy.Field()
    energieges = scrapy.Field()
    annonceh = scrapy.Field()
    annonced = scrapy.Field()
    logcodepost = scrapy.Field()
    logville = scrapy.Field()
    scrapdate = scrapy.Field()
    scrapheure = scrapy.Field()
    ID_URL = scrapy.Field()
    agenceimmonom = scrapy.Field()
    agenceimmoadresse = scrapy.Field()
    agenceimmosiret = scrapy.Field()
    agenceimmosiren = scrapy.Field()
    agenceimmotel = scrapy.Field()
    insee = scrapy.Field()
    typetransaction = scrapy.Field()

# the AnnonceLoc class is used to store the data collected on houses and apartments rentals
class AnnoncesRental(scrapy.Item):
    annoncet = scrapy.Field()
    logprix = scrapy.Field()
    url = scrapy.Field()
    #descr = scrapy.Field()
    logcharges = scrapy.Field()
```

```
    logtypebien = scrapy.Field()
    lognbpieces = scrapy.Field()
    logsurface = scrapy.Field()
    energieclass = scrapy.Field()
    energieges = scrapy.Field()
    annonceh = scrapy.Field()
    annonced = scrapy.Field()
    logcodepost = scrapy.Field()
    logville = scrapy.Field()
    scrapdate = scrapy.Field()
    scrapheure = scrapy.Field()
    ID_URL = scrapy.Field()
    agenceimmonom = scrapy.Field()
    agenceimmoadresse = scrapy.Field()
    agenceimmosiret = scrapy.Field()
    agenceimmosiren = scrapy.Field()
    agenceimmotel = scrapy.Field()
    insee = scrapy.Field()
    typetransaction = scrapy.Field()

 ##########################################################################

    ####################          SPIDER_RENT          ####################

# SPIDER
# used to scrape only rentals

# list of required packages to successfully run the spider
from bs4 import BeautifulSoup
import scrapy
import cfscrape
from fake_useragent import UserAgent
import pendulum
import json
import random
from boncoin_project.items.Items import *
import csvkit
import pandas as pd

class Locations(scrapy.Spider):
    name = "locations"
    # specifying already existing settings, this part can filled out by the user
    custom_settings = {
        # gives the number of concurrent requests that will be run at the same time, to avoid detection on the
scraped
        # website, it is best to use one request at a time
        'CONCURRENT_REQUESTS': '1',
        # allows us to set how long each request will take. If our program scraps too quickly, it might get
detected by
        # the website security. For Leboncoin, two seconds for each request should work fine.
        'DOWNLOAD_DELAY':'2',
        'COOKIES_ENABLED': True,
        # HTTPERROR_ALLOWED_CODES allows certain errors, meaning the program keeps running after
encountering them
        'HTTPERROR_ALLOWED_CODES':[404],
        'FEED_EXPORTERS': {
            'csv': 'scrapy.exporters.CsvItemExporter'},
        # allows us to choose the type of file that is produced in output by the program
        'FEED_FORMAT' : 'csv',
        # allows us to choose the encoding of the file in output
        'FEED_EXPORT_ENCODING' : 'utf-8',
        # allows us to choose the name of the file in output
        'FEED_URI' : 'Locations_Com_Out_10.csv',
        'DEFAULT_REQUEST_HEADERS': {
            # prepares the user-agent, the id of the program while it scraps information from the website

            'User-agent' : 'Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:62.0) Gecko/20100101 Firefox/62.0',
        'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
        'Accept-Language': 'fr,fr-FR;q=0.8,en-US;q=0.5,en;q=0.3'},
    }
    # defines the web page where our spider starts scraping
    start_urls = ['https://www.leboncoin.fr/recherche/?category=10&real_estate_type=1,2']
    # specifies the domain of the website to scrap using this specific spider
    allowed_domains = ['leboncoin.fr']
```

```
    # calls the user-agent that is specified in the settings of our spider
    ua = 'Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:62.0) Gecko/20100101 Firefox/62.0'

    # allows us to get the date and time for each scraped item
    def __init__(self, aDate = pendulum.today()):
        super(Locations, self).__init__()
        self.aDate = aDate
        self.timestamp = self.aDate.timestamp()
        print("PENDULUM UTC TODAY", self.aDate.today())
        print("PENDULUM UTC TIMESTAMP TODAY ", self.timestamp)

