UCLouvain

FACULTE DES SCIENCES

Faculté des sciences

Spatial and statistical modelling of air pollution in Belgium

Author : **Lucas BOUCHÉ**
Supervisor : **Christian HAFNER**
Reader : **Christian RITTER**
Academic year 2022-2023
Master [120] en science des données, orientation statistique

Preface

This dissertation has been prepared in partial fulfillment of the requirements for the master degree in Data Science (statistical orientation) delivered by the Université Catholique de Louvain.

First of all, I would like to express my deepest gratitude to Mr. Christian Hafner, my dissertation supervisor. I sincerely thank him for the freedom he gave me to accomplish this piece of work. His valuable advices helped me to carry out this exciting study.

I also would like to thank the agency IRCELINE who provided the data shown in this study. Their contribution to this project was essential and they went above and beyond to help me by affording me their precious time. Without them, this study couldn't have been conducted.

I would like to thank my family and especially my brother. I am grateful for the time they invested for the completion of this study.

Contents

List of Tables

List of Figures

Introduction

Climate change becomes more and more visible as the years pass. Forest fires, shifting seasons, rising temperatures and the thawing of permafrost are phenomena that are increasingly more common and occur more frequently nowadays. The consequences on global health of such phenomena are numerous and are usually underestimated. For instance, outside air pollution is responsible for the death of nearly 10 million of people per year which is considerable (Shindell, 2022).

My strong interest for environmental issues led me to define a subject around this topic. Moreover, I wanted to develop my analytic skills. Thus, I decided to contact organisms that collect environmental data. I have found an organisation called IRCELINE that registers air pollution data from monitoring stations located in Belgium. They accepted to share their data giving me access to a large dataset with data recorded over a long time period (2011-2020). Then, my objective was to analyze and extract relevant information from the collected data. After some time studying the subject, I have decided to model the data in space (spatial modelling) and then in time using statistical methods. A more complicated approach was to incorporate the spatial and temporal variability in one model (spatio-temporal modelling). Spatio-temporal modelling is thus considered as a major component of the dissertation.

The first chapter introduces what atmospheric pollution is and its different effects. A second chapter is dedicated to the presentation of the studied data. An exploratory analysis is also performed in order to get a better understanding of the data. The third chapter corresponds to the analysis and modelling of the data in space. In the fourth chapter, data are analyzed and modelled over time. Finally, the last chapter integrates the spatial and temporal variability in the data.

Chapter 1

Atmospheric pollution

1.1 Definition

Atmospheric pollution is the air contamination caused by a biological, chemical, or physical agent. This phenomenon results in the modification of the natural properties of the atmosphere. Pollution sources could be motor vehicles, forest fires, industries, agriculture, road traffic and so on. Emissions are usually emitted locally. They can also move from one country to another over long distances in the atmosphere.

Air pollution is a major issue both in healthcare and environment. On the one hand, it can cause lung diseases and in worst cases, death. Indoor air pollution is also responsible for causing serious health issues. According to Drew Shindell (2022), this type of pollution causes the death of around 4 million of people per year (mostly women and children). On the other hand, air pollution contributes significantly to climate change. According to the European Environment Agency (EEA), air pollution is the single largest environmental health risk in Europe. Concentration levels in the air can be very high and can exceed pollution levels recommended by health authorities.

Pollutant	PM10	PM2.5	NO2	O3
EU standards	11%	$< 1\%$	1%	12%
WHO guidelines	71%	96%	89%	95%

Table 1.1.1: Share of the european urban population exposed to concentrations levels that exceed European Union (EU) standards and WHO guidelines in 2020 (EEA, 2022)

Note that EU standards are not as strict as those from the World Health Organization (WHO).

The reduction of global emissions can prevent the death of millions of people. As an example, in the USA, a reduction of global emissions over the next 50 years (in accordance with the Paris Agreement) could save the life of nearly 4.5 million of people (Drew Shindel, 2022).

1.2 Particulate matters

Particulate matters (PM) denote microscopic solid or liquid particles suspended in the air. PM10 and PM2.5 are examples of particulate matters. These particles can stay many hours or months in the air.

Particles can be emitted in the atmosphere from natural sources. Natural sources could be forest fires, volcanic eruption, erosion and so on. Particles can be also emitted by sources related to human activity. Most of emissions from human activity come from transport, agriculture, industry and building heating. Particles can be divided in two categories depending on the way they are formed : primary particles and secondary particles. Primary particles are particles emitted directly into the atmosphere or formed by a mechanical fragmentation of a material (e.g metallurgy). Whereas secondary particles are formed in the atmosphere due to physical-chemical transformations of gaseous components such as NH_3 or NOx.

Epidemiological studies show that effects of air pollution on health are mostly caused by particulate matters rather than ozone. Moreover, according to the World Health Organization, there is no threshold from which no risk is incurred. Even a small exposure to particulate matters can worsen health issues such as asthma. In some cases, particulate matters can be also corrosive and can thus degrade some cultural monuments.

In contrast to greenhouse gases, particulate matters have a totally different impact on climate change : they reduce the global temperature (Samset, 2022). Particulate matters can form a fine cloud in the atmosphere. Hence, a part of the solar rays are reflecting and send back into space. Particulate matters have thus a cooling power on the earth's surface. Without the presence of PM, the earth's surface can warm faster. As a consequence, it can increase the intensity and frequency of heatwaves.

Nowadays, there are a lot of unanswered questions regarding the particulate matters : we still don't know specifically how they move, what chemical reactions they undergo in the atmosphere. Moreover, there is still a questioning on how they may impact the meteorological conditions. Many researchers are currently working on this subject.

1.3 Air pollution in Belgium

The university of Yales publishes indicators regarding the environmental performance of a country. 180 countries are studied with the most recent available data. An overall score is attributed to each country and is called the *Environmental performance index* (EPI). Different categories are also analyzed such as air quality, ecosystem vitality, biodiversity and habitat, agriculture, water resources and more. An EPI is also computed for each category. This index is expressed in percentage. The greater the value, the better the environmental conditions of the country. Without considering categories, Belgium has an overall EPI of 58.20% which is below its neighbouring countries.

The air quality indicator measures the direct consequences of air pollution on health

within the country. This indicator is based on the following components : PM2.5 exposure, household solid fuels, ozone exposure, nitrogen oxides exposure, sulfur dioxide exposure, carbon monoxide exposure, and volatile organic compound exposure. By looking at the air quality category, Belgium has a score of 74.60% which is again below the rank of its neighbouring countries. For context, France, Germany, Luxembourg and Netherlands have the respective scores : 82%, 75.20%, 81% and 76.80%.

The European Environment Agency has segmented the emissions in 7 categories or pollution sources :

- Transport
- Agriculture
- Waste
- Residential commercial and institutional
- Manufacturing and extractive industry
- Other

In Belgium and for the pollutant PM2.5 specifically, the source of pollution that emits the most is *Residential commercial and institutional*. *Transport* is the second largest emitting source of PM2.5. Although most of the emissions have decreased in Belgium, the impact of air pollution on health and economics is still sigfnificant.

Chapter 2

Data

This chapter presents the data used in the study. Details about the different pollutants and variables are given. Moreover, a first exploratory analysis is conducted.

2.1 Original data

Data have been collected in order to analyze and model the air pollution in Belgium. They have been provided by IRCELINE. It is a belgian interregional environment agency. This organism is a result of an agreement between the Wallonia, Brussels and Flemish region for providing air pollution data. IRCELINE keeps a common database for all the regions.

The statistical study is focused on 4 pollutants. Among the 4 pollutants, 2 are particulate matters : PM10 and PM2.5. The pollutants are defined below.

- PM10 : this pollutant refers to particles whose diameter is less than 10 microns (μm). It could be dust, pollen, mold or particles from tyre wear.
- PM2.5 : this pollutant corresponds to particles whose diameter is less than 2.5 microns. This could be combustion particles, organic compounds, metals, etc.
- O3 : O_3 is the chemical formula used for the ozone. There are two kinds of ozone, the one that is located between 15 and 45 km from the Earth's surface. It is commonly called the stratospheric ozone. This type of ozone is natural and protects us from UV rays coming from solar radiation. The other one is the tropospheric ozone which is also called the bad ozone because of its harmful effects on health. The study is focused on this one in particular. This type of ozone is located in the lower layer of the atmosphere (troposphere).
- NO2 : This pollutant is called nitrogen dioxide. It can cause health issues such as eye, nose or throat irritation and even lung irritation. In some cases, lung functions are damaged and the risk of asthma attacks increases.

The original dataset contains **27,353,664** observations. The proportion of missing values is **28%**. The database contains data for each hour. One line (observation) in the database corresponds to the concentration level registered for a pollutant in a station at a specific time period. As an example, a concentration marked at 00:00 corresponds to the average concentration registered between 11pm and midnight. Concentration levels are registered in different stations.

Some values in the original dataset are negative. This can be explained by a degree of uncertainty. For instance, the nitrogen dioxide is computed from two other pollutants : the nitrogen monoxide (NO) and the nitrogen oxide (NOx). The pollutant NO2 is computed as follows.

$$NO2 = NOx - NO \tag{2.1.1}$$

Negative values can be obtained if NOx concentrations are very low. According to IRCELINE, the limit of detection can be negative. From a physical point of view, a negative concentration is not realistic. Therefore, negative values are replaced by 0.

The distribution of the observations and missing values are shown below.

pollutant	PM10	PM2.5	O3	NO2
number of observations	7,890,480	7,539,792	4,032,912	7,890,480
% of missing values	27.5	35.6	17.6	26

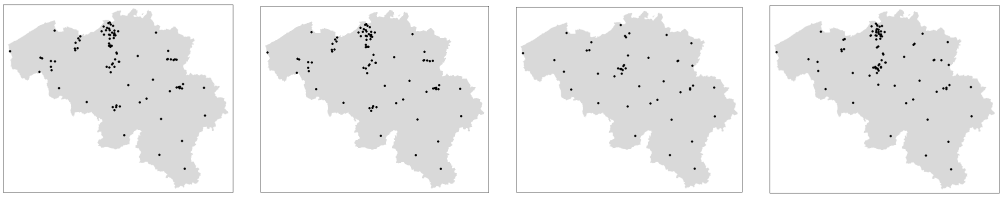
Table 2.1.1: Dataset summary

Stations do not register all pollutants. Some stations register only a specific pollutant. Therefore, stations considered for one pollutant can vary from one to another.

Region	PM10	PM2.5	O3	NO2
Brussels	6	5	8	14
Flanders	60	57	21	59
Wallonia	24	24	17	17
Total	90	86	46	90

Table 2.1.2: Number of stations for each pollutant and region

The Flemisch region has more stations than Wallonia and Brussels. Moreover, the pollutant O3 has almost half less stations than the other pollutants. The stations are represented on a map. Note that there are more stations in the South than in the North of the country.



(a) PM10 (b) PM2.5 (c) O3 (d) NO2

Figure 2.1.1: Localisation of the stations in the country

Sometimes for a specific time and a specific station, no observation is available. Maps shown above represent all stations for which there are observations without taking the time into account. For example, if one consider the year, the following table is obtained.

Pollutants	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
PM10	65	66	65	68	69	72	74	71	74	74
PM25	37	38	38	67	68	71	73	70	73	73
O3	41	41	42	41	41	41	38	39	41	42
NO2	68	71	73	71	74	76	73	72	74	76

Table 2.1.3: Number of stations for which there are observations

2.2 Additional variables

External data are collected in order to enrich the dataset. Indeed, the add of new variables can be interesting for modelling purposes. Hence, variables that might influence air pollution are added to the dataset. In this study, we will distinguish 3 types of variables :

- **Time invariant variables** : variables that vary only in space but not over time. The time does not influence the variable.

  LPS_closest : the closest distance from a station to a Large Point source (LPS). LPS can be strongly polluting industries. They can contribute heavily to the total share of the country's emissions.

  city_closest : the closest distance from a station to a big city. 8 major cities in Belgium are selected. This includes Antwerp, Ghent, Charleroi, Liege, Brussels, Bruges, Namur and Leuven. Distances are computed using the euclidean distance.

  airport_closest : the closest distance from a station to an airport. Airports must be considered as important pollution sources.

  region : the region where the station is located (Brussels, Flanders or Wallonia). One can expect more pollution in Brussels than in Wallonia for example.

vegetation : the proportion of green spaces in the city of the corresponding station : This includes, gardens and parcs, woods, orchards, farmland, pastures and meadows (source Statbel).

denspop : the population density of the city of the station of measurement (source Statbel).

x and y : coordinates of the stations.

- **Space invariant variables** : variables that vary only over time but not in space.

  week_time : categorical variable that takes the value weekend if the day corresponds to the weekend or workdays if the day corresponds to a workday (monday to friday).

- **Space-time variables** : variables that vary both in space and time. These data have been collected from a website (Weather And Climate) that provides historical meteorological data for more than 80,000 locations in the world. These variables are available only at a daily scale and for the period 2019-2020. They are used in the last chapter (spatio-temporal modelling).

  temperature : average temperature in degree Celsius.

  precipitation : total precipitation in millimeters (mm).

  humidity : proportion of humidity in the air (%).

  pressure : average pressure in the air (in inch of mercury).

  windpseed : average windpseed (kph).

2.3 Primary analysis

2.3.1 Univariate analysis

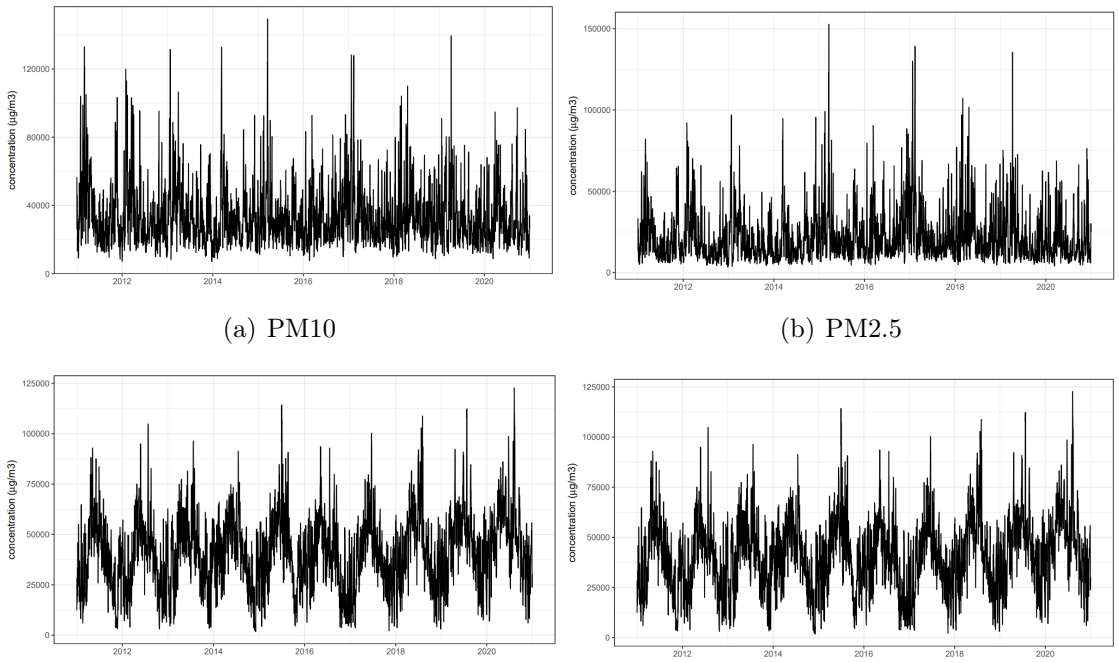
A first univariate analysis is performed. All concentration levels are expressed in micrograms per cubic meter ($\mu g/m^3$). A summary of the data can be shown below.

	min	Q1	median	mean	Q3	max	standard deviation
PM10	0	11	18	21.76	28	2,201	16.92
PM2.5	0	4	9	16.89	17.30	1,306	36.64
O3	0	24	44	45.12	62	269	28.68
NO2	0	10	20	31.85	36	3,763	63.03

Table 2.3.1: Statistical summary

Maximum values of the pollutants PM10, PM2.5 and NO2 can exceed a thousand of micrograms per cubic metre. The maximum value for O3 is much less (269). Recall that the original data are means. Therefore, maximum values can be even higher. NO2 has the

biggest variance. A first plot is made to get a better understanding of the data. The sum of the concentrations is calculated for each day.



(a) PM10 (b) PM2.5

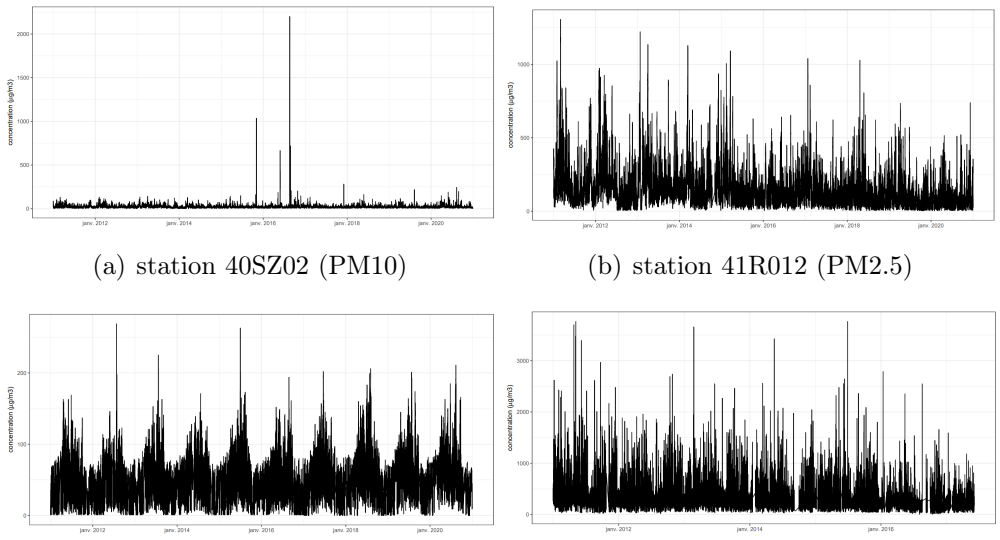
(c) O3 (d) NO2

Figure 2.3.1: Sum across stations for each day

PM10 and PM2.5 show irregular fluctuations over time. In contrast, O3 and NO2 seem to have a seasonal structure over time. These plots give a first indication of the distribution of the data for each pollutant. Further analysis are conducted in the following sections.

2.3.2 Extreme values

Analysing extreme values is essential when dealing with air pollution data. Indeed, knowing when peaks of concentration are most likely to occur can be very useful. This can help public organisations to prevent peaks of pollution in different cities. For each pollutant, the station that registers the maximum concentration is analyzed over a large period of time. The result is shown below.