    # allows us to use beautifulSoup, a python library used to pull data from webpages
    def clean_html(self, html_text):
        soup = BeautifulSoup(html_text, 'html.parser')
        return soup.get_text()

    # launches the first request, using the start url defined earlier
    def start_requests(self):
        for url in self.start_urls:
            yield scrapy.Request(url=url, callback = self.parse_page)

    # allows us to build the url of each township we want to scrap information about
    def parse_page(self, response):
        # we first import the csv containing the names and postal codes of the townships we wish to get
information on
         csvcible1 = pd.read_csv('All_Com_IN_6.csv')
        # then, for each row in our input file (csvcible1) we build the url according to leboncoins' structure :
         for index, row in csvcible1.iterrows():
            # ligne1 will host the ID_URL value for the selected row; ID_URL is the concatenation of the name of
the township(for exemple "Paris"), of "_" and of its postal number (for exemple "75001") so that the output
fits leboncoins' structure ("Paris_75001")
            ligne1 = row["ID_URL"]
            # csvinsee is calling "LAU2_" , which is the insee code of the selected township; this code is of no
use for scraping but must be kept in order to use it later for merging purposes
            csvinsee = row["LAU2_"]
            print(ligne1)
            print(csvinsee)
            # urllen creates, by concatenation, a fake url that could be similar to one of Leboncoin containing all
the offers for a specific township. If this url already exists in Leboncoin, it will possible to go further and
scrap the data that it contains. If this URL doesn't exist, the program will skip to the next line of the input
document
            urllen = 'https://www.leboncoin.fr/recherche/?category=10&locations=' + ligne1 + '&real_estate_type=1,2'
            print(urllen)
            yield scrapy.Request(url=urllen, callback=self.parse_nbpages, meta={"ligne1": ligne1,
                                                    "csvinsee": csvinsee})

    # this function allows us to look for the right number of pages, to avoid leaving out offers and to avoid
scraping
    # empty pages (see Leboncoin documentation)
    def parse_nbpages (self, response):
        # first, we call back previous information (the "ID_URL" and the insee code from the previous function
        ligne1 = response.meta['ligne1']
        csvinsee = response.meta['csvinsee']
        # we start scraping information : here, the number of offers per township
        nbannonces = str(' '.join(response.xpath('//p/span[@class="_2ilNG"]/text()').extract()).replace(" ", ""))
        # we change the format of the information scraped to "integer' to reuse it easier
        nbpages = int(nbannonces)
        print(nbpages)
        # we divide the number of offers by 35 (the maximum number of offers by offers page) and we add 2 to
also have
        # the offers that could be on the last page with less than 35 offers
        nbpages = round(nbpages/35)+2
        print(nbpages)
        # this loop is used to obtain, for each offers page, the correct number of url that will be needed for the
rest of the scraping
        for p in range(1,nbpages):
            print(p)
            # the scraped url here is modified to stretch the search for the maximum number of offers page as far
as there arer still offers pages for each specific township
            urls = 'https://www.leboncoin.fr/recherche/?category=10&locations=' + ligne1 + '&real_estate_type=1,2'+\
                '&page='+ str(p)
            yield scrapy.Request(url = urls, callback = self.parse, priority=1, meta= {"ligne1" : ligne1,
                                                    "csvinsee":csvinsee})
```

```
    # This function allows us to get the exact number of offers per offers page, to get the url of each individual offer, and to make sure that said offer has not been scraped before
    def parse(self, response):
        # the "clearfix trackable object is the unique id of each offer, and is used to create the unique URL of each offer on Leboncoin
        extra = response.xpath('//a[@class="clearfix trackable"]/@href').extract()
        ligne1 = response.meta['ligne1']
        csvinsee = response.meta['csvinsee']
        for i in extra:
            # by concatenation, we bring together leboncoin standard url start ""https://www.leboncoin.fr" and the unique id (the "clearfix trackable" mention before) to get the unique offers' URL
            url2 = "https://www.leboncoin.fr"+i
            annonce = AnnoncesRental()
            annonce['url'] = url2
            annonce['ID_URL'] = ligne1
            print(url2)
            # we check if this offer has been scraped before by using the url we have just created. If it has, it is ignored, if is not be scraped before, it is scraped now
            if url2 not in open('Locations_Com_Out_8.csv', encoding='utf8').read():
                yield scrapy.Request(url=url2, callback=self.parse_annonce,meta={"annonce": annonce,
                                                                                 "csvinsee": csvinsee})
            else:
                print("Déjà scrapé")

    # this last function scraps the information we need from each offer and prepares it to be added to the output csv
    def parse_annonce(self, response):

        # we call information gathered or created previously in the program
        annonce = response.meta['annonce']
        csvinsee = response.meta['csvinsee']

        # we specify if the advertised good is for sale or for rent and create a new column in the output csv that contains that information
        annonce['typetransaction'] = "Location"

        # using the input csv, whe get the insee code of the township in which is located the good mentioned in the offer
        annonce['insee'] = csvinsee

        # scraping the title of the offer
        annonce['annoncet'] = ' '.join(response.xpath('//h1[@class="_246DF _2S4wz"]/text()').extract()).replace(";"," ")

        # scraping the rental price of the good
        annonce['logprix'] = response.xpath('//span[@class ="_1F5u3"]/text()').extract()[0]

        # scraping the full text that the user typed to describe the good
        annonce['descr'] = ' '.join(response.xpath('//span[@class ="content-CxPmi"]/text()').extract()).replace(";"," ")

        # time and date of the scraping of the this particular offer
        annonce['scrapdate'] = self.aDate.today().to_date_string().replace("-","/")
        annonce['scrapheure'] = self.aDate.today().to_time_string() #.replace(":","/")

        # scraping if there are any additional fees (electricty, water, etc. not included in the rental price
        annonce['logcharges'] = response.xpath('//div[@class="_2B0Bw _1nLtd"]//text()').extract()[1]

        # scraping what type of good it is (house or apartment)
        annonce['logtypebien'] = response.xpath('//div[@data-qa-id="criteria_item_real_estate_type"]/div/div[2]/text()')\
            .extract()

        # scraping the number of rooms
        annonce['lognbpieces'] = response.xpath('//div[@data-qa-id="criteria_item_rooms"]/div/div[2]/text()').extract()

        # scraping the area of the good (in square meters)
        annonce['logsurface'] = ' '.join(response.xpath('//div[@data-qa-id="criteria_item_square"]/div/div[2]/text()')\
                                .extract())[:-2]

        # scraping the energy class of the good ("A", "B", "C", "D", "E", "F" or "G")
        annonce['energieclass'] = response.xpath('//div[@class="_2Fdg_1kx3G"]/div[contains(@class,"_1sd0z")]/text()')\
            .extract()

        # scraping the Greenhouse gas emission category if the good ("A", "B", "C", "D", "E", "F" or "G")
        annonce['energieges'] = response.xpath('//div[@class="_2Fdg- QGdfG"]/div[contains(@class,"_1sd0z")]/text()')\
            .extract()
```

```
        # scraping the time where the offer was submitted
        annonce['annonceh'] = ' '.join(response.xpath('//div[@data-qa-id="adview_date"]/text()').extract())[-5:]

        # scraping the date where the offer was submitted
        annonce['annonced'] = ' '.join(response.xpath('//div[@data-qa-id="adview_date"]/text()').extract())[:10]