(a) station 40SZ02 (PM10)

(b) station 41R012 (PM2.5)

(c) station 42N054 (O3)

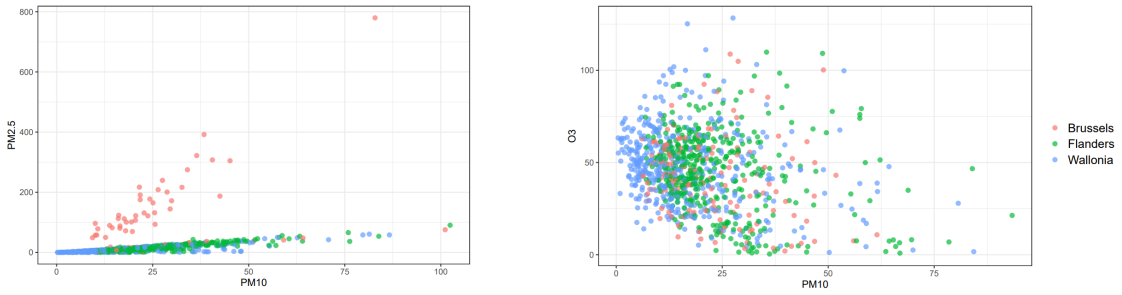
(d) station 41LEC1 (NO2)

Figure 2.3.2: Stations with maximum values

The station that registers the highest concentration for PM10 is 40SZ02. This station is located in Steenokkerzeel. A part of the Brussels airport is located in this city. The presence of this airport can explain this extreme value. For PM2.5, the station is located in Uccle. For O3, it is the station 42N054 located in the city of Landen. For NO2, the station is located near the tunnel Annie Cordy in Molenbeek-Saint-Jean. Except for O3, stations that register the maximum value are located in high population density areas.

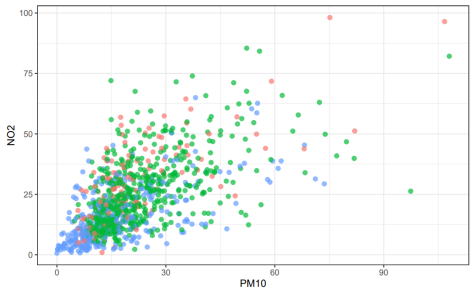
2.3.3 Bivariate analysis

A bivariate analysis is carried out. The data are aggregated per day. The correlation between pollutants is analyzed in each region. A sample of size 1,000 is taken with common dates and stations between pollutants. The result is shown below.

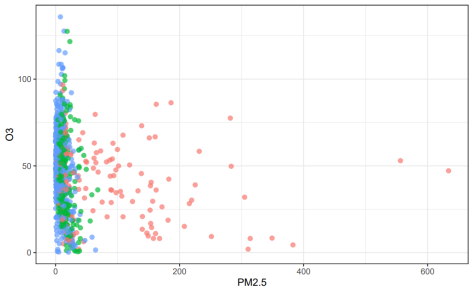


(a) PM2.5 against PM10

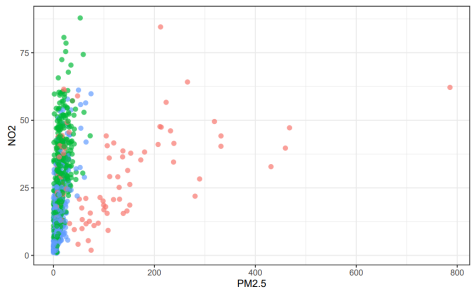
(b) O3 against PM10



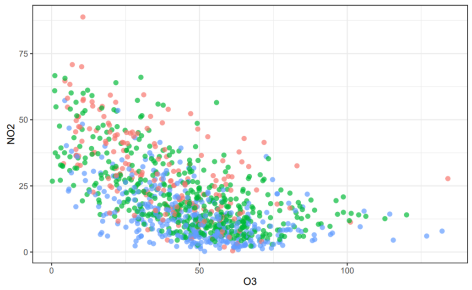
(c) NO2 against PM10



(d) O3 against PM2.5



(e) NO2 against PM2.5



(f) NO2 against O3

PM10 and PM2.5 are correlated. The scatter plot (PM2.5 against PM10) shows that the pollutant PM2.5 can reach concentrations that are far above those of PM10, especially in the region of Brussels. This is not surprising, as the region of Brussels is a zone with a high road traffic. The pollutant PM10 seems not correlated with the pollutant O3. However, PM10 is also correlated with NO2. PM2.5 is not correlated with O3 and NO2. But NO2 shows a correlation with the pollutant O3 : as O3 concentrations increase, the NO2 concentrations decrease.

Chapter 3

Spatial Analysis

The data of the study can be analyzed in space as each concentration is registered at a specific location. This chapter will cover the analysis and modelling of the data in space.

3.1 Projection system

2-dimensional representation of geographical data requires a projection system. Projection systems produce necessarily a distortion in the visual representation. However, they are useful for creating maps of specific geographic zones on earth. There exists many projection systems. The one usually used for Belgium is *Belgian Lamber 72 (EPSG 31370)*. This projection system is used for the study. The unit associated to this projection system is meter.

3.2 Spatial autocorrelation

Spatial autocorrelation is a phenomenon that occurs when observed values show a dependence in space. In other words, there is a spatial structure in the data. In statistical modelling, spatial autocorrelation can be the result of a missspecification of the model. There exists two kinds of spatial autocorrelation :

- Positive : values that are close to each other tend to be similar.
- Negative : values that are close to each other tend to have different values.

Spatial autocorrelation can be detected visually by representing a map of the data. Another more rigorous way is to apply the statistical test of Moran. The I of Moran is expressed as follows.

$$I = \frac{N}{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}} \times \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}(y_i - \overline{y})(y_j - \overline{y})}{\sum_{i=1}^{N}(y_i - \overline{y})^2} \quad i \neq j \tag{3.2.1}$$

w : spatial weight matrix of size N × N.

The spatial weight matrix quantifies the closeness between two locations i and j. Different methods are possible to quantify this closeness. In the context of the study, the inverse of the distance between locations is calculated. This method attributes more weight to closed locations.

The null hypothesis (H0) and alternative hypothesis (H1) are cited below.

H0 : absence of spatial autocorrelation against H1 : presence of spatial autocorrelation

The I of Moran is approximately normally distributed if N is sufficiently large. Under the null hypothesis H0,

$$E(I) = \frac{-1}{N-1} \qquad Z = \frac{I - E(I)}{\sqrt{V(I)}} \sim N(0,1) \tag{3.2.2}$$

Hence, a Z test can be performed. Z is used as a test statistic. The null hypothesis is rejected if $Z \geq z_{\alpha/2}$.

If the autocorrelation is significant, results from the Moran test indicate also whether the spatial autocorrelation is positive or negative. If the I of Moran is greater than the expected value, the autocorrelation is positive. In contrast, if the I of Moran is lower than the expected value, the autocorrelation is negative.

3.3 Kriging

A very powerful geostatistics method that enables to predict values in space is kriging. It is very used in the environmental field. By using this method, we can predict values at unknown locations from known values (sample).

3.3.1 Definition and notation

The geographical region in which the study is performed is called the domain. In the context of the study, the domain is noted S. Thus, the variable of interest will be a realisation of a random variable called $Z(s)$ where $s \in S$. The values $Z(s_i)$ (i = 1,..., n) are the sample values respectively located in (s_1,..., s_n). The kriging method usually makes a strong assumption regarding the variability of the data. It is called the hypothesis of isotropy. This hypothesis is commonly used in geostatistics and geosciences. Isotropy means uniform in all directions. More details are given in the next section. The global formula of the kriging estimator is a linear combination of the observed values.

$$\hat{Z}(s_0) = \sum_{i=1}^{n} \lambda_i Z(s_i) \tag{3.3.1}$$

$\hat{Z}(s_0)$: predicted value located at the location s_0.
n : size of the sample.
λ_i : weight attributed to location s_i.
$Z(s_i)$: observed data at the location s_i.

The kriging estimator has the following properties :

- It is an exact interpolator : That is, the value estimated at a known point (observed from the sample) is the value observed. This property is most of the time false in practice. Indeed, when estimating on a specific grid, the probability that the center of a pixel in the grid falls exactly on a sample location is very low.

- It is the best linear unbiased estimator (BLUE). It is the one that minimizes at most : $E[(Z(s_0) - \hat{Z}(s_0))^2]$. Amongst all unbiased estimators, it is the one that has the minimum variance.

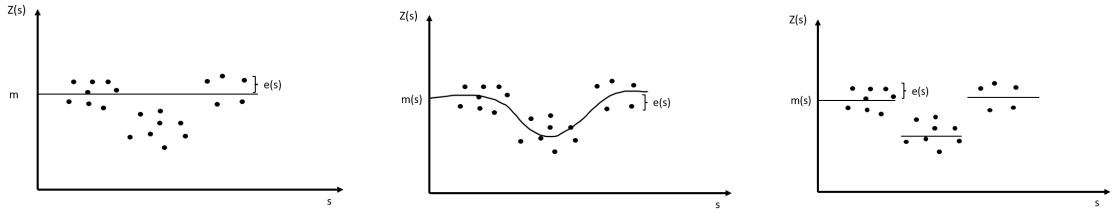
There are 2 kinds of kriging predictions : point kriging or block kriging. Point kriging is used to predict values at a specific location (a point). Usually, the prediction corresponds to the center of a pixel. Whereas block kriging refers to predictions on a surface rather than a point. Block kriging is used when the sample is low or the uncertainty is high. The advantage of this form of prediction is that it lowers the prediction variance because of the larger area taken into account.

3.3.2 Types of kriging

The general model of kriging is composed of a deterministic and stochastic part. The model takes the following form :

$$Z(s) = m + e(s) \quad s \in S \tag{3.3.2}$$

Where m is the deterministic part and $e(s)$, the stochastic part. Different assumptions can be made regarding the deterministic part. These assumptions are related to the type of kriging : simple, universal or ordinary. The simple kriging assumes that the mean is known. And that the trend is constant for the whole spatial domain. As opposed to the simple kriging, the universal kriging assumes that the trend is not constant over the spatial domain. Finally, the ordinary kriging makes the hypothesis of a locally unknown constant trend m. m is estimated locally and determined from the neighbouring values. This type of kriging has the advantage to take into account the local variations of the variable of interest. Illustrations are given below with the assumption that the spatial domain is defined over 1 dimension.



(a) Simple kriging (b) Universal kriging (c) Ordinary kriging

In this chapter, we will focus only on the ordinary kriging. Recall that the kriging predictor is unbiased and is optimal (minimum variance). The unbiasedness invlolves that :

$$\begin{aligned} E[\hat{Z}(s_0)] &= m \\ &= \sum_{i=1}^{n} \lambda_i E(Z_{s_i}) \\ &= \sum_{i=1}^{n} \lambda_i m \quad \rightarrow \sum_{i=1}^{n} \lambda_i = 1 \end{aligned} \tag{3.3.3}$$

The kriging formulas are :

$$\begin{aligned} \sum_{j=1}^{n} \lambda_j K_{ij} + m &= K_{i0} \\ \sum_{i}^{n} \lambda_i &= 1 \end{aligned} \tag{3.3.4}$$

Where $K_{ij} = \gamma(s_i - s_j)$. γ is the variogram function (see following section for more details). And $K_{i0} = \gamma(s_0 - s_i)$. $(s_i - s_j)$ is the distance between location s_i and location s_j. s_0 is the prediction location.

Matricially, the system can be written with the following components : K (covariance matrix), λ (the weight matrix), K_0 the covariances with the known measures (0 represents the target prediction location). Z represents the observed values.

$$\begin{cases} K\lambda + m = K_0 \\ \hat{Z}_0 = \lambda^T Z \end{cases} \tag{3.3.5}$$

$$K = \begin{pmatrix} K_{1,1} & ... & K_{1,n} & 1 \\ ... & ... & ... & 1 \\ K_{n,1} & ... & K_{n,n} & 1 \\ 1 & ... & 1 & 0 \end{pmatrix} \lambda = \begin{pmatrix} \lambda_1 \\ ... \\ \lambda_n \\ m \end{pmatrix} K_0 = \begin{pmatrix} K_{1,0} \\ ... \\ Kn, 0 \\ 1 \end{pmatrix} Z = \begin{pmatrix} Z_1 \\ ... \\ Z_n \\ 0 \end{pmatrix} \tag{3.3.6}$$

3.3.3 Variogram

The kriging method requires the computation of the variogram. It corresponds to the variance of all pairs from the sample by taking into account the distance between two points in the space. Usually, it is the semivariogram that is computed using a factor 1/2 (see formula 3.3.7). The semivariogram depends only on the distance. The direction in the space (e.g East, West, North...) is not taken into account (isotropy assumption). It is usually a bounded increasing function. Other specific forms of semivariograms exist such as linear or pure nugget effect (absence of spatial autocorrelation). When the variogram is not bounded, the mean and variance are not defined (Loonis et al., 2018). This phenomenon usually indicates a trend at a bigger scale. The formula of the semivariogram is shown below.

$$\gamma(h) = \frac{1}{2} \times E[(Z(s) - Z(s+h))^2] \tag{3.3.7}$$

In theory, when the distance h is equal to 0 (no variation), the semivariogram is equal to 0. But in practice it is usually a constant c. This can be justified by simply an error of measurement or a high variability at a low scale. This phenomenon is commonly called the nugget effect.

An illustration of a variogram model is given below. Note that it is an example, various models are possible depending on the shape of the empirical data.

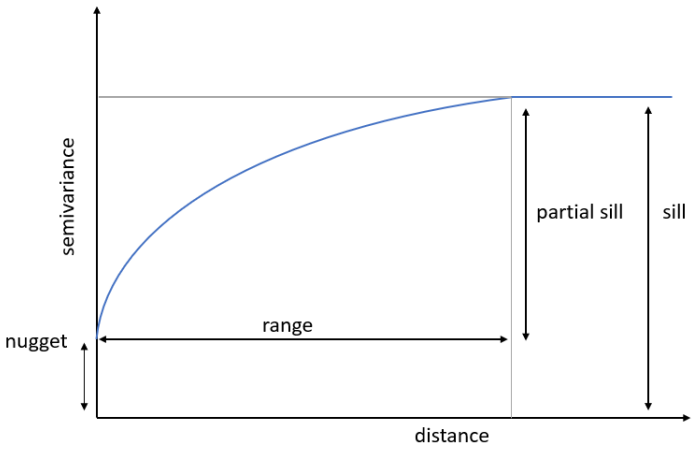


Figure 3.3.1: Illustration of a variogram model

The nugget, range and partial sills are model parameters. The nugget is the semivariance at a distance of 0. The range is the distance from which there is an absence of spatial dependence between observed values. The sill is a threshold from which the semivariogram becomes constant.

In geostatistics, different assumptions can be made regarding the stationarity of the process. 3 forms of stationarity exist.

- **strict stationarity** : All characteristics from the random function remain the same. That is, the joint distribution of the $Z(s_i)$ is the same as that of $Z(s_{i+h})$. This form of stationarity is usually not applicable for most real-world cases (very restrictive).

- **second-order stationarity** or weak stationarity : The random variable has a constant mean, constant variance over the whole spatial domain. And the covariance only depends on the distance lag h.

$$Cov[Z(s+h), Z(s)] = E[(Z(s+h) - m)(Z(s) - m)] = C(h) \quad \forall\ s, s+h \in S \tag{3.3.8}$$

- **intrinsic stationarity** : The hypothesis is that $E[(Z((s+h) - Z(s))^2] = 0$. And that the variance of the random variable $Z(s+h) - Z(s)$ does not depend on the location s but only on the distance lag h :

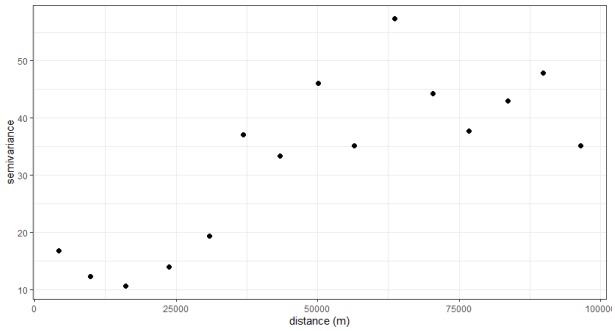
$$V[(Z(s+h) - Z(s))] = 2\gamma(h) \quad \forall\ s, s+h \in S \tag{3.3.9}$$

  All second-order stationary processes are intrinsically stationary but the opposite is not true (Loonis et al., 2018). The second-order stationarity is more restrictive than the intrinic stationarity. In contrast to second-order stationarity, intrinsic stationarity does not make any assumption regarding the moments of $Z(s)$.

3.4 Application

3.4.1 Variogram modelling

An ordinary kriging is performed for a specific time (1st January, 2020) and a specific pollutant (PM10). The empirical semivariogram is represented below.



(a) Empirical semivariogram

Before the modelling procedure, It would be rigorous to test if this estimated semivariogram is due to randomness or not. A bootstrap is performed : the observations are randomly allocated to the stations of measurements. This step is repeated 1,000 times. The result is shown below.

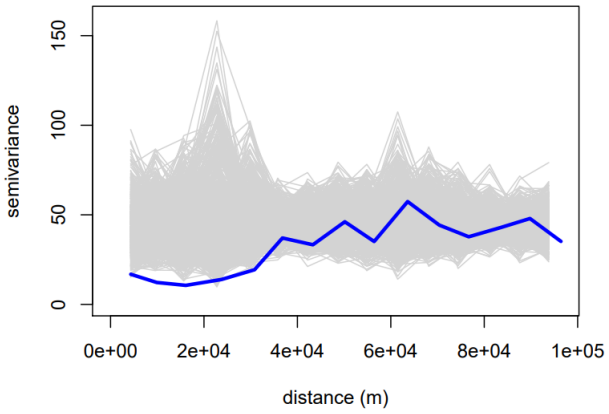
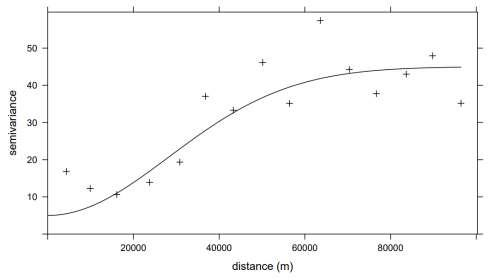


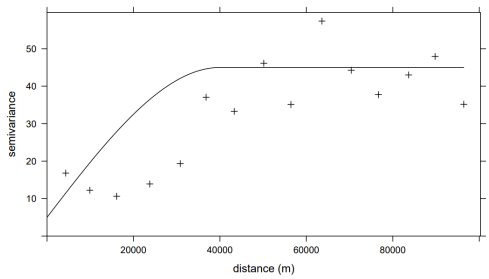
Figure 3.4.1: Randomly generated semivariograms (grey) and empirical semivariogram (blue)

The estimated semivariogram (blue) seems different from the random generated semivariograms (grey). Although most of the estimation falls into the grey part, the blue line shows a slight increase. Visually, the randomness hypothesis can be rejected.