        # for a few particular townships, it was necessary to specify exceptions in order to be able to scrap
them
        # It was also believed necessary to obtain in the output file the name in full caps of each township
scraped
        listeSaints = ('Lieusaint','Saints','Saintry-sur-Seine', 'Saintlieu', 'Saint-Symphorien-le-Château',
                    'Saint-Maur-de-Fossés', 'Saints', "Saint-Cyr-l'Ecole")
        if response.xpath('//div[@data-qa-id="adview_location_informations"]//text()').extract()[0] not in listeSaints:
            annonce['logville'] = response.xpath('//div[@data-qa-id="adview_location_informations"]//text()')\
            .extract()[0]\
            .replace("á", "a").replace("à", "a").replace("â", "a").replace("ä", "a") \
            .replace("é", "e").replace("è", "e").replace("ê", "e").replace("ë", "e") \
            .replace("í", "i").replace("ì", "i").replace("î", "i").replace("ï", "i") \
            .replace("ó", "o").replace("ò", "o").replace("ô", "o").replace("ö", "o") \
            .replace("ú", "u").replace("ù", "u").replace("û", "u").replace("ü", "u") \
            .replace("'", " ").replace("-", " ").replace("É", "e").replace("ç", "c") \
            .replace("Ç ","c").replace("Ÿ","Y").replace("ÿ","y").replace("Î","I")\
            .upper().replace(" SAINT "," ST ").replace(" SAINTS "," ST ").replace("-SAINT ", "-ST ")\
            .replace("-SAINTS ", "-ST ").replace(" SAINT-"," ST-").replace(" SAINTS-"," ST-") \
            .replace("-SAINT-", "-ST-").replace("-SAINTS-", "-ST-").replace("SAINT","ST")\
            .replace("SAINTS","ST")
        elif response.xpath('//div[@data-qa-id="adview_location_informations"]//text()').extract()[0] in listeSaints:
            annonce['logville'] = response.xpath('//div[@data-qa-id="adview_location_informations"]//text()').extract()[0] \
            .replace("á", "a").replace("à", "a").replace("â", "a").replace("ä", "a") \
            .replace("é", "e").replace("è", "e").replace("ê", "e").replace("ë", "e") \
            .replace("í", "i").replace("ì", "i").replace("î", "i").replace("ï", "i") \
            .replace("ó", "o").replace("ò", "o").replace("ô", "o").replace("ö", "o") \
            .replace("ú", "u").replace("ù", "u").replace("û", "u").replace("ü", "u") \
            .replace("Ç ","c").replace("'", " ").replace("-", " ").replace("É", "e").replace("ç", "c").upper()

        # finally, scraping the postal code of each offer
        annonce['logcodepost'] = str(response.xpath('//div[@data-qa-
id="adview_location_informations"]//text()').extract()[2]).zfill(5)

        yield annonce
```

##

########## SPIDER_SALES ##########

```
# SPIDER
# used to scrape only sales

# list of required packages to successfully run the spider
from bs4 import BeautifulSoup
import scrapy
import cfscrape
from fake_useragent import UserAgent
import pendulum
import json
import random
from boncoin_project.items.Items import *
import csvkit
import pandas as pd

class Cities(scrapy.Spider):
    name = "cities"
    # specifying already existing settings, this part can filled out by the user
    custom_settings = {
        # gives the number of concurrent requests that will be run at the same time, to avoid detection on the
scraped website, it is best to use one request at a time
        'CONCURRENT_REQUESTS': '1',
        # allows us to set how long each request will take. If our program scraps too quickly, it might get
detected by the website security. For Leboncoin, two seconds for each request should work fine.
        'DOWNLOAD_DELAY': '2',
        'COOKIES_ENABLED': True,
        # HTTPERROR_ALLOWED_CODES allows certain errors, meaning the program keeps running after
encountering them
        'HTTPERROR_ALLOWED_CODES': [404],
        'FEED_EXPORTERS': {
```

```
        'csv': 'scrapy.exporters.CsvItemExporter'},
    # allows us to choose the type of file that is produced in output by the program
    'FEED_FORMAT': 'csv',
    # allows us to choose the encoding of the file in output
    'FEED_EXPORT_ENCODING': 'utf-8',
    # allows us to choose the name of the file in output
    'FEED_URI': 'All_Com_OUT_11.csv',
    # prepares the user-agent, the id of the program while it scraps information from the website
    'DEFAULT_REQUEST_HEADERS': {
        'User-agent': 'Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:62.0) Gecko/20100101 Firefox/62.0',
        'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
        'Accept-Language': 'fr,fr-FR;q=0.8,en-US;q=0.5,en;q=0.3'},
}
# defines the web page where our spider starts scraping
start_urls = ['https://www.leboncoin.fr/recherche/?category=9&locations=d_75&real_estate_type=1,2,3']
# specifies the domain of the website to scrap using this specific spider
allowed_domains = ['leboncoin.fr']

# calls the user-agent that is specified in the settings of our spider
ua = 'Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:62.0) Gecko/20100101 Firefox/62.0'

# allows us to get the date and time for each scraped item
def __init__(self, aDate=pendulum.today()):
    super(Cities, self).__init__()
    self.aDate = aDate
    self.timestamp = self.aDate.timestamp()
    print("PENDULUM UTC TODAY", self.aDate.today())
    print("PENDULUM UTC TIMESTAMP TODAY ", self.timestamp)
    # binary = FirefoxBinary('path/to/binary')
    # self.driver = webdriver.Firefox(firefox_binary=binary)

# allows us to use beautifulSoup, a python library used to pull data from webpages
def clean_html(self, html_text):
    soup = BeautifulSoup(html_text, 'html.parser')
    return soup.get_text()

# launches the first request, using the start url defined earlier
def start_requests(self):
    for url in self.start_urls:
        yield scrapy.Request(url=url, callback=self.parse_page)

# allows us to build the url of each township we want to scrap information about
def parse_page(self, response):
    # we first import the csv containing the names and postal codes of the townships we wish to get
information on
    csvcible1 = pd.read_csv('All_Com_IN_6.csv')
    # then, for each row in our input file (csvcible1) we build the url according to leboncoins' structure :
    for index, row in csvcible1.iterrows():
        # ligne1 will host the ID_URL value for the selected row; ID_URL is the concatenation of the name of
the township(for exemple "Paris"), of "_" and of its postal number (for exemple "75001") so that the output
fits leboncoins' structure ("Paris_75001")
        ligne1 = row["ID_URL"]
        # csvinsee is calling "LAU2_" , which is the insee code of the selected township; this code is of no
use for scraping but must be kept in order to use it later for merging purposes
        csvinsee = row["LAU2_"]
        print(ligne1)
        print(csvinsee)
        # urllen creates, by concatenation, a fake url that could be similar to one of Leboncoin containing all
the offers for a specific township. If this url already exists in Leboncoin, it will possible to go further and
scrap the data that it contains. If this URL doesn't exist, the program will skip to the next line of the input
document
        urllen = 'https://www.leboncoin.fr/recherche/?category=9&locations=' + ligne1 + '&real_estate_type=1,2'
        print(urllen)
        yield scrapy.Request(url=urllen, callback=self.parse_nbpages, meta={"ligne1": ligne1,
                                                                            "csvinsee": csvinsee})

# this function allows us to look for the right number of pages, to avoid leaving out offers and to avoid
scraping empty pages (see Leboncoin documentation)
def parse_nbpages(self, response):
    # first, we call back previous information (the "ID_URL" and the insee code from the previous function
    ligne1 = response.meta['ligne1']
    csvinsee = response.meta['csvinsee']
    # we start scraping information : here, the number of offers per township
    nbannonces = str(' '.join(response.xpath('//p/span[@class="_2ilNG"]/text()').extract()).replace(" ", ""))
    # we change the format of the information scraped to "integer' to reuse it easier
```

```
        nbpages = int(nbannonces)
        print(nbpages)
        # we divide the number of offers by 35 (the maximum number of offers by offers page) and we add 2 to
also have the offers that could be on the last page with less than 35 offers
        nbpages = round(nbpages / 35) + 2
        print(nbpages)
        # this loop is used to obtain, for each offers page, the correct number of url that will be needed for the
rest of the scraping
        for p in range(1, nbpages):
            print(p)
            # the scraped url here is modified to stretch the search for the maximum number of offers page as far
as there arer still offers pages for each specific township
            urls = 'https://www.leboncoin.fr/recherche/?category=9&locations=' + ligne1 + '&real_estate_type=1,2' + \
                '&page=' + str(p)
            yield scrapy.Request(url=urls, callback=self.parse, priority=1, meta={"ligne1": ligne1,
                                                                    "csvinsee": csvinsee})