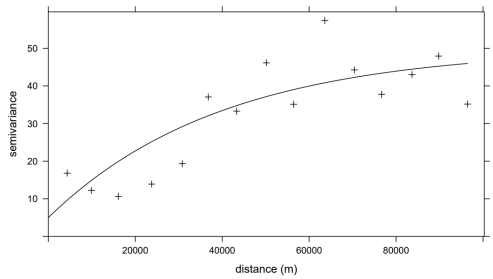
The next step is to find an appropriate model that fits the estimates of the semivariogram. Different parametric models can be used to model an empircial semivariogram. The choice of a parametric model ensures that the prediction variances are positive. 4 models are compared : Gaussian, Spherical, Pentaspherical and Exponential. Parameters of each model are manually specified.



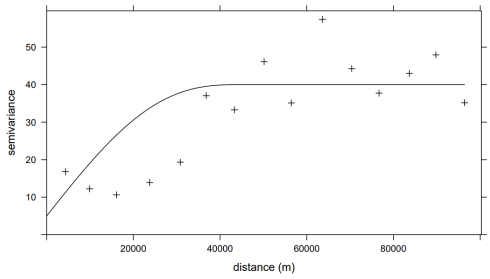
(a) Gaussian model



(b) Spherical model



(c) Exponential model



(d) Pentaspherical model

Visually it is difficult to say which one is better. Therefore, a K FOLD cross validation is performed for each model. The model with the lowest cross validation error will be retained for kriging. This method works as follows : the dataset is divided into k parts. In this K FOLD cross validation, k = 4 and n = 71. 75% of the observations will be used for training and the remaining observations for prediction. The process is repeated k times so that for each observation, a prediction is obtained. At the end of the procedure, a residual (prediction error) for each prediction can be obtained.

	Gaussian	Spherical	Pentaspherical	Exponential
Cross validation RMSE	5.1	6.12	6.02	5.3

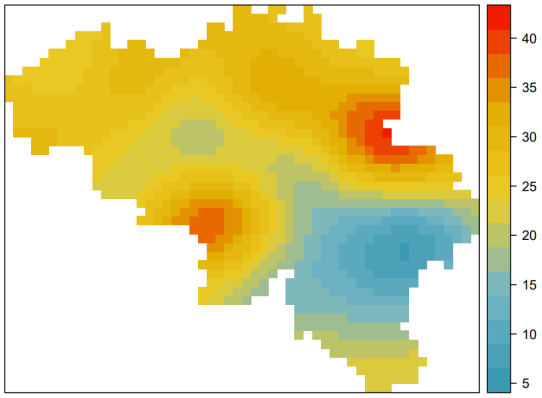
Table 3.4.1: Root mean squared errors from K FOLD cross validation

The chosen model is the Gaussian model and can be expressed as follows.

$$\gamma(h) = C_0 + C_e(1 - e^{-\frac{h^2}{a^2}}) = 5 + 40(1 - e^{-\frac{h^2}{40000^2}}) \quad h \geq 0 \tag{3.4.1}$$

C_0 is the nugget. C_e is the partial sill. a is the distance from which 75% of the sill is reached. $a\sqrt{3}$ is the range or effective range (see figure 3.3.1).

Choosing a model with respect to a statistical criterion does not guaranty that the best model has been selected (Waller et al., 2014). Sometimes, the model seems unrealistic. Finally, the choice of a model should match both with the visual pattern of the empirical data and the understanding of the physical process.
The ordinary kriging is performed. The kriging predictions are shown below.



(e) Prediction

3.4.1.1 Validation

If the model is good, residuals from cross validation should be small, distributed around 0 with no spatial structure.

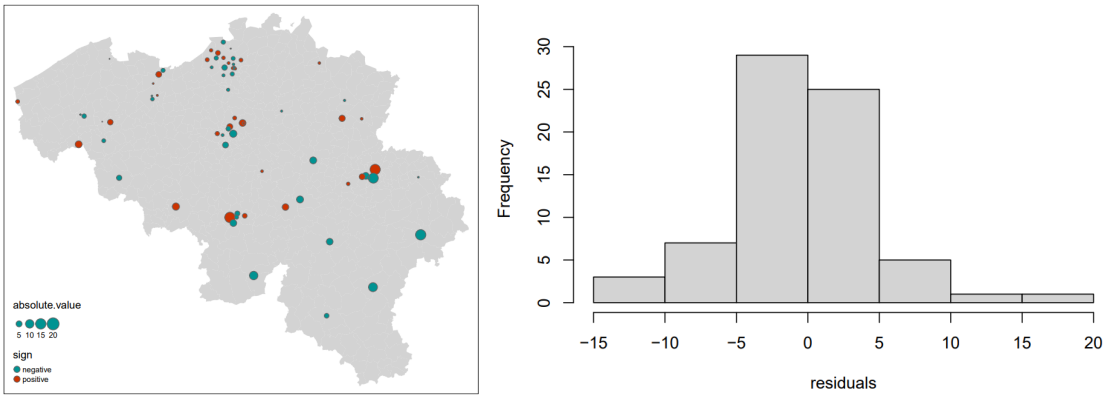


Figure 3.4.2: Residuals from K FOLD cross validation

The size of the circles denotes the magnitude of the residuals. The color indicates the sign of the residuals (positive or negative). Residuals in the South-East of the country are negative. This can indicate a potential spatial dependence. In addition, the low presence of stations in the South-East of the country can also explain such high residuals. The spatial autocorrelation is tested using the test of Moran.

	I of Moran	Expected	pvalue	H0
Moran test	-0.053	-0.014	0.32	accepted

Table 3.4.2: Results from the Moran test

The null hypothesis is not rejected. The model is valid. However, there are very high residuals. This indicates that the model can be improved. Various solutions exist. Adding covariates into the model is one of them. Cokriging is also a good solution. It is a form of kriging where other variables are used to improve the prediction. This method is not treated here but can be very efficient. Especially when the co-variables are highly correlated with the variable of interest.

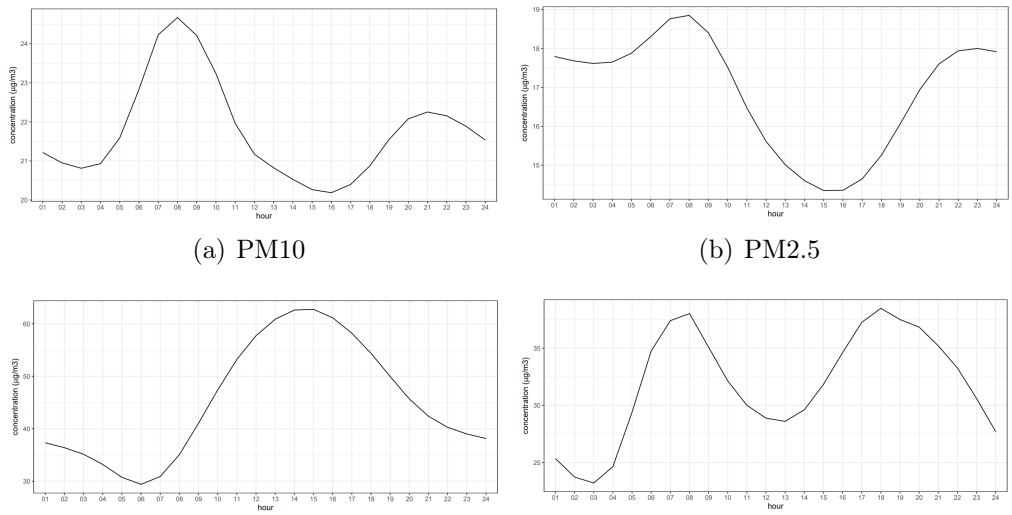
Chapter 4

Temporal analysis

In this chapter, the variable of concern is the time. The aim is to detect if there are interesting patterns over time. The analysis is carried out at different time scales.

4.1 Exploratory analysis

4.1.1 Hourly analysis



(a) PM10 (b) PM2.5

(c) O3 (d) NO2

Figure 4.1.1: Mean per hour for each pollutant

PM10 and PM2.5 concentrations seem to vary accordingly to the daily work schedule : they increase the early morning and fall suddenly in the middle-end of the morning. There is a rise again when people are leaving their workplace (5pm). For the case of NO2, a similar behavior is observed. But the concentrations decrease until noon and rise rapidly

in the beginning of the afternoon. Concentrations start to decrease around 18h. O3 concentrations seem to vary accordingly to the sunlight or the temperature. A peak is reached at 3pm. Then concentrations decrease slightly until the early morning.

4.1.2 Weekly analysis

The concentrations over the week are now investigated. The concentrations of the 4 pollutants are analyzed during the workdays (Monday to Friday) and the weekend. The logarithm is applied to the data to obtain a distribution that is closer to the Normal distribution.

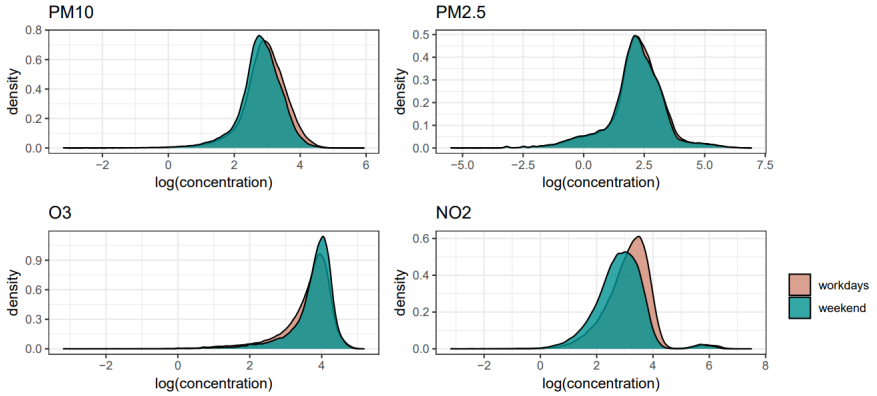
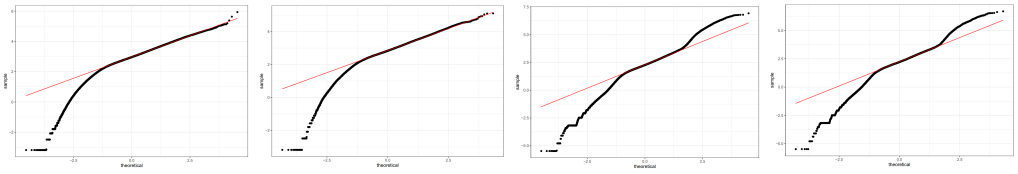
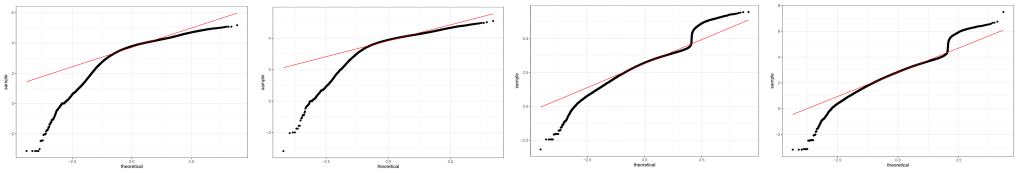


Figure 4.1.2: Smoothed densities

A test is performed for each pollutant to check whether the distributions are significantly different. Normality of each sample is checked using the Q-Q plots.



(a) PM10-workdays (b) PM10-weekend (c) PM2.5-workdays (d) PM2.5-weekend



(e) O3-workdays (f) O3-weekend (g) NO2-workdays (h) NO2-weekend

Figure 4.1.3: Normal Q-Q plots

Q-Q plots show enough evidence that each sample is not normally distributed. Consequently, a non-parametric test is conducted. The Wilcoxon Rank-Sum test is used and requires no assumptions. The test works as follows. Sample 1 and sample 2 are combined as a whole sample set with size n1 + n2. Then each observed value from the new sample is ranked in the increasing order. Under the null hypothesis (distributions are equal), sample 1 and sample 2 are expected to be a random realisation of the n1 + n2 ranks. The sum of the rank of the sample 1 is used as a test statistic. And if the sum of the rank is higher/lower than expected (more high/low values than expected), the null hypothesis is rejected.

A one-sided test is performed to see if the concentrations are greater during the workdays than during the weekend. For O3, the concentrations during the workdays seem smaller than during the weekend. Therefore, we test if the concentrations are smaller during the workdays for this pollutant.

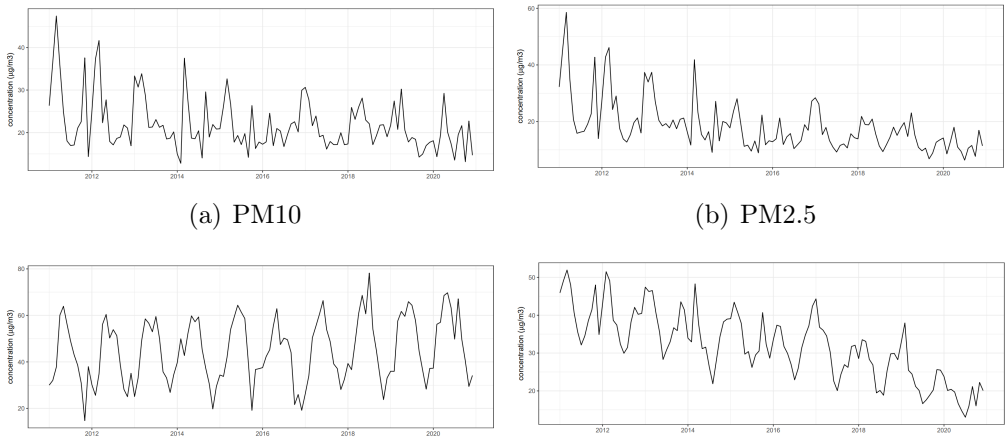
H0 : true location shift is equal to 0 or distributions are equal.
H1 : true location shift is greater than 0 (PM10, PM2.5 and NO2) / true location shift is less than 0 (O3).

	PM10	PM2.5	O3	NO2
pvalue	$<$ 2.2e-16	$<$ 2.2e-16	$<$ 2.2e-16	$<$ 2.2e-16

Table 4.1.1: Results of the Wilcoxon Rank-Sum test

P-values are less than 0.05. Therefore, the null hypothesis can be rejected for each pollutant. There is thus an evidence that the concentrations are greater during the workdays for PM10, PM2.5 and NO2. And that the concentrations are lower during the workdays for O3.

4.1.3 Month and year analysis



(a) PM10 (b) PM2.5

(c) O3 (d) NO2

Figure 4.1.4: Average concentrations per month from 2011 to 2020

The average of the concentrations (PM10 and PM2.5) are both decreasing over time. Moreover, irregular fluctuations are observed. In contrast, O3 average concentrations are slightly increasing over time. Moreover, a clear seasonal structure is present. The overall trend for NO2 is downward. As for O3, a clear seasonal structure can be observed.

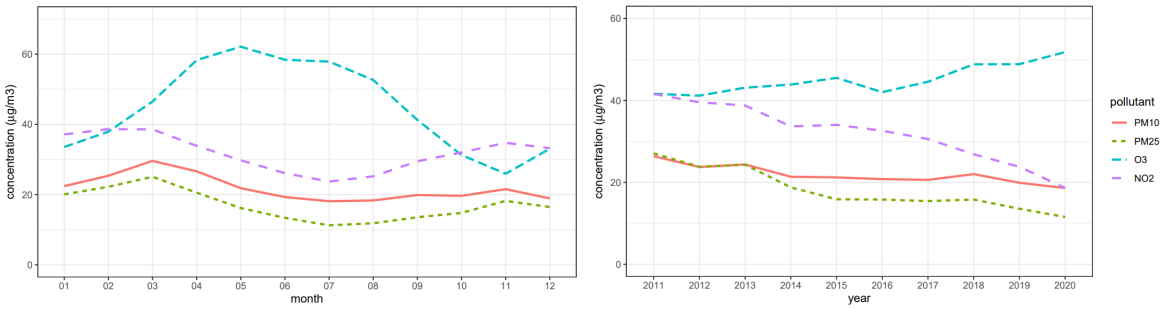


Figure 4.1.5: Average concentration per month and per year

O3 concentrations increase from the beginning of the year. Then, they reach a peak around may and decrease until the end of the year. However, the seasonal structure is different for the other pollutants. PM10, PM2.5 and NO2 have a similar seasonal structure. Concentrations increase slightly before reaching a peak in mars. Then the concentrations of the 3 pollutants decrease until july. From july, concentrations start to increase again until november.

4.2 COVID 19 pandemic

The virus COVID 19 had a major impact on our daily life. The pandemic has drastically contracted the global economy and caused the death of millions of people. Besides the negative effects, air quality during lockdown has considerably improved, especially in urban cities. A binary variable called *covid* is created. This variables indicates whether the observed value is registered during the covid period or not. The values taken by the variable are *yes* or *no*. The start of the period is fixed to the 1st march of 2020. And the end is the most recent day from the data set : 31st of December, 2020.

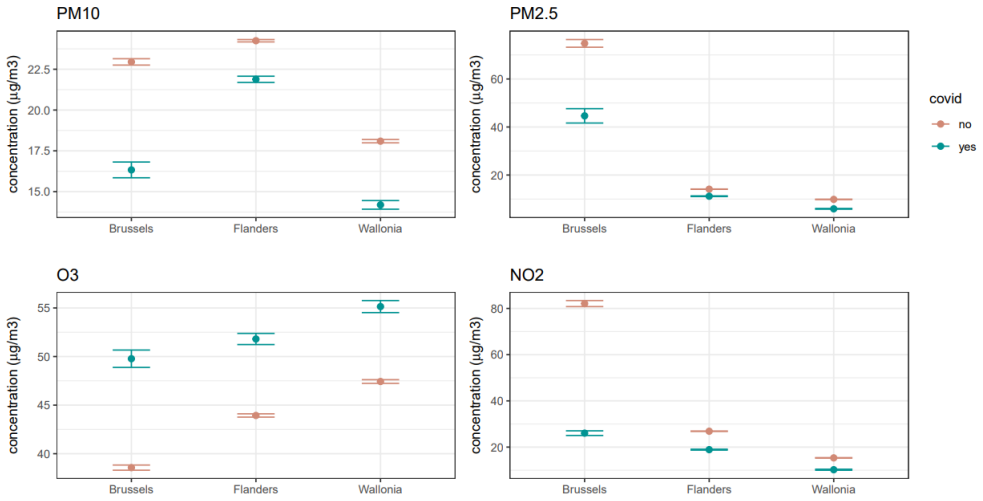


Figure 4.2.1: Average concentrations per pollutant and region with 95% confidence interval

The difference of the mean between the pandemic period and non pandemic period is greater in the region of Brussels (PM10, PM2.5 and NO2). The pollutant O3 shows a totally different behavior : concentrations are higher in Wallonia. And the difference between the periods is roughly equal regardless of the region (O3).