    # This function allows us to get the exact number of offers per offers page, to get the url of each individual
offer, and to make sure that said offer has not been scraped before
    def parse(self, response):
        # the "clearfix trackable object is the unique id of each offer, and is used to create the unique URL of
each offer on Leboncoin
        extra = response.xpath('//a[@class="clearfix trackable"]/@href').extract()
        ligne1 = response.meta['ligne1']
        csvinsee = response.meta['csvinsee']
        for i in extra:
            # by concatenation, we bring together leboncoin standard url start ""https://www.leboncoin.fr" and
the unique id (the "clearfix trackable" mention before) to get the unique offers' URL
            url2 = "https://www.leboncoin.fr" + i
            annonce = AnnoncesSale()
            annonce['url'] = url2
            annonce['ID_URL'] = ligne1
            print(url2)
            # we check if this offer has been scraped before by using the url we have just created. If it has, it is
ignored, if is not be scraped before, it is scraped now
            if url2 not in open('Ventes_Com_Out_7.csv', encoding='utf8').read():
                yield scrapy.Request(url=url2, callback=self.parse_annonce, meta={"annonce": annonce,
                                                                    "csvinsee": csvinsee})
            else:
                print("Déjà scrapé")

    # this last function scraps the information we need from each offer and prepares it to be added to the
output csv
    def parse_annonce(self, response):

        # we call information gathered or created previously in the program
        annonce = response.meta['annonce']
        csvinsee = response.meta['csvinsee']

        # we specify if the advertised good is for sale or for rent and create a new column in the output csv that
contains that information
        annonce['typetransaction'] = "Vente"

        # using the input csv, whe get the insee code of the township in which is located the good mentioned in
the offer
        annonce['insee'] = csvinsee

        # scraping the title of the offer
        annonce['annoncet'] = ' '.join(response.xpath('//h1[@class="_246DF _2S4wz"]/text()').extract()).replace(";",
                                                                                " ")

        # scraping the price of the good for sale
        annonce['logprix'] = response.xpath('//span[@class ="_1F5u3"]/text()').extract()[0]

        # scraping the full text that the user typed to describe the good
        annonce['descr'] = ' '.join(response.xpath('//span[@class ="content-CxPmi"]/text()').extract()).replace(";"," ")

        # time and date of the scraping of the this particular offer
        annonce['scrapdate'] = self.aDate.today().to_date_string().replace("-", "/")
        annonce['scrapheure'] = self.aDate.today().to_time_string()  # .replace(":","/") (SI BESOIN)

        # scraping if they are fees associated with the transaction ("YES", "NO", "No data")
        annonce['loghonoraires'] = response.xpath('//div[@data-qa-id="criteria_item_fai_included"]/div/div[2]/text()') \
            .extract()
        # scraping what type of good it is (house or apartment)
```

```
        annonce['logtypebien'] = response.xpath('//div[@data-qa-id="criteria_item_real_estate_type"]/div/div[2]/text()') \
            .extract()

        # scraping the number of rooms
        annonce['lognbpieces'] = response.xpath('//div[@data-qa-id="criteria_item_rooms"]/div/div[2]/text()').extract()

        # scraping the area of the good (in square meters)
        annonce['logsurface'] = ' '.join(response.xpath('//div[@data-qa-id="criteria_item_square"]/div/div[2]/text()')
                                         .extract())[:-2]

        # scraping the energy class of the good ("A", "B", "C", "D", "E", "F" or "G")
        annonce['energieclass'] = response.xpath('//div[@class="_2Fdg-
_1kx3G"]/div[contains(@class,"_1sd0z")]/text()') \
            .extract()

        # scraping the Greenhouse gas emission category if the good ("A", "B", "C", "D", "E", "F" or "G")
        annonce['energieges'] = response.xpath('//div[@class="_2Fdg- QGdfG"]/div[contains(@class,"_1sd0z")]/text()') \
            .extract()

        # scraping the time where the offer was submitted
        annonce['annonceh'] = ' '.join(response.xpath('//div[@data-qa-id="adview_date"]/text()').extract())[-5:]

        # scraping the date where the offer was submitted
        annonce['annonced'] = ' '.join(response.xpath('//div[@data-qa-id="adview_date"]/text()').extract())[:10]

        # for a few particular townships, it was necessary to specify exceptions in order to be able to scrap
them
        # It was also believed necessary to obtain in the output file the name in full caps of each township
scraped
        listeSaints = ('Lieusaint', 'Saints', 'Saintry-sur-Seine', 'Saintlieu', 'Saint-Symphorien-le-Château',
                       'Saint-Maur-de-Fossés', 'Saints', "Saint-Cyr-l'Ecole")
        if response.xpath('//div[@data-qa-id="adview_location_informations"]//text()').extract()[0] not in listeSaints:
            annonce['logville'] = response.xpath('//div[@data-qa-id="adview_location_informations"]//text()') \
                .extract()[0] \
                .replace("á", "a").replace("à", "a").replace("â", "a").replace("ä", "a") \
                .replace("é", "e").replace("è", "e").replace("ê", "e").replace("ë", "e") \
                .replace("í", "i").replace("ì", "i").replace("î", "i").replace("ï", "i") \
                .replace("ó", "o").replace("ò", "o").replace("ô", "o").replace("ö", "o") \
                .replace("ú", "u").replace("ù", "u").replace("û", "u").replace("ü", "u") \
                .replace("'", " ").replace("-", " ").replace("É", "e").replace("ç", "c") \
                .replace("Ç ", "c").replace("Ÿ", "Y").replace("ÿ", "y").replace("Î", "I") \
                .upper().replace(" SAINT ", " ST ").replace(" SAINTS ", " ST ").replace("-SAINT ", "-ST ") \
                .replace("-SAINTS ", "-ST ").replace(" SAINT-", " ST-").replace(" SAINTS-", " ST-") \
                .replace("-SAINT-", "-ST-").replace("-SAINTS-", "-ST-").replace("SAINT", "ST") \
                .replace("SAINTS", "ST")
        elif response.xpath('//div[@data-qa-id="adview_location_informations"]//text()').extract()[0] in listeSaints:
            annonce['logville'] = response.xpath('//div[@data-qa-id="adview_location_informations"]//text()') \
                .extract()[0] \
                .replace("á", "a").replace("à", "a").replace("â", "a").replace("ä", "a") \
                .replace("é", "e").replace("è", "e").replace("ê", "e").replace("ë", "e") \
                .replace("í", "i").replace("ì", "i").replace("î", "i").replace("ï", "i") \
                .replace("ó", "o").replace("ò", "o").replace("ô", "o").replace("ö", "o") \
                .replace("ú", "u").replace("ù", "u").replace("û", "u").replace("ü", "u") \
                .replace("Ç ", "c").replace("'", " ").replace("-", " ").replace("É", "e").replace("ç", "c").upper()

        # finally, scraping the postal code of each offer
        annonce['logcodepost'] = str(response.xpath('//div[@data-qa-id="adview_location_informations"]//text()')
                                     .extract()[2]).zfill(5)

        yield annonce

    ##############################################################################
```

References

Main R packages used in the framework of the Housing Dynamics project

R. Ysebaert, **'housing'** : This package created for the project imports reference layers for all the case studies of the ESPON Housing dynamics project (Paris, Avignon, Barcelona, Madrid, Palma de Majorque, Warsaw, Lodz, Krakow and Geneva). The mapping functions implemented allow to create an ESPON map with all the required styles (colors, labels, etc.)

Giraud T. Lambert N., **'cartography'** : This package is used to create and integrate maps in R workflows. This package helps to design cartographic representations such as proportional symbols, choropleth, typology, flows or discontinuities maps. It also offers several features that improve the graphic presentation of maps, for instance, map palettes, layout elements (scale, north arrow, title...).

Giraud T, Commenges H., **'SpatialPosition'** : Description: Computes spatial position models: Stewart potentials, Reilly catchment areas, Huff catchment areas.

Pebesma et al. '**sf'** : Support for simple features, a standardized way to encode spatial vector data. Binds to 'GDAL' for reading and writing data, to 'GEOS' for geometrical operations, and to 'PROJ' for projection conversions and datum transformations.

Wickham H., '**dplyr'**, A grammar for Data Manipulation. A fast, consistent tool for working with data frame like objects, both in memory and out of memory

ESPON 2020 – More information

ESPON EGTC
4 rue Erasme, L-1468 Luxembourg - Grand Duchy of Luxembourg
Phone: +352 20 600 280
Email: **info@espon.eu**
www.espon.eu, **Twitter**, **LinkedIn**, **YouTube**

The ESPON EGTC is the Single Beneficiary of the ESPON 2020 Cooperation Programme. The Single Operation within the programme is implemented by the ESPON EGTC and co-financed by the European Regional Development Fund, the EU Member States and the Partner States, Iceland, Liechtenstein, Norway and Switzerland.