4.3 O3 concentrations prediction

The goal of this section is to predict new values for the next 12 months based on the monthly mean observations of the pollutant O3 from 2011 to 2020.

4.3.1 Primary analysis

At first, the time series is visually inspected.

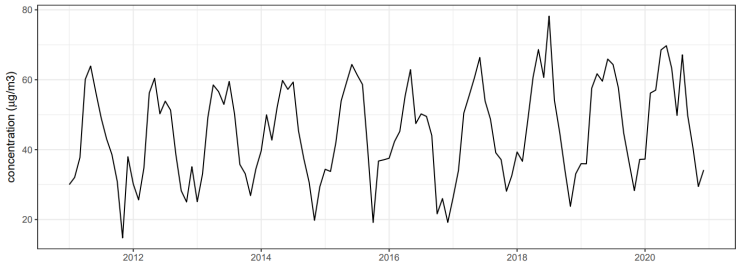


Figure 4.3.1: O3 concentrations mean

- **Trend :** In the period 2011-2016, the trend seems constant. From 2017 to 2020, the trend seems constant again but at a slightly higher level than the previous mentioned period. This highlights a level structural break : a sudden change in the mean.

- **Seasonality :** Each year, there are peaks of concentration at the same period of the year. The time series of each year are represented below.

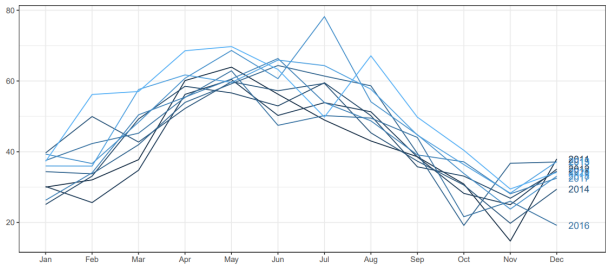


Figure 4.3.2: Seasonal plot

The seasonal effect remains the same regardless of the year. The highest concentrations are reached during the hottest months.

- **Outliers and Extreme values :** The time series does not show extreme values.

	mean	median	standard deviation	skewness	interquartile range
O3 mean concentrations	44.86	44.33	13.58	0.04	22.36

Table 4.3.1: Statistical summary

A transformation of the data is not required in this case. Indeed, the standard deviation is relatively low. And the amplitude of the fluctuations does not increase over time.

4.3.2 Modelling

The time series is not stationary. Because of the presence of the structural break and the seasonal effect, the mean changes over time. Therefore, the time series needs to be differentiated. The differentiation is a method that enables to obtain stationarity. The main idea is to remove the different components of the time series (trend, seasonality...).

A series is stationary when its statistical properties do not change over time. In other words, the statistical properties are independent of time. There exists different form of stationarity. The one studied is the following.

$$E(X_t) = \mu \quad V(X_t) = \sigma^2 \quad Cov(X_t, X_{t+h}) = r_h \tag{4.3.1}$$

Where r_h depends only on h but not on t.

At first, a seasonal differentiation is applied to the data (lag 12). Mathematically, the differentiation can be written as follows :

$$Y_t = \nabla_{12} X_t = (1 - B^{12}) X_t \tag{4.3.2}$$

Where X_t represents the original time series and Y_t the differentiated data. B is a back shift operator (i.e $B^d(X_t) = X_{t-d}$). And ∇_d is a differentiated operator (i.e $\nabla_d = X_t - X_{t-d}$). The differentiated data are represented below.

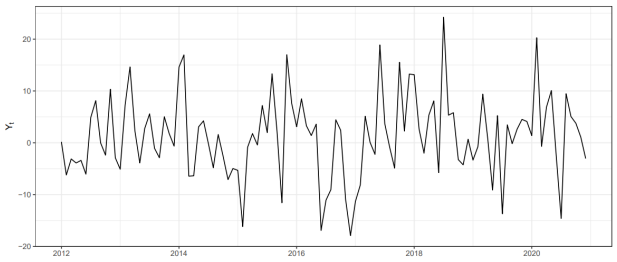


Figure 4.3.3: Differentiated data

A visual representation of the data is not sufficient to confirm stationarity. Hence, the Dickey-Fuller test is performed. It is a unit root test. That is, the test checks whether there is a unit root in the process. A unit root process is also called a *random walk*. A random walk is a specific process that is unpredictable. Random walks are non stationary processes : their mean and variance change over time.

$$\nabla_1 Y_t = c + \beta t + \gamma Y_{t-1} + e_t \tag{4.3.3}$$

Where $\gamma = (\alpha - 1)$.
The model has a constant c and linear trend βt. Yt represents the seasonal differentiated data. 3 tests can be achieved from this model. They are described in the table below.

Hypothesis	H0	H1
Test 1 ($c = 0$ and $\beta = 0$)	random walk or $\gamma = 0$	stationary or $\gamma < 0$
Test 2 ($c \neq 0$ and $\beta = 0$)	random walk with a constant or $\gamma = 0$	level stationary or $\gamma < 0$
Test 3 ($c \neq 0$ and $\beta \neq 0$)	random walk with a trend and a constant or $\gamma = 0$	trend stationary or $\gamma < 0$

Table 4.3.2: Hypotheses of the Dickey-Fuller test

The test 1 is used because no trend or constant seems to be present in the time series. The corresponding p-value of the test is 0.01. Thus, there is no evidence that the process is a random walk. The series can be considered as stationary.

The ACF (autocorrelation function) and PACF (partial autocorrelation function) are now analyzed.

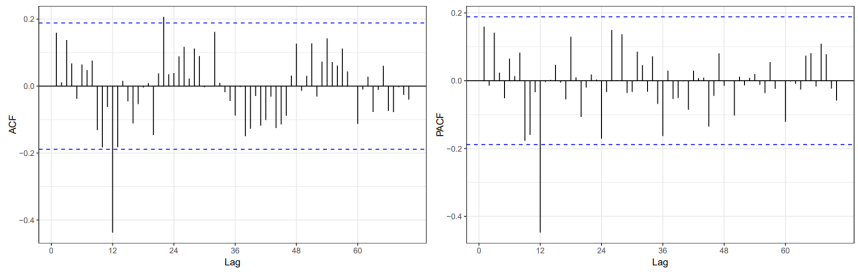


Figure 4.3.4: ACF and PACF of the seasonal differentiated data

A highly significant spike at lag 12 is present in the ACF. This means that values at time t and values at time t + 12 are correlated. From the PACF, there is an exponential decay in the seasonal lags (12, 24 and 36). A Seasonal Autoregressive Integrated Moving Average (SARIMA) model is considered for modelling since the seasonal effect represents a major component in the time series. A SARIMA model $(p, d, q) \times (P, D, Q)_s$ with period s is written as follows.

$$(1 - \phi B)^p (1 - \Phi B^s)^P (1 - B)^d (1 - B^s)^D X_t = (1 + \theta B)^q (1 + \Theta B^s) e_t \tag{4.3.4}$$

ϕ and θ are respectively auto-regressive (AR) and moving average (MA) coefficients. Whereas Φ and Θ are respectively seasonal AR and MA coefficients. Parameters d and D are already chosen (d = 0 and D = 1). The PACF and ACF suggest that either a seasonal AR or MA term could be present in the model. Hence, 2 models are considered : a SARIMA $(0, 0, 0) \times (1, 1, 0)_{12}$ (model 1) and a SARIMA $(0, 0, 0) \times (0, 1, 1)_{12}$ (model 2) .

	coefficient	Estimate	Std. Error	z value	Pr(>|z|)	AIC
Model 1	sar1	-0.44	0.09	-5.05	4.349e-07 ***	738.04
Model 2	sma1	-0.59	0.09	-6.35	2.206e-10 ***	731.36

Table 4.3.3: Models estimates

Model 1 and model 2 have both significant coefficients. However, model 2 has a lower AIC. Model 2 is thus retained. Model 2 can be written as follows :

$$(1 - B^{12})X_t = (1 - 0.59B^{12})e_t \tag{4.3.5}$$

The residuals are analyzed to check whether they are independent and identically distributed (iid).

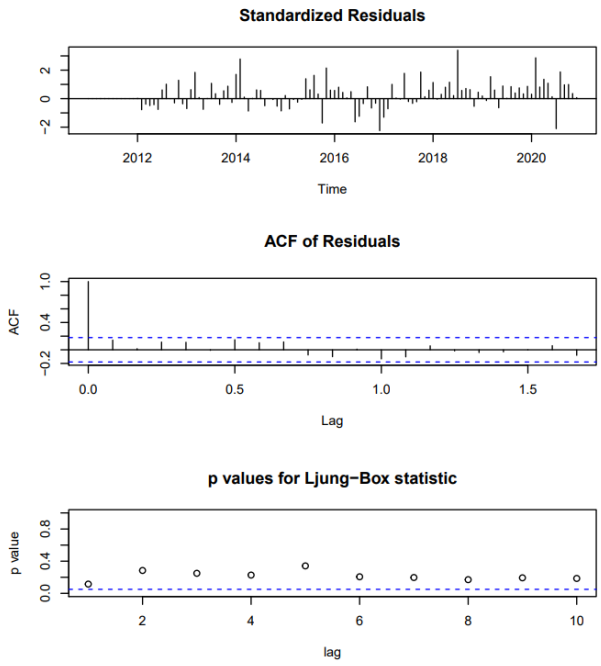


Figure 4.3.5: Residuals diagnostic

In the ACF, no spikes are outside of the interval which is a good point. Then the LjungBox statistic is computed for lag 1 until lag 10. Instead of testing the autocorrelation at a specific lag, the test checks if there is a presence of serial autocorrelation until a fixed lag k.

$$Q(k) = n(n+2)\sum_{i=1}^{k}\frac{\hat{\rho}_i^2}{n-i} \tag{4.3.6}$$

n is the sample size. $\hat{\rho}_k$ is the empirical autocorrelation coefficient. The hypotheses of the test are :

H0 : $\rho_1 = \rho_2 = ... = \rho_k = 0$ against H1 : at least one ρ_i is different from 0.

The test will reject the null hypothesis for high values of $Q(k)$. All pvalues are greater than 0.05. The null hypothesis is not rejected for each lag k. We can therefore assume an absence of serial correlation in the residuals of the model. Finally, predictions with the prediction interval are shown below.

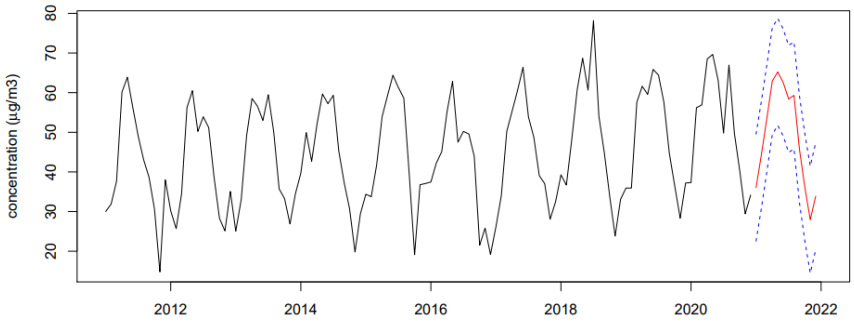


Figure 4.3.6: 12 step ahead prediction with prediction interval

Chapter 5

Spatio-temporal modelling

This chapter covers the modelling of the data with respect to time and space. Time and space are variability sources that can influence considerably the concentration levels. For instance, the pollution can be stronger in urban areas than in rural areas (spatial dependence). And concentrations can vary depending on the period of the year (temporal dependence). In this chapter, each pollutant is modelled at various times and locations.

5.1 Spatio-temporal data

5.1.1 Data types and structures

According to Güting and Schneider (as cited in Bivand et al., 2013), there are different types of spatio-temporal data.

- Events that do not last in time and appear at a specific location : It could be a death, an accident.
- Events that last for a period of time but no movement is observed. For example, a tree, the address of a person.
- A moving point : Something that moves from a place to another during a certain period of time. For instance, a person that moves from his house to his workplace.

The data of the study are moving points as concentrations may be different from one place to another and from one time to another. These variations could be explained by physical factors such as windspeed or temperature.

There are two types of data structure :

- Regular data : the number of locations considered for each time is the same. For this type of data structure, the spatial grid can be full (data for each location in the spatial domain) or sparse (only few locations in the spatial domain).

- Irregular data : The number of locations considered for each time is different. This type of structure is the most common. In the context of the study, the dataset has an irregular data structure.

The number of available locations and the time coverage of the dataset usually determin the modelling method to use. If data are dense in time but very sparse in space, multivariate time series models can be considered as an appropriate choice. However, data has to be full over time. Otherwise, building Auto-regressive models will not be possible. If data are dense in space but very sparse in time, performing a kriging for each available time can be a solution. If data are dense in space and time, the space-time interaction can be modelled. And various models are possible to model such interaction. The data from the study are dense enough in space and time. Therefore, the spatio-temporal variability is modelled. More information is given later.

5.1.2 Space-time interaction

A first important question that should be asked when considering these two factors of variability (space and time) together is : Is there an interaction between space and time ?

If the interaction is not significant : This would mean that their impacts are separable. As a result, the response variable could be expressed as follows :

$$Y(s,t) = \phi(t) + \psi(s) + \epsilon \tag{5.1.1}$$

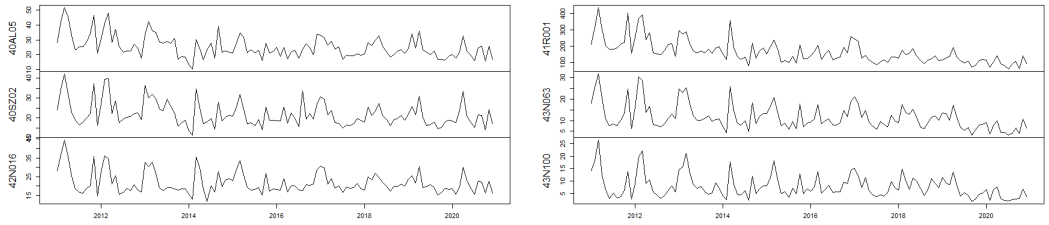
Where ϕ and ψ are respectively functions of time t and space s. And ϵ is an error term.

If the interaction is significant : This would indicate that the temporal structure varies from place to place or that the spatial structure differs from time to time. In such case, the response could be expressed as follows.

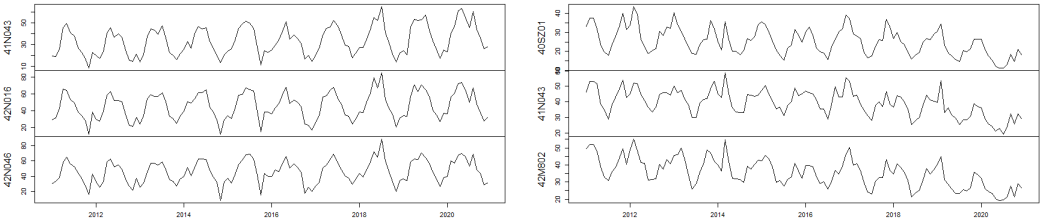
$$Y(s,t) = \phi(t) + \psi(s) + \theta(s,t) + \epsilon \tag{5.1.2}$$

Where ϕ and ψ are respectively functions of time t and space s. And θ is a space-time function (interaction function). ϵ is an error term.

Each pollutant may react differently in time and space. A first approach would be to visualize the evolution of the concentrations for some randomly chosen stations for the whole period of time. 3 stations are randomly chosen and the time series are plotted for each pollutant. Concentrations are aggregated per month. The results are shown below.



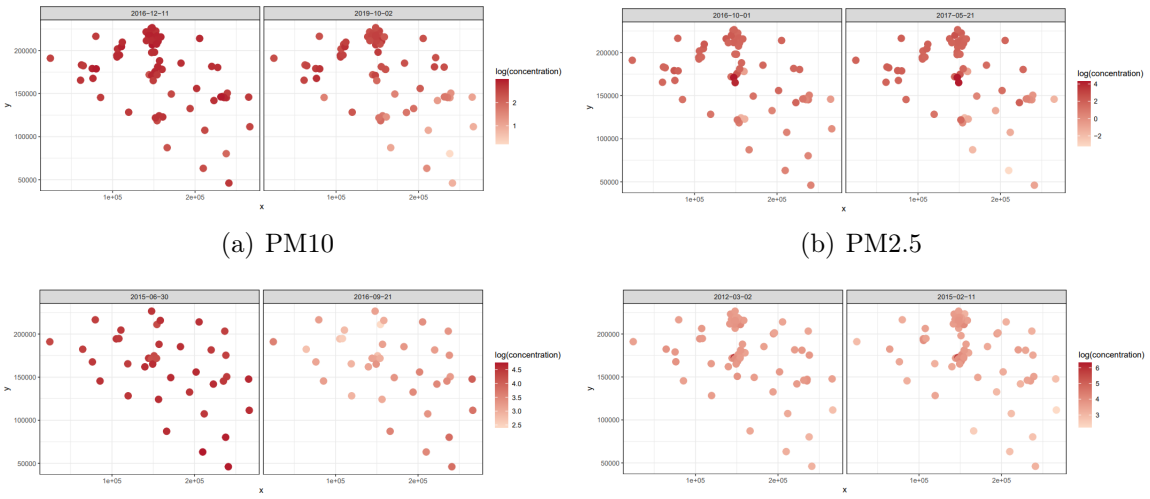
(a) PM10 (b) PM2.5



(c) O3 (d) NO2

Figure 5.1.1: Mean of the concentrations from January 2011 to December 2020

The temporal structure seems to be roughly the same regardless of the station. PM10 and PM2.5 show irregular peaks. O3 and NO2 have a more regular structure with clear seasonal patterns. Now, the contrary is done : 2 random dates are chosen. The logarithm is applied in order to obtain a more linear growth in the values. The results are shown for the whole spatial domain.



(a) PM10 (b) PM2.5

(c) O3 (d) NO2

The maps shown above give an indication of how concentrations can vary through space. Values can be higher in the north of the country at a time and not at another time. As seen before, O3 concentrations are expected to be higher during the summer period. The maps for the pollutant O3 confirm this particular behavior : higher values in July than in September.

5.2 Spatio-temporal regression kriging (STRK)

The method used for modelling is spatio-temporal regression kriging (STRK). This method combines a spatio-temporal regression with a spatio-temporal kriging of the regression residuals. Regression residuals are thus assumed to be correlated in space and time. The regression is said to be spatio-temporal because of the space and time dependence. It is basically a multiple linear regression.

The response variable is the logarithm of the average concentration of each day. As seen in the previous chapters, concentration levels can reach very high values at some period of times. Therefore, the logarithm is particularly adapted in this case. The logarithm will make the data more linear and will reduce the effect of the extreme values.

Some explanatory variables cited in chapter 1 are available only for the year 2019 and 2020. Hence, the model is built from the period : January 2019 to November 2020. This ensures that all available variables are present in the model. The amount of data is assumed to be sufficiently large enough to obtain good results. Indeed, for a station that has no missing value in the whole period, this would represent 700 observations. The goal is to predict concentration levels both in time and space for each day in December 2020.

5.2.1 Regression

5.2.1.1 Model description

The proposed model is a model that makes the assumption that the impacts of the explanatory variables on the concentrations are different depending on the pollutant. For example, the effect of the temperature on O3 may be different than that on PM10. The dependent variable is explained by variables that are dependent on time, on space and both. The simplified model can be expressed as follows :

$$Y(s,t) = \alpha + \underbrace{\phi(t) + \psi(s) + \theta(s,t)}_{\text{determinist}} + \underbrace{\epsilon(s,t)}_{\text{stochastic}} \quad (s,t) \in S \times T \tag{5.2.1}$$

Y(s,t) : response variable at time t in location s
α : intercept
$\phi(t)$: function of time (includes space invariant variables)
$\psi(s)$: function of space (includes time invariant variables)
$\theta(s,t)$: function of space and time (includes variables that vary in space and time (e.g

temperature)
$\epsilon(s,t)$: Random field that can be decomposed as follows :

$$\epsilon(s,t) = Z(s,t) + e \tag{5.2.2}$$

Where $Z(s,t)$ is a second-order stationary random field with mean 0 and covariance function $C_{st}(h,u)$ and e are error terms without dependence structure (white noise). More details about this term are given in the sequel.

Second-order stationary : Processus that has a constant mean and a covariance function that can be expressed with spatial and temporal lags.

$$C_{st}(h,u) = Cov(Z(s,t), Z(s+h,t+u)) \quad (s,t) \text{ and } (s+h,t+u) \in S \times T \tag{5.2.3}$$

h : space lag
u : time lag

To get a better understanding of this model, an illustration is given below where the dependence is only restricted to space with only one dimension.

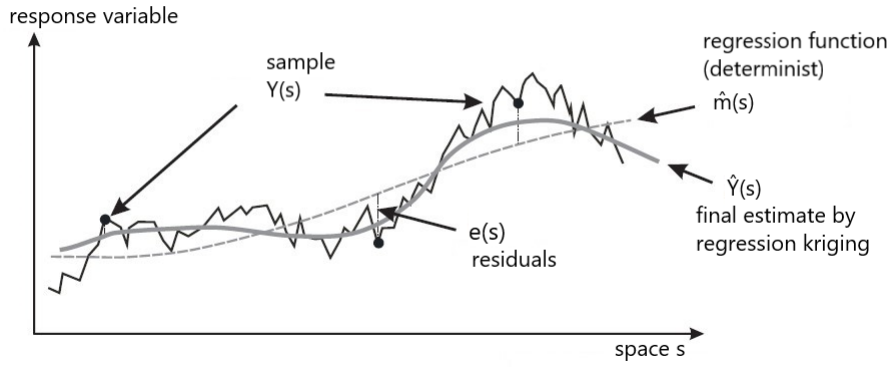


Figure 5.2.1: Regression kriging with spatial dependence

Model 1

A first model is performed using all the available variables and two new variables described below:

- t : quantitative variable that goes from 1 to 731 (number of days between January 2019 and December 2020). This variable is used to estimate a coefficient relative to the trend.

- *jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov* and *dec* : Dummy variable that takes the value 1 if it corresponds to the associated month, 0 otherwise. These variables are used to represent the seasonality.

The first model is expressed as follows :

$$Y_{ijt} = \alpha_j + \underbrace{\beta_j t + \sum_{m=1}^{M-1} \delta_{jm} D_{mt} + \phi_{jwkd} D_{wkdt}}_{\text{function of time}} + \underbrace{\sum_{k=1}^{K} \gamma_{jk} X_{ik} + \sum_{r=1}^{R-1} \eta_{jr} D_{ir}}_{\text{function of space}} + \underbrace{\sum_{l=1}^{L} \tau_{jl} W_{itl}}_{\text{interaction}} + \epsilon_{ijt} \tag{5.2.4}$$

j = 1,2,3,4 m = 1,...,12 t = 1,...,700

In the model, there are qualitative variables : D_{mt}, D_{wkdt} and D_{ir}. To avoid a problem of multicollinearity in the model, for a categorical variable that has K levels, there will be K - 1 estimates. The level that is not present in the model is called the reference level. The reference levels of the qualitative variables D_{mt}, D_{wkdt}, D_{ir} are respectively *jan* (january), *workdays* and *Wallonia*. The model terms are defined below.

Y_{ijt} : logarithm of the concentration of the pollutant j at time t in station i.
α_j : intercept of the pollutant j.

Time components

β_j : trend estimate coefficient of the pollutant j.
t : time, quantitative variable that goes from 1 to 700 (700 days between 1st January 2019 to 30 November 2020).
δ_{jm} : estimate coefficient of the pollutant j for the month m.
D_{mt} : dummy variable that takes the value 1 if the variable t corresponds to the month m, 0 otherwise.
ϕ_{jwkd} : estimate coefficient of the variable *week_ time* for the pollutant j.
D_{wkdt} : dummy variable that takes the value 1 if the variable t corresponds to the weekend, 0 otherwise.

Space components

γ_{jk} : estimate coefficient of the explanatory variable X_k for the pollutant j.
X_{ik} : quantitative explanatory variables X_k observed in the station i : *LPS_ closest*, *city_ closest*, *airport_ closest*, *denspop*, *vegetation*, *x coordinate*, *y coordinate*.
η_{jr} : estimate coefficient for the region r and pollutant j.
D_{ir} : dummy variable that takes the value 1 if the station i is located in the region r, 0 otherwise.

Space-time interactions

τ_{jl} : estimate coefficient of the variable W_l for the pollutant j.
W_{itl} : explanatory variables W_l observed at time t in station i : *temperature*, *pressure*, *precipitation*, *humidity* and *windspeed*.

The idea behind this model is to capture the following components : the trend over time that varies according to the pollutant, the seasonality represented by the dummy variable D_{mt}, the spatial variability of the dependent variable that partially depends both on the pollutant and the characteristics of the location (region, population density...). The variables W_l are used to explain the potential interaction between space and time on the response variable.

Model 2

The number of estimated parameters relative to seasonality in the previous model is 11 which is high. To obtain a more parsimonious model, a different method is used. Instead, the sinus and cosinus functions are used to explain the seasonal variability. Two new variables are created (X1 and X2). These two variables are functions of d. d corresponds to the number of the day within a year (1 to 365). Expressions are given below.

$$X1(d) = sin\left(\frac{2\pi d}{365}\right) \qquad X2(d) = cos\left(\frac{2\pi d}{365}\right) \qquad d = 1, 2, ..., 365 \tag{5.2.5}$$

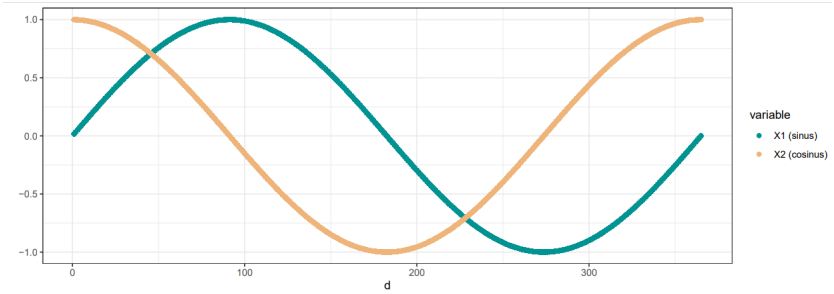


Figure 5.2.2: X1 and X2 represented as a function of d

From these two variables, coefficient estimates can be obtained for each pollutant. Finally, one can express a seasonal value to a pollutant for a value of X1 and X2.

$$S_j = \alpha_j X1 + \beta_j X2 \qquad j = 1, 2, 3, 4 \tag{5.2.6}$$

Where S_j is the seasonal value attributed to pollutant j for two values of X1 and X2. By using this method, the number of parameters for seasonality drops from 11 to 2. The updated model is shown below.

$$Y_{ijt} = \alpha_j + \beta_j t + \delta_j X_1 + \omega_j X2 + \phi_{jwkd} D_{wkdt} + \sum_{k=1}^{K} \gamma_{jk} X_{ik} + \sum_{r=1}^{R-1} \eta_{jr} D_{ir} + \sum_{l=1}^{L} \tau_{jl} W_{itl} + \epsilon_{ijt} \tag{5.2.7}$$

5.2.1.2 Model Estimates

The estimates from both model are shown for each pollutant. The method used for estimation is Ordinary Least Squares (OLS).

Each estimate results from a Student test that tests if the coefficient is significantly different from 0 or not. The test statistic is expressed as follows.

$$T_i = \frac{\hat{\beta}_i - \beta_i}{s.e(\hat{\beta}_i)} \tag{5.2.8}$$

$\hat{\beta}_i$ is the ith estimate. $s.e(\hat{\beta}_i)$ is the standard error of the ith estimate.

The test hypotheses are : H0 = $\beta_i = 0$ against H1 : $\beta_i \neq 0$.
Under the null hypothesis H0, the test statistic T_i follows a Student distribution with n-p degrees of freedom.

$$T_i = \frac{\hat{\beta}_i}{s.e(\hat{\beta}_i)} \sim t_{n-p} \tag{5.2.9}$$

n is the size of the sample. p is the number of estimated parameters. p = 28 for model 1 and p = 19 for model 2. Once the test statistic is computed, a p-value can be retrieved. The p-value is a measure of discrepancy between data and a null hypothesis (Spiegelhalter, 2019). The lower the p-value, the better the evidence against the null hypothesis. The p-value represents the probability of observing such an extreme value when H0 is true. Usually a bi-sided test is performed. The p-value noted P_{obs} is expressed as follows.

$$P_{obs} = 2 \times P(T > |t|) \tag{5.2.10}$$

Where t is the observed test statistic obtained from the sample. The common threshold used for p-values is $\alpha = 0.05$. If the p-value is less than or equal to α, the null hypothesis is rejected. Another way to make the decision is to compare t with the Student quantile $t_{n-p}(1-\alpha/2)$. If $|t| > t_{n-p}(1-\alpha/2)$, the null hypothesis is rejected. The coefficient estimates are shown below for model 1 and model 2. The stars ($*$) represent the significance level of the test. The higher the number of stars, the higher the evidence against the null hypothesis. If a point is displayed just after the p-value, this means that the pvalue is very close to the threshold (0.05).

	PM10			PM2.5		
	Estimate	Std.error	Pr(>|t|)	Estimate	Std.error	Pr(>|t|)
(Intercept)	4.272e+00	1.060e-01	<2e-16***	-8.822e-01	1.848e-01	1.81e-06***
t	-1.522e-04	1.210e-05	<2e-16***	-3.074e-04	2.120e-05	<2e-16***
feb1	7.555e-02	1.075e-02	2.16e-12***	5.290e-02	1.890e-02	0.005142**
mar1	-3.414e-02	1.081e-02	0.001586**	-3.528e-02	1.896e-02	0.062818.
apr1	4.412e-02	1.201e-02	0.000240***	-9.704e-03	2.107e-02	0.645112
may1	-2.954e-01	1.207e-02	<2e-16***	-4.854e-01	2.115e-02	<2e-16***
jun1	-5.226e-01	1.407e-02	<2e-16***	-8.269e-01	2.467e-02	<2e-16***
jul1	-6.791e-01	1.433e-02	<2e-16***	-1.080e+00	2.512e-02	<2e-16***
aug1	-5.896e-01	1.532e-02	<2e-16***	-8.890e-01	2.686e-02	<2e-16***
sep1	-5.157e-01	1.333e-02	<2e-16***	-8.707e-01	2.338e-02	<2e-16***
oct1	-4.255e-01	1.170e-02	<2e-16***	-6.436e-01	2.051e-02	<2e-16***
nov1	-8.219e-02	1.118e-02	1.99e-13***	-1.118e-01	1.959e-02	1.15e-08***
dec1	-2.944e-02	1.261e-02	0.019552*	-3.340e-02	2.210e-02	0.130806
week_timeweekend	-6.722e-02	4.766e-03	<2e-16***	3.914e-02	8.349e-03	2.77e-06***
vegetation	3.680e-04	1.736e-04	0.033984*	1.161e-03	3.039e-04	0.000133***
denspop	4.428e-06	1.955e-06	0.023516*	8.945e-05	3.424e-06	<2e-16***
regionBrussels	1.992e-01	1.604e-02	<2e-16***	1.041e+00	2.894e-02	<2e-16***
regionFlanders	5.849e-01	1.379e-02	<2e-16***	2.318e-01	2.403e-02	<2e-16***
airport_closest	-1.987e-06	1.856e-07	<2e-16***	1.399e-05	3.227e-07	<2e-16***
LPS_closest	-2.846e-05	7.292e-07	<2e-16***	-1.898e-05	1.275e-06	<2e-16***
city_closest	-1.985e-06	2.226e-07	<2e-16***	-2.618e-06	3.883e-07	1.56e-11***
x	-1.661e-06	5.330e-08	<2e-16***	-8.059e-07	9.284e-08	<2e-16***
y	-2.900e-06	1.707e-07	<2e-16***	9.239e-06	2.972e-07	<2e-16***
temperature	2.196e-02	6.772e-04	<2e-16***	1.915e-02	1.185e-03	<2e-16***
precipitation	-1.770e-02	5.114e-04	<2e-16***	-1.221e-02	9.022e-04	<2e-16***
pressure	1.126e-02	2.990e-03	0.000167***	6.561e-02	5.210e-03	<2e-16***
windspeed	-4.288e-02	3.467e-04	<2e-16***	-6.545e-02	6.079e-04	<2e-16***
humidity	-6.622e-03	2.839e-04	<2e-16***	8.818e-04	4.969e-04	0.075996.

Table 5.2.1: Model 1 (PM10 and PM2.5)

	O3			NO2		
	Estimate	Std.error	Pr(>|t|)	Estimate	Std.error	Pr(>|t|)
(Intercept)	4.324e+00	1.052e-01	<2e-16***	5.248e+00	1.149e-01	<2e-16***
t	1.252e-05	1.295e-05	0.33347	-4.909e-04	1.228e-05	<2e-16***
feb1	1.009e-02	1.167e-02	0.38729	2.455e-02	1.093e-02	0.024710*
mar1	2.522e-01	1.171e-02	<2e-16***	-2.066e-01	1.098e-02	<2e-16***
apr1	3.450e-01	1.276e-02	<2e-16***	-4.429e-01	1.222e-02	<2e-16***
may1	3.487e-01	1.281e-02	<2e-16***	-5.611e-01	1.231e-02	<2e-16***
jun1	1.412e-01	1.489e-02	<2e-16***	-6.304e-01	1.430e-02	<2e-16***
jul1	-4.455e-02	1.519e-02	0.00335**	-7.696e-01	1.459e-02	<2e-16***
aug1	-6.421e-02	1.616e-02	7.12e-05***	-6.519e-01	1.559e-02	<2e-16***
sep1	-1.194e-01	1.414e-02	<2e-16***	-3.954e-01	1.357e-02	<2e-16***
oct1	-1.943e-01	1.253e-02	<2e-16***	-2.896e-01	1.185e-02	<2e-16***
nov1	-3.782e-01	1.186e-02	<2e-16***	-3.556e-02	1.134e-02	0.001713**
dec1	-1.995e-01	1.343e-02	<2e-16***	3.614e-02	1.281e-02	0.004768**
week_timeweekend	7.203e-03	5.095e-03	0.15748	-2.711e-01	4.835e-03	<2e-16***
vegetation	3.070e-03	1.845e-04	<2e-16***	-6.190e-03	1.641e-04	<2e-16***
denspop	-2.418e-06	1.599e-06	0.13053	2.986e-05	1.521e-06	<2e-16***
regionBrussels	2.089e-02	1.375e-02	0.12869	-5.296e-02	1.437e-02	0.000229***
regionFlanders	-4.522e-02	1.058e-02	1.94e-05***	2.704e-02	1.192e-02	0.023321*
airport_closest	1.807e-06	2.072e-07	<2e-16***	-3.506e-06	2.003e-07	<2e-16***
LPS_closest	1.103e-05	6.930e-07	<2e-16***	-2.424e-05	7.706e-07	<2e-16***
city_closest	-4.675e-07	2.182e-07	0.03211*	-3.001e-06	2.305e-07	<2e-16***
x	1.021e-06	4.827e-08	<2e-16***	-1.637e-06	6.018e-08	<2e-16***
y	-8.306e-07	1.476e-07	1.86e-08***	1.624e-06	1.634e-07	<2e-16***
temperature	3.891e-02	7.203e-04	<2e-16***	-4.666e-05	6.898e-04	0.946063
precipitation	3.708e-03	5.244e-04	1.57e-12***	4.306e-03	5.140e-04	<2e-16***
pressure	-3.132e-02	2.936e-03	<2e-16***	-8.156e-03	3.298e-03	0.013394*
windspeed	3.235e-02	3.727e-04	<2e-16***	-4.954e-02	3.591e-04	<2e-16***
humidity	-1.183e-02	3.079e-04	<2e-16***	-6.572e-03	2.884e-04	<2e-16***

Table 5.2.2: Model 1 (O3 and NO2)

	PM10			PM2.5		
	Estimate	Std.error	Pr(>|t|)	Estimate	Std.error	Pr(>|t|)
(Intercept)	4.239e+00	1.052e-01	<2e-16***	-9.656e-01	1.828e-01	1.27e-07***
t	-8.904e-05	1.176e-05	3.68e-14***	-2.036e-04	2.053e-05	<2e-16***
X1	2.333e-01	3.795e-03	<2e-16***	3.485e-01	6.628e-03	<2e-16***
X2	3.424e-01	5.742e-03	<2e-16***	5.397e-01	1.003e-02	<2e-16***
week_timeweekend	-6.446e-02	4.794e-03	<2e-16***	4.397e-02	8.374e-03	1.51e-07***
vegetation	4.699e-04	1.746e-04	0.00712**	1.316e-03	3.049e-04	1.59e-05***
denspop	4.873e-06	1.967e-06	0.01326*	9.006e-05	3.436e-06	<2e-16***
regionBrussels	1.929e-01	1.614e-02	<2e-16***	1.033e+00	2.904e-02	<2e-16***
regionFlanders	5.854e-01	1.387e-02	<2e-16***	2.332e-01	2.411e-02	<2e-16***
airport_closest	-2.071e-06	1.867e-07	<2e-16***	1.386e-05	3.238e-07	<2e-16***
LPS_closest	-2.877e-05	7.336e-07	<2e-16***	-1.943e-05	1.279e-06	<2e-16***
city_closest	-1.871e-06	2.239e-07	<2e-16***	-2.461e-06	3.896e-07	2.70e-10***
x	-1.685e-06	5.363e-08	<2e-16***	-8.390e-07	9.315e-08	<2e-16***
y	-3.090e-06	1.715e-07	<2e-16***	8.961e-06	2.980e-07	<2e-16***
temperature	2.608e-02	6.727e-04	<2e-16***	2.553e-02	1.174e-03	<2e-16***
precipitation	-1.786e-02	5.092e-04	<2e-16***	-1.261e-02	8.956e-04	<2e-16***
pressure	5.969e-03	2.996e-03	0.04631*	5.675e-02	5.206e-03	<2e-16***
windspeed	-4.471e-02	3.420e-04	<2E-16***	-6.809e-02	5.981e-04	<2E-16***
humidity	-7.640e-03	2.730e-04	<2E-16***	-3.358e-04	4.763e-04	0.481

Table 5.2.3: Model 2 (PM10 and PM2.5)

	O3			NO2		
	Estimate	Std.error	Pr(>|t|)	Estimate	Std.error	Pr(>|t|)
(Intercept)	4.360e+00	1.060e-01	<2e-16***	4.824e+00	1.132e-01	<2e-16***
t	-1.449e-05	1.277e-05	0.256	-4.731e-04	1.185e-05	<2e-16***
X1	2.464e-01	4.115e-03	<2e-16***	4.029e-02	3.820e-03	<2e-16***
X2	-1.323e-01	6.158e-03	<2e-16***	4.174e-01	5.793e-03	<2e-16***
week_timeweekend	7.007e-03	5.203e-03	0.178	-2.695e-01	4.830e-03	<2e-16***
vegetation	3.108e-03	1.885e-04	<2e-16***	-6.234e-03	1.639e-04	<2e-16***
denspop	-2.478e-06	1.634e-06	0.129	2.978e-05	1.520e-06	<2e-16***
regionBrussels	2.112e-02	1.404e-02	0.133	-5.239e-02	1.436e-02	0.000265***
regionFlanders	-4.805e-02	1.081e-02	8.89e-06***	2.394e-02	1.191e-02	0.044451*
airport_closest	1.802e-06	2.118e-07	<2e-16***	-3.447e-06	2.002e-07	<2e-16***
LPS_closest	1.101e-05	7.082e-07	<2e-16***	-2.415e-05	7.702e-07	<2e-16***
city_closest	-4.700e-07	2.229e-07	0.035*	-3.002e-06	2.303e-07	<2e-16***
x	1.022e-06	4.931e-08	<2e-16***	-1.632e-06	6.014e-08	<2e-16***
y	-8.335e-07	1.507e-07	3.24e-08***	1.624e-06	1.632e-07	<2e-16***
temperature	3.781e-02	7.276e-04	<2e-16***	3.214e-03	6.804e-04	2.32e-06***
precipitation	4.563e-03	5.298e-04	<2e-16***	3.957e-03	5.080e-04	6.82e-15***
pressure	-2.907e-02	2.989e-03	<2e-16***	-6.684e-03	3.279e-03	0.041489*
windspeed	3.244e-02	3.734e-04	<2e-16***	-4.971e-02	3.518e-04	<2e-16***
humidity	-1.284e-02	2.999e-04	<2e-16***	-6.485e-03	2.754e-04	<2e-16***

Table 5.2.4: Model 2 (O3 and NO2)

In linear regression, coefficients can be interpreted differently with the presence or absence of other variable(s). As a result, a variable that was not significant in model 1 can become significant in model 2 (e.g temperature with the pollutant NO2).
The performances of both models are studied using the following indicators.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2} \tag{5.2.11}$$

The root mean squared error is used instead of the mean squared error (MSE) because it is expressed in the same unit as the response variable.

$$\text{AIC} = -2\log(\tilde{L}) + 2p \tag{5.2.12}$$

Where p is the number of estimated parameters in the model. AIC is a famous information criterion in statistical modelling that takes into account the likelihood and complexity of the model. $\tilde{L}$ is the maximum of the likelihood function. The likelihood function is the probability of observing the sample data, given a set of model parameters.

$$\text{BIC} = -2\log(\tilde{L}) + \text{p} \times \log(n) \tag{5.2.13}$$

As AIC, BIC takes both the likelihood and complexity of the model. In contrast to AIC, BIC will penalise models with more variables. The lower the AIC and BIC, the better the

model. Note that AIC and BIC are not useful if they are used alone. They must be used to compare different models.

$$R^2_{adj} = 1 - \frac{SSE/(n-1)}{SST(n-p)} \tag{5.2.14}$$

The adjusted R^2 penalises models with a high number of parameters. As opposed to the original R^2, adjusted R^2 can be used to compare models with different number of parameters. Since model 1 and model 2 have different number of parameters, adjusted R^2 is used.

	model 1				model 2			
	PM10	PM2.5	O3	NO2	PM10	PM2.5	O3	NO2
RMSE	0.48	0.83	0.38	0.48	0.48	0.83	0.39	0.48
AIC	-73,712.67	-18,104.54	-53,422.09	-72,133.48	-73,081.41	-17,760.05	-52,226.79	-72,188.51
BIC	68,779.58	121,926.8	25,504.4	69,240.41	69,331.43	122,192	26,625.63	69,106.06
R^2_{adj}	0.54	0.49	0.58	0.64	0.53	0.49	0.56	0.64

Table 5.2.5: Performances of model 1 and model 2 for each pollutant

RMSE are equal between the two models except for the pollutant O3 where model 1 has a slightly lower RMSE than model 2. AIC and BIC in model 1 are smaller (better) for PM10, PM2.5 and O3. Model 2 is better for the pollutant NO2 : lower AIC and BIC. Performances from both models are nearly similar. But model 2 has less parameters (more parsimonious). In addition, as opposed to model 2, seasonal variables in model 1 are all significant. Model 2 is thus retained for the rest of the study.

5.2.1.3 Multicollinearity

Multicollinearity is an important problem that should be carefully checked in a modelling procedure. It can have undesirable consequences on the model such as : high variance of the estimators, sensibility of the coefficients when adding or removing variables... 2 methods are used to check potential multicollinearity in the model : pairwise correlation to detect linear relationship between two quantitative variables and the variance inflation factor (VIF) that detects more complicated correlations between variables.

Pairwise correlation

The coefficient of correlation used is the correlation of Pearson. This coefficient measures the relationship between two quantitative variables.

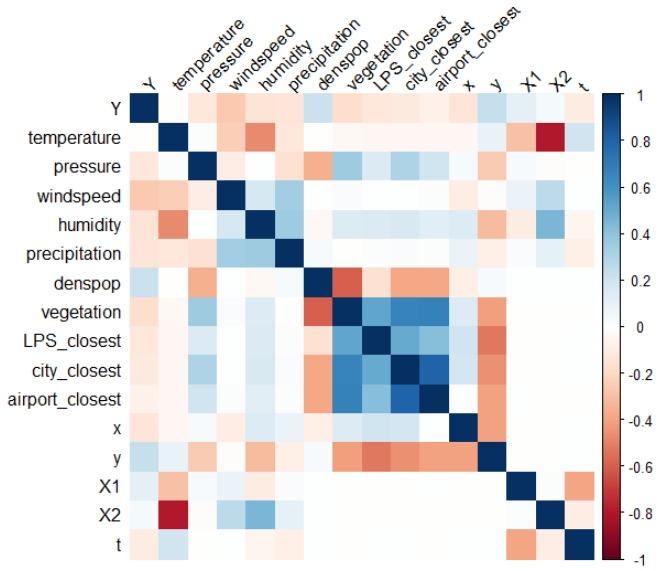
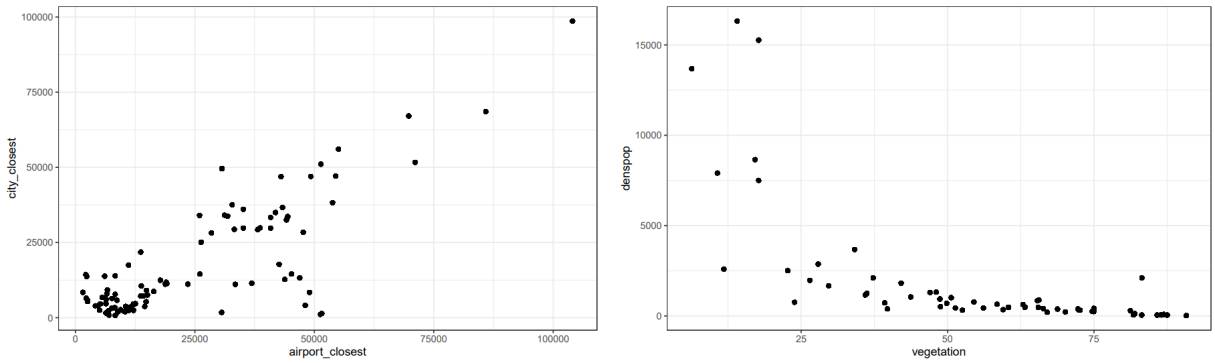


Figure 5.2.3: Correlation matrix

High pairwise correlations are plotted below.



(a) r = 0.81 (b) r = - 0.6

As we can see from the scatter plots, some relationships are linear but some are non linear. The variables *airport_closest* and *city_closest* are highly correlated (0.81) because most of the airports in Belgium are close to major cities. *vegetation* is correlated with *denspop* (-0.6) but the relationship is exponential (exponential decay rather than linear).

Variation inflation factor (VIF)

The VIF is computed as follows :

$$VIF_k = \frac{1}{1 - R_k^2} \qquad k = 1, .., p \tag{5.2.15}$$

Where R_k^2 is a coefficient of determination. This coefficient is the result of a regression where the explanatory variable X_k is regressed over all the other explanatory variables $X_1, ..., X_{k-1}, X_{k+1}, ..., X_p$.

Variable	Variation inflation factor (VIF)
temperature	3.99
pressure	1.38
precipitation	1.34
windspeed	1.27
humidity	1.82
X1	1.61
X2	3.41
vegetation	3.15
denspop	1.85
city_closest	3.66
airport_closest	3.77
LPS_closest	1.74
x	1.41
y	2.14
t	1.22

Table 5.2.6: VIF of the quantitative variables

A commonly used threshold is 10. If the VIF exceeds 10, the presence of multicollinearity in the model can be claimed. The average VIF is **2.25**. Hence there is no evidence of multicollinearity in the model.

5.2.1.4 Variable selection

Model 2 is used as a full starting model as he is more parsimonious than model 1. We have seen previously that some variables were not significant for some pollutant. We could then apply a backward stepwise selection procedure from the full starting model. This method removes sequentially variables that are statistically insignificant. A variable that is removed can be re-injected in the model if the variable becomes significant with the presence of the other variables. The p-value is used as a selection criterion so that all variables become significant at the end of the procedure. A Bonferroni adjustment is used on the p-values to avoid accumulation of type I error. The accumulation of type I error is explained below.

Suppose that one test is performed with level $\alpha = 0.05$, the probability of getting no false positive is :

$$P(\text{no false positive}) = 1 - \alpha = 0.95 \tag{5.2.16}$$

Hence the contrary is :

$$P(\text{getting 1 false positive}) = 1 - (1 - \alpha) = 0.05 \tag{5.2.17}$$

Suppose now that there are K independent tests with significance level α. The probability P of getting at least 1 false positive among the K tests is :

$$P = 1 - (1 - \alpha)^K \tag{5.2.18}$$

In others words, as the number of test increases, the probability of getting at least 1 false positive increases.

Some variables are fixed in the model : t, X1 and X2. Hence, they cannot be removed in the selection procedure. The formulas of the final models are given below.

Note that other famous variable selection methods could have been used such as a forward selection, Lasso or Ridge. The final models of each pollutant are shown below.

PM10 model

$$Y_{it} = \alpha + \beta_t + \delta X_1 + \omega X_2 + \phi_{wkd} D_{wkdt} + \sum_{k=1}^{K} \gamma_k X_{ik} + \sum_{l=1}^{L} \tau_l W_{itl} + \epsilon_{it} \tag{5.2.19}$$

Variable excluded by the variable selection method : region.

PM2.5 model

$$Y_{it} = \alpha + \beta t + \delta X_1 + \omega X_2 + \phi_{wkd} D_{wkdt} + \sum_{k=1}^{K} \gamma_k X_{ik} + \sum_{r=1}^{R-1} \eta_r D_{ir} + \sum_{l=1}^{L} \tau_l W_{itl} + \epsilon_{it} \tag{5.2.20}$$

Variable excluded by the variable selection method : precipitation.

O3 model

$$Y_{it} = \alpha + \beta t + \delta X_1 + \omega X_2 + \phi_{wkd} D_{wkdt} + \sum_{k=1}^{K} \gamma_k X_{ik} + \sum_{r=1}^{R-1} \eta_r D_{ir} + \sum_{l=1}^{L} \tau_l W_{itl} + \epsilon_{it} \tag{5.2.21}$$

Variables excluded by the variable selection method : denspop, pressure and windpseed.

NO2 model

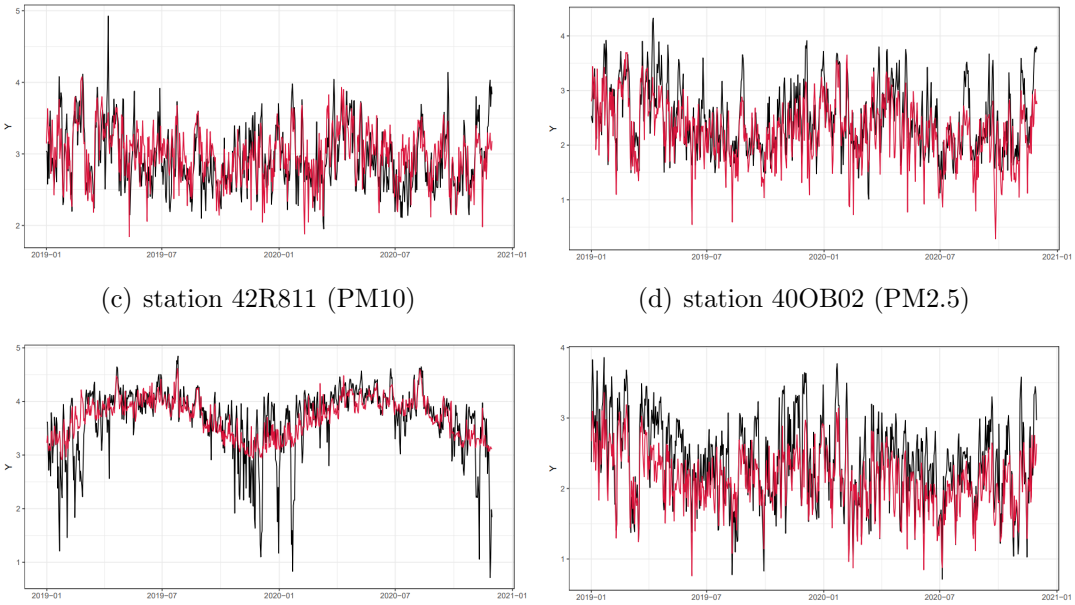
$$Y_{it} = \alpha + \beta_t + \delta X_1 + \omega X_2 + \phi_{wkd} D_{wkdt} + \sum_{k=1}^{K} \gamma_k X_{ik} + \sum_{r=1}^{R-1} \eta_r D_{ir} + \sum_{l=1}^{L} \tau_l W_{itl} + \epsilon_{it} \tag{5.2.22}$$

Variable excluded by the variable selection method : pressure

Coefficient estimates of each model are given in Appendix.

5.2.1.5 Performance

The performances of the new models are now inspected. The graphs of the observed values (black) against the predicted values (red) over time are represented below. One station is chosen randomly for each pollutant.

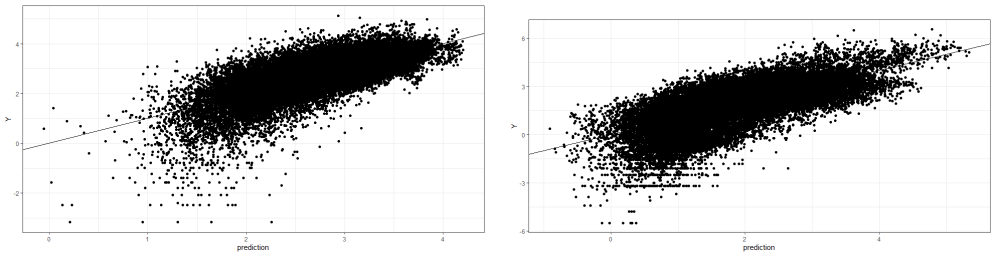


(c) station 42R811 (PM10) (d) station 40OB02 (PM2.5)

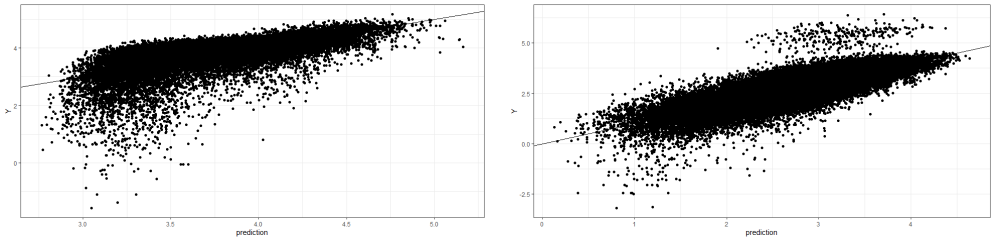
(e) station 41R001 (O3) (f) station 43N063 (NO2)

Figure 5.2.4: Observed against predicted for 1 randomly chosen station

The temporal structure is well captured in each model. However, a constant bias persists over time (e.g NO2 : station 43N063).



(a) PM10 : Pearson correlation = 0.72 (b) PM2.5 : Pearson correlation = 0.7



(c) O3 : Pearson correlation = 0.66 (d) NO2 : Pearson correlation = 0.8

Figure 5.2.5: Observed against predicted

The Pearson correlation between the observed values and the predictions is computed. In addition, a 1:1 line is added to see how well each model predicts.

One can observe a group of observed values greater or equal than 5 for NO2. This group of points corresponds to high observed concentrations. It corresponds mainly to stations located in the region of Brussels.

High residuals are observed for the O3 model (particularly low values). This model overestimates the very low concentration levels.

Each regression model is now evaluated using different indicators. Results are shown in the table below.

Model	PM10	PM2.5	O3	NO2
RMSE	0.49	0.84	0.44	0.48
AIC	-71,331.97	-17,64.02	-45,375.52	-72,186.35
BIC	71,063.22	122,379.2	33,452.21	69,099.4
Adjusted R^2	0.51	0.48	0.44	0.64

Table 5.2.7: Performance of each model

The adjusted R^2 is computed so that the performance of each model can be compared. NO2 seems to be the most efficient model with an adjusted R^2 of 0.64.

5.2.1.6 Cross validation

The performance of each model can be better evaluated by using cross validation. A K-FOLD cross validation wouldn't be appropriate since there is a dependence in time between observations. Indeed, if a K-FOLD cross validation would have been done, some models would have been built to predict past data instead of future data. To tackle this problem, a one-step ahead prediction can be done using 80% of the data from the training set. And successively use the remaining data to predict values for one day. The number of predicted values for one day corresponds to the number of stations for which there are observations for this day in particular.

There are 700 days in the training set. 560 days (80%) are used to predict the day number 561. Then, 561 days are used to predict the day number 562 and so on. The RMSE is computed for each predicted day. At the end, the average and standard deviation of the RMSE are calculated. Results are shown below :

	PM10	PM2.5	O3	NO2
Average RMSE	0.45	0.83	0.36	0.44
Standard deviation of the RMSE	0.19	0.2	0.34	0.09

Table 5.2.8: Results from cross validation

The average RMSE is similar than the RMSE computed before (see table 5.2.7). The standard deviation of the RMSE for NO2 is low (0.09). This indicates that the performance of the model remains constant over time. It is less the case for PM10, PM2.5 and O3.

5.2.2 Diagnostics

5.2.2.1 Residuals analysis

Spatial autocorrelation

Each observed valued is related to a specific location and time period. Thus, multiple Moran tests can be computed for each time t. Performing multiple tests leads to the problem of accumulation of type 1 error. In the study, 700 (number of days for which a Moran test is applied) tests are performed which results in a very high probability of getting at least 1 false positive among the 700 tests (roughly 1). The method used is the Bonferroni correction. In this method, the level α is divided by the number of tests. Hence the Bonferroni corrected pvalue is:

$$\alpha^* = \frac{\alpha}{n} = \frac{0.05}{700} = 0.0000714 \tag{5.2.23}$$

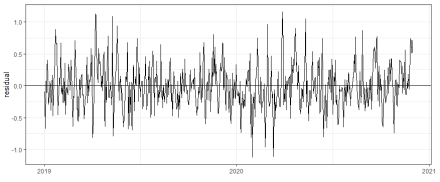
Model	PM10	PM2.5	O3	NO2
Proportion of significant test (%)	46	44	28	6

Table 5.2.9: Proportion of significant Moran tests

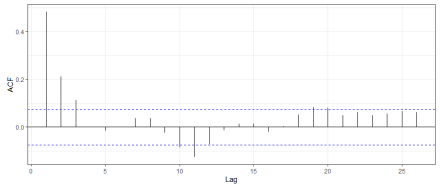
There is a strong evidence that spatial autocorrelation remains in the regression residuals of PM10, PM2.5 and O3. Spatial autocorrelation can be present at a time and not present at another time. However, it seems that for NO2, the regression model captures well the spatial variability : only 6% of the tests are significant. That is, only 6% of the tests reject the hypothesis of absence of spatial autocorrelation.

Temporal autocorrelation

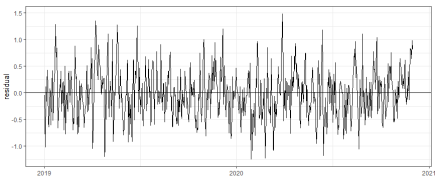
In addition to the spatial correlation, temporal correlation can be also analyzed. This time, the data have to be aggregated with respect to the time.



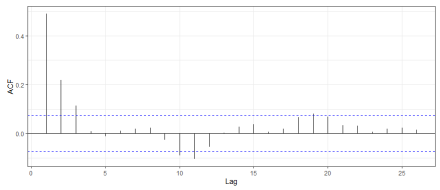
(a) Aggregated residuals (PM10)



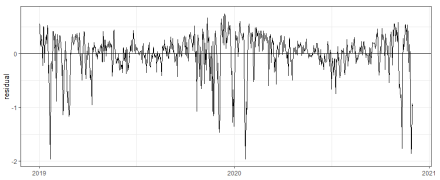
(b) Autocorrelation function (PM10)



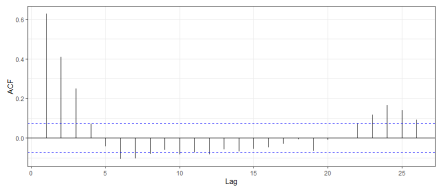
(c) Aggregated residuals (PM2.5)



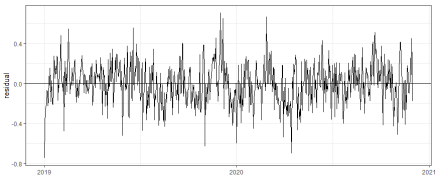
(d) Autocorrelation function (PM2.5)



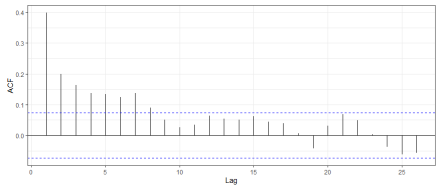
(e) Aggregated residuals (O3)



(f) Autocorrelation function (O3)



(g) Aggregated residuals (NO2)



(h) Autocorrelation function (NO2)

There is an evidence of autocorrelation at different lags for each pollutant. This indicates that a model for the aggregated residuals with (seasonal) auto-regressive or/and moving average terms can be considered.

O3 : The model overestimates the concentration levels, especially during the cold periods of the year. Indeed, very low residuals are observed for these periods. This could simply be explained by a high variability of the concentration levels at some period of time. Indeed, concentration levels can suddenly drop (see figure 5.2.4 during the winter period 2019-2020).

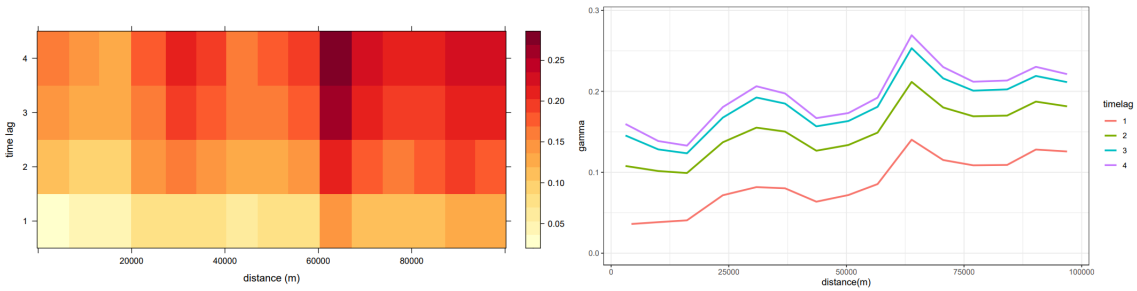
5.2.2.2 Spatio-temporal variogram

A useful indicator to use when there is a dependence both in time and space is the spatio-temporal variogram. Its formula is similar to the initial semivariogram (see formula 3.3.7) except that the time is taken into account. Let $Z(s,t)$ be a regionalized variable where $(s,t) \in S \times T$. The ST semivariogram can be expressed as follows :

$$\gamma(h,u) = \frac{1}{2} \times E[(Z(s,t) - Z(s+h,t+u))^2] \tag{5.2.24}$$

On a map, all pairs of points are separated by a specific distance. However, it is rare to obtain 2 different pairs of points separated by a same distance h. Hence, spatial lag intervals are considered where the points are separated by a distance that falls into a specific distance interval. This variogram is thus computed using time lags and spatial lag intervals.

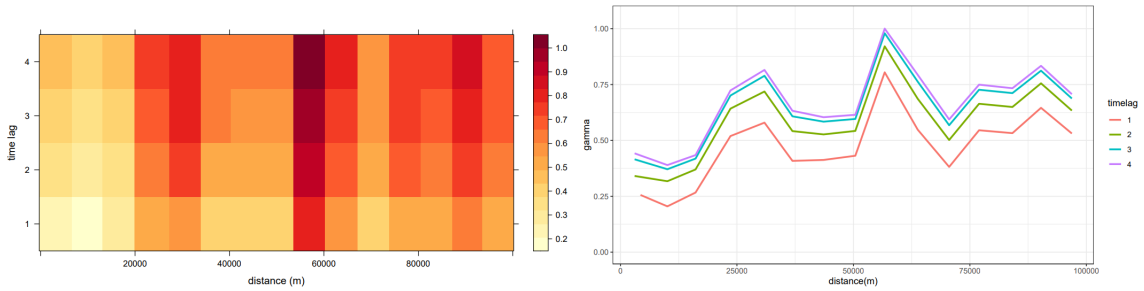
The ST variogram is applied to the regression residuals. Results are shown below.



(i) Empirical ST semivariogram (j) Marginal plot of the ST semivariogram

Figure 5.2.6: ST semivariogram of the regresssion residuals (PM10)

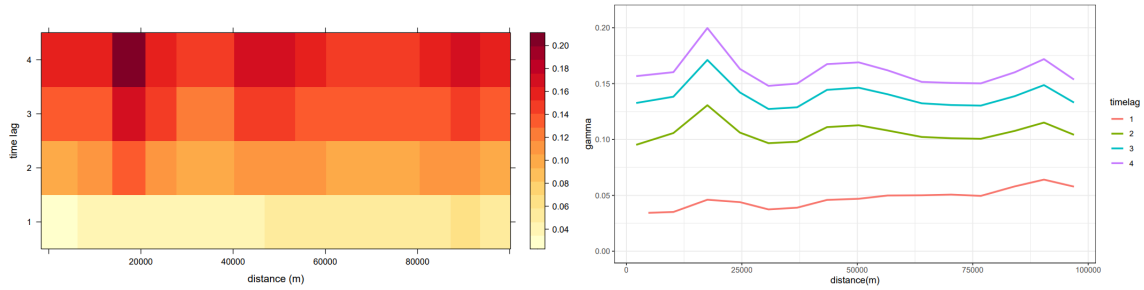
Regression residuals from PM10 shows autocorrelation both in space and time. As the distance and time increase between observations, the variability increases.



(a) Empirical ST semivariogram (b) Marginal plot of the ST semivariogram

Figure 5.2.7: ST semivariogram of the regresssion residuals (PM2.5)

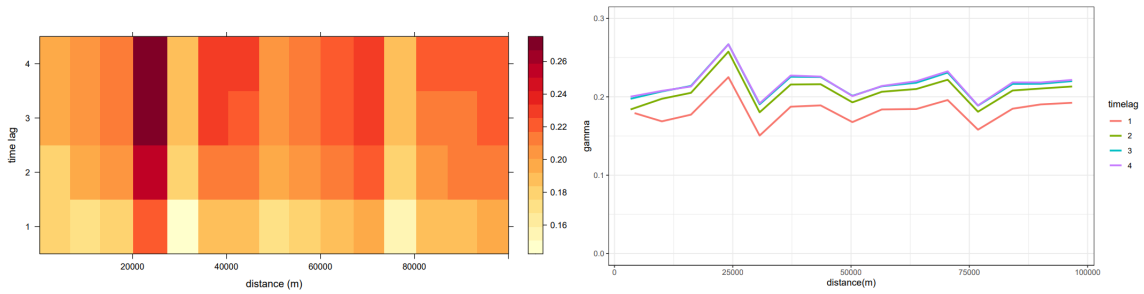
As for PM10, there is also autocorrelation both in space and time in the regression residuals. However, the difference of variability between time lags is less pronounced than for PM10.



(a) Empirical ST semivariogram (b) Marginal plot of the ST semivariogram

Figure 5.2.8: ST semivariogram of the regresssion residuals (O3)

Regression residuals from O3 shows a clear temporal autocorrelation. There is a strong difference in the variability between time lags. However, there is a low evidence of spatial autocorrelation. But the test of Moran shows that there is still a presence of spatial autocorrelation in the residuals for certain times (see table 5.2.9). This indicates that some spatial structures can be present for some days. But a regular spatial structure over time does not seem to be present.



(a) Empirical ST semivariogram (b) Marginal plot of the ST semivariogram

Figure 5.2.9: ST semivariogram of the regresssion residuals (NO2)

Regression residuals from NO2 shows a pure nugget effect for each time lag. That is, there is no evidence of spatial autocorrelation : the values of the ST semivariogram remain constant regardless of the spatial lag interval. The visualisation coincides with the previous results found with the statistical test of Moran (see table 5.2.9) : the spatial structure does not change that much for the 3 random days.

The difference of variability in the data between time lags is rather small. But there is still temporal autocorrelation (see figure 5.6(h)). Consequently, regression residuals could be modelled using the dependence in time only.

In the 3D plots, one can observe peaks of variability. This could be explained by extreme or high values in specific locations.

5.2.3 NO2 residuals modelling

We have seen previously that the NO2 residuals were autocorrelated only in time but not in space. Instead of modelling the residuals for each station, the average residuals are modeled. The type of model considered is an ARIMA model.

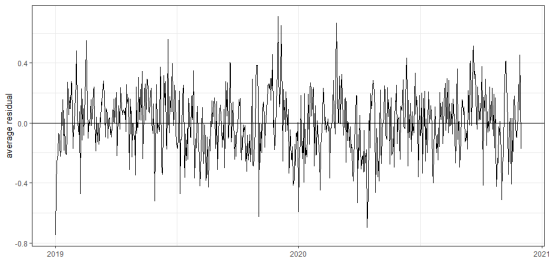


Figure 5.2.10: Average residuals per day

Two tests are used to assess stationarity of the time series. The augmented Dickey-Fuller test and the KPSS test. These two tests do not reject stationarity.

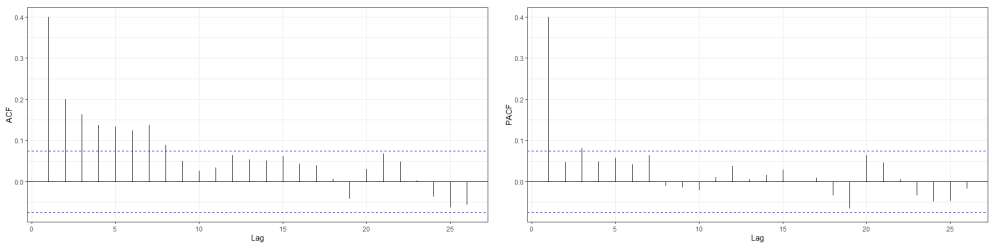


Figure 5.2.11: ACF (left) and PACF (right)

There are significant spikes from lag 1 to lag 8 in the ACF. The ACF values are decreasing exponentially. Moreover, there is a highly significant spike at lag 1 in the PACF. The two graphs show that an AR(1) model seems a good choice. The coefficient estimates are given below.

	Estimate	Std. Error	z value	Pr(>|z|)
ar1	0.41	0.03	11.7	< 2.2e-16 ***

Table 5.2.10: Coefficient estimates for an AR(1)

The AR1 coefficient estimate is significant. Another important step is to check if not serial correlation remains in the residuals of the model. The Ljung-Box diagnostic (as shown below) enables to detect such correlation.

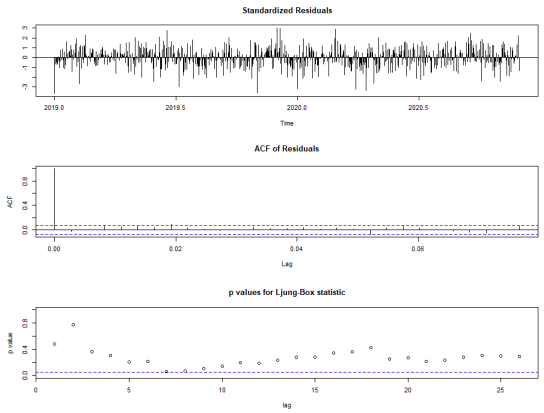


Figure 5.2.12: Ljung-Box diagnostic

The pvalues of the Ljung-Box statistic are all greater than 0.05. The model is therefore valid. Predictions are computed with the corresponding prediction interval.

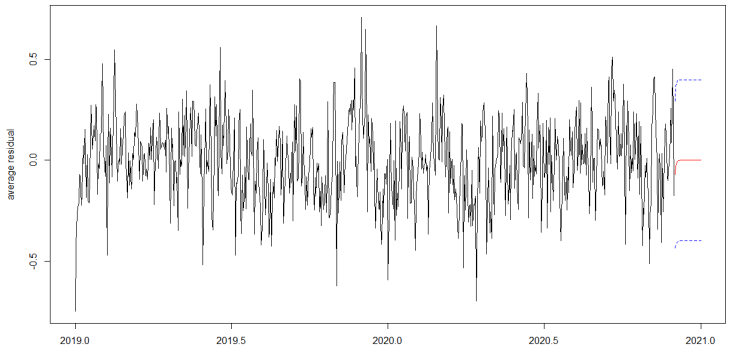


Figure 5.2.13: Predictions (red) with prediction interval (blue)

From a certain time, the predictions stabilize around 0. The dependency in time for the NO2 residuals is thus negligible. Indeed, the spatial autocorrelation has been removed by the deterministic component of the model. And it is not surprising that a dependence in time still remains in the residuals.

5.2.4 Space-time covariance functions

Explaining the whole spatio-temporal variability in the data using only covariates is very hard in practice. If the spatio-temporal structure was known, a Generalized Least squares (GLS) model could have been performed. Indeed, GLS models accept dependence structure in the residuals. However, the covariance matrix is often unknown. Therefore, covariance functions are considered.

A natural question could come in mind when thinking about this model concept : Why should we care about the dependence in the errors since the OLS parameter estimates and predictions are unbiased ? If there is dependence, standard errors of the parameter estimates and prediction standard errors become unreliable (Wikle, C. K. et al., 2019). The standard errors are usually underestimated. Hence, the quality of the prediction cannot be well evaluated.

5.2.4.1 Properties

Stationarity

Space-time covariance functions can be characterized by different properties. The one is the form of the stationarity. The space-time process can be *spatially stationary* : the dependence is only explained through the spatial lags. The space-time process can be *temporally stationary* : the dependence is explained only through the temporal lags. Finally, a space-time process can have a *stationary* covariance : the dependence is explained through spatial and temporal lags (form of stationarity chosen for the model).

Full symmetry

A ST covariance function can be fully symmetric. A *stationary* space-time process is fully symmetric if the ST covariance can be expressed as follows (Gneiting et al., 2006) :

$$C_{st}(h,u) = C_{st}(-h,u) = C_{st}(h,-u) = C_{st}(-h,-u) \tag{5.2.25}$$

A fully symmetric model is not able to detect a difference when the time goes backward or forward. Moreover, the use of a fully symmetric ST covariance function could be inappropriate in practice (Wikle et al., 2019). Indeed, would it be reasonable to consider that the covariance between today's temperature in Lisbon and yesterday's temperature in Paris is the same as that between yesterday's temperature in Lisbon and today's temperature in Paris ? Some meteorologists wouldn't accept such assumption. One can imagine a weather process that goes from the North East to the South west. In this case, the direction would account as a factor of variability.

Separable

A ST covariance function is *separable* when the impact of time and space is dissociable. Separable models could be expressed as follows.

$$C_{st}(h,u) = C_s(h)C_t(u) \tag{5.2.26}$$

$C_{st}(.)$: ST Covariance defined over$\mathbb{R}^3$
$C_s(.)$: Covariance defined over $\mathbb{R}^2$
$C_t(.)$: Covariance defined over $\mathbb{R}^1$
h : space lag
u : time lag

For two fixed spatial lag h and h', the relationship between $C_{st}(h,u)$ and $C_{st}(h',u)$ is proportional. The disadvantage of this type of model is that the model is not able to explain complex interactions between space and time.

An illustration with the different relationships between properties is given below.

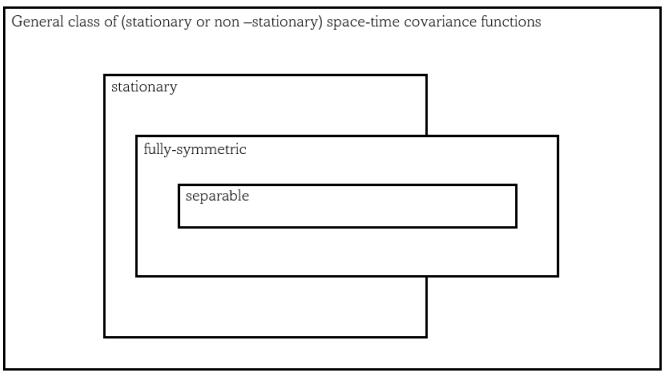


Figure 5.2.14: Relationships between properties

Separable covariance models are necessarily fully symmetric. But fully symmetric covariance models are not necessarily separable.

5.2.4.2 Description

Different models exist : *separable*, *metric*, *sum-metric*, *product sum* etc. The metric, sum-metric and product sum models are chosen because the separable model is not able to detect complex interactions between space and time. These covariance functions must have an essential common property : they must be positive definite. This property guarantees that the kriging variances (prediction variances) are positive. A general formula of the covariance can be written as follows (Wikle et al., 2019).

$$c(s, s'; t, t') \equiv Cov(Z(s, t), Z(s', t')) \tag{5.2.27}$$

- **Metric**

  The Metric model proposes one variogram for both time and space (joint variogram). Since time is not in the same dimension as space, time is rescaled so that it fits to the spatial dimension.

$$C_{\text{st}}(h, u) = C_{\text{joint}}\left(\sqrt{h^2 + (\kappa \cdot u)^2}\right) \qquad \kappa > 0 \tag{5.2.28}$$

  $C_{joint}(.)$: Covariance defined over $\mathbb{R}^3$
  κ : Spatio-temporal anisotropy parameter or scaling parameter. It is expressed as spatial unit per time unit (e.g 18km/day). This parameter allows to match spatial and time units.
  h : space lag
  u : time lag

  The corresponding spatio-temporal variogram formula is given below.

$$\gamma_{\text{st}}(h, u) = \gamma_{\text{joint}}\left(\sqrt{h^2 + (\kappa \cdot u)^2}\right) \tag{5.2.29}$$

  γ_{joint} : A Joint variogram for time and space together.

- **Sum-metric**

$$C_{\text{st}}(h, u) = C_{\text{s}}(h) + C_{\text{t}}(u) + C_{\text{joint}}\left(\sqrt{h^2 + (\kappa \cdot u)^2}\right) \kappa > 0 \tag{5.2.30}$$

  The corresponding spatio-temporal variogram formula is given below.

$$\gamma_{\text{st}}(h, u) = \gamma_{\text{s}}(h) + \gamma_{\text{t}}(u) + \gamma_{\text{joint}}\left(\sqrt{h^2 + (\kappa \cdot u)^2}\right) \tag{5.2.31}$$

$\gamma_s(h)$: Spatial variogram.
$\gamma_t(u)$: Temporal variogram.
γ_{joint} : Joint variogram

- **Product sum**

  The product sum model is a fully symmetric and non separable function.

$$C_{\text{st}}(h,u) = kC_{\text{s}}(h)C_{\text{t}}(u) + C_{\text{s}}(h) + C_{\text{t}}(u) \quad k > 0 \tag{5.2.32}$$

  The corresponding spatio-temporal variogram formula is given below.

$$\gamma_{\text{st}}(h,u) = (k \cdot \text{sill}_{\text{t}} + 1)\,\gamma_{\text{s}}(h) + (k \cdot \text{sill}_{\text{s}} + 1)\,\gamma_{\text{t}}(u) - k\gamma_{\text{s}}(h)\gamma_{\text{t}}(u) \tag{5.2.33}$$

$\gamma_s(h)$: Spatial variogram.
$\gamma_t(u)$: Temporal variogram.
sill_{s} : Sill of the spatial variogram
sill_{t} : Sill of the temporal variogram

5.2.4.3 Model fitting

Models are fitted using the package *gstat* in R. The function that creates the model is *vgmST*. This function takes initial parameters. The function that fits the model is *fit.StVariogram*. This function takes the empirical ST semivariogram and the chosen model as parameters. Final model parameters are found by a convergence procedure from the initial parameters. The choice of good initial parameters (in *vgmST*) is thus mandatory to obtain good final parameters. Multiple set of values are tested and the MSE is computed to check the improvement of the model.

An argument of the function *fit.StVariogram* called *fit.method* is available. This argument allow the user to associate a weight to the squared residuals in the least squares estimation. The default method is chosen (no weights). An example of code is shown below.

```
metricVgm <- vgmST(stModel = "metric", joint = vgm(0.20, "Exp", 60000, nugget =
0.05), sill = 0.20, stAni = 30000, temporalUnit = "days")
metricVgm <- fit.StVariogram(varST, metricVgm)
```

The fitted models are shown with the empirical ST semivariogram. Moreover, the difference between the sample and the fitted values are plotted. The plots of the differences give a first idea of the performance of each model. Finally, the mean squared error is calculated for each model. All the results are shown below for each pollutant.

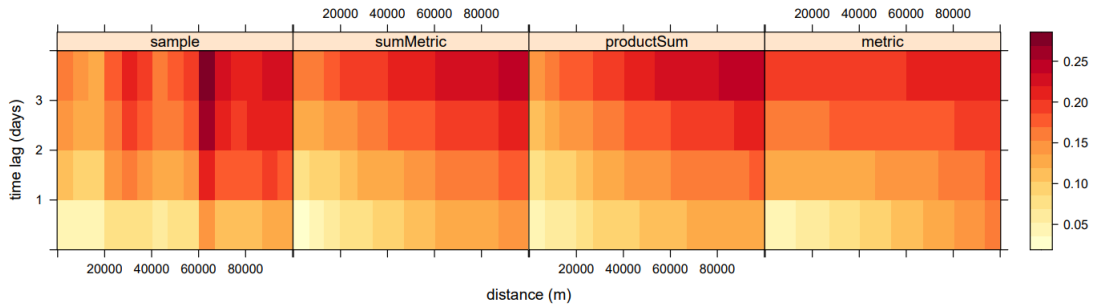


Figure 5.2.15: Sample against fitted (PM10)

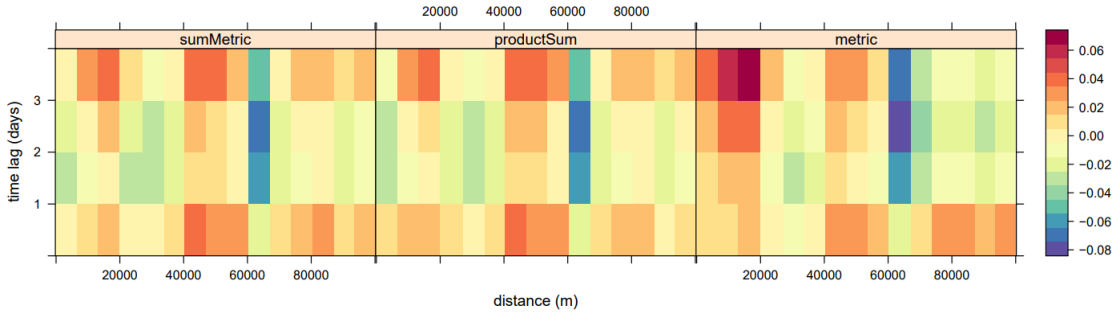


Figure 5.2.16: Difference between sample and fitted (PM10)

model	sum metric	product sum	metric
MSE	0.00052	0.00053	0.00075

Table 5.2.11: Performance of each model (PM10)

The model retained is the sum metric model since it has the lower MSE.

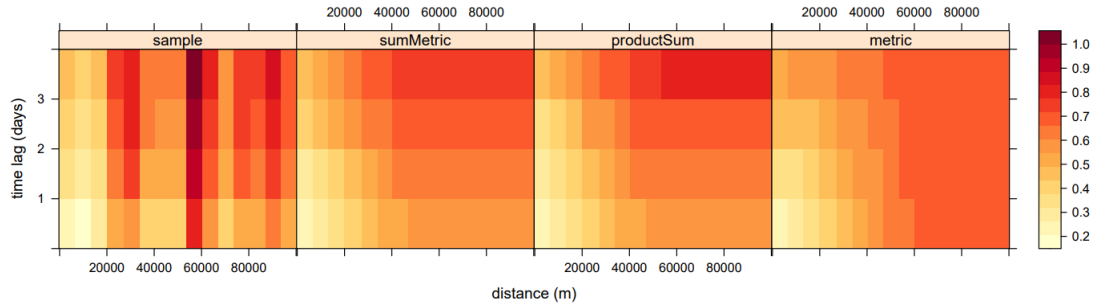


Figure 5.2.17: Sample vs fitted (PM2.5)

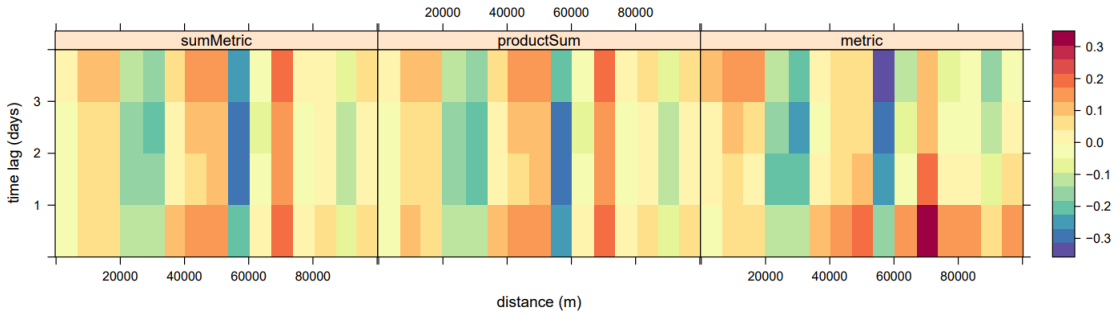


Figure 5.2.18: Difference between sample and fitted (PM2.5)

model	sum metric	product sum	metric
MSE	0.01286	0.01292	0.017

Table 5.2.12: Performance of each model (PM2.5)

The model retained is the sum metric model again.

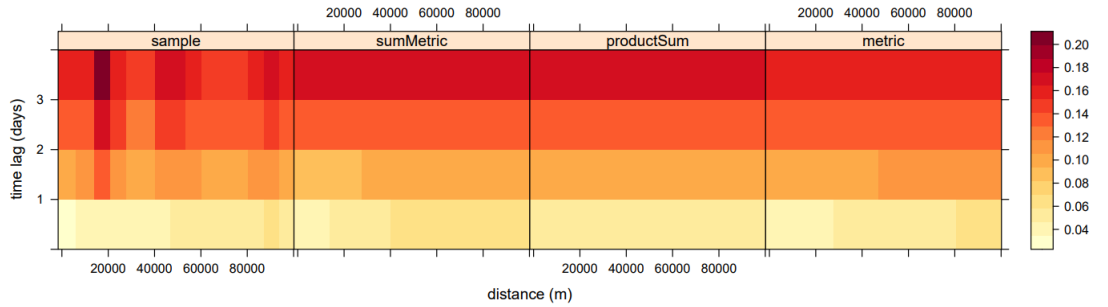


Figure 5.2.19: Sample vs fitted (O3)

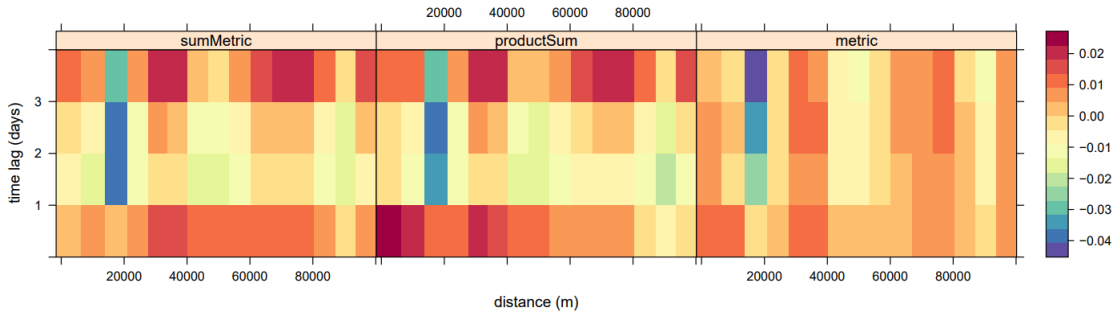


Figure 5.2.20: Difference between sample and fitted (O3)

model	sum metric	product sum	metric
MSE	0.00018	0.00019	0.0000995

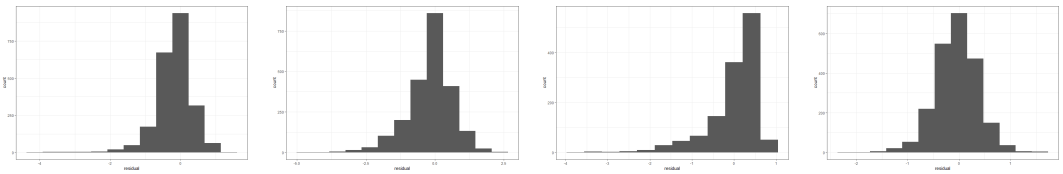
Table 5.2.13: Performance of each model (O3)

The model retained is the metric model.

5.2.5 Prediction

5.2.5.1 Prediction residuals

Predictions are computed using the testset (data from 1st to 31st of December 2020). The model is tested with data that were not used to create the model. The histograms of the prediction residuals are plotted below. Root mean squared errors and the average of the residuals (average bias) are also computed. This gives a first idea of the performance of each regression model. Recall that the space-time dependence is not taken into account yet.



(a) PM10 (b) PM2.5 (c) O3 (d) NO2

Figure 5.2.21: Prediction residuals

	PM10	PM2.5	O3	NO2
RMSE	0.57	0.83	0.63	0.42
Average bias	-0.18	-0.18	0.01	-0.05

Table 5.2.14: Prediction performance of each pollutant model

Except for O3, RMSE from predictions are higher than those obtained from the training set (see table 5.2.7). Each model predicts from new observed data (out of sample). RMSE are thus expected to be higher. The bias is greater for the pollutant PM10 and PM2.5. This could be explained by the presence of a strong spatio-temporal dependency.

Prediction residuals can be also analyzed over time as shown below. A vertical line at 0 is displayed on each histogram.

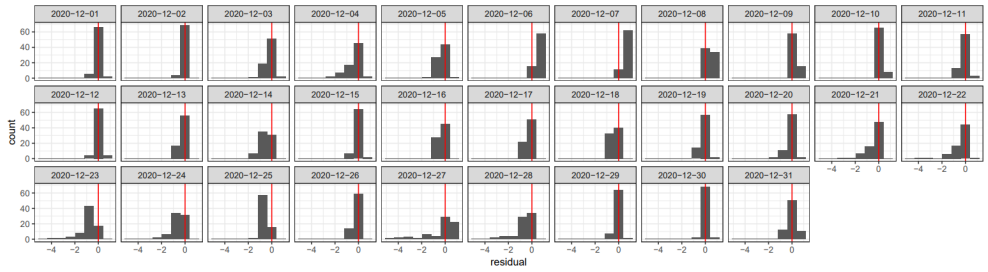


Figure 5.2.22: Prediction residuals (PM10)

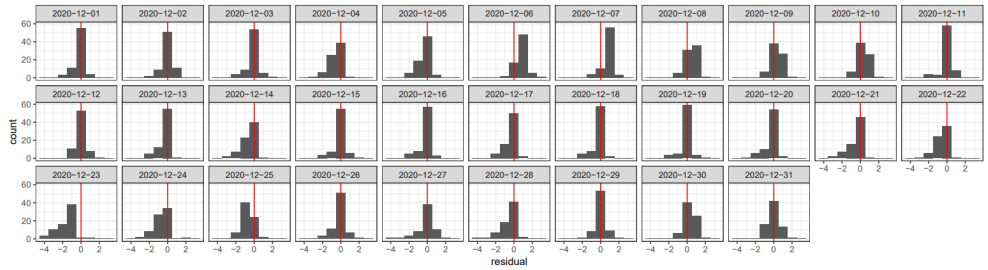


Figure 5.2.23: Prediction residuals (PM2.5)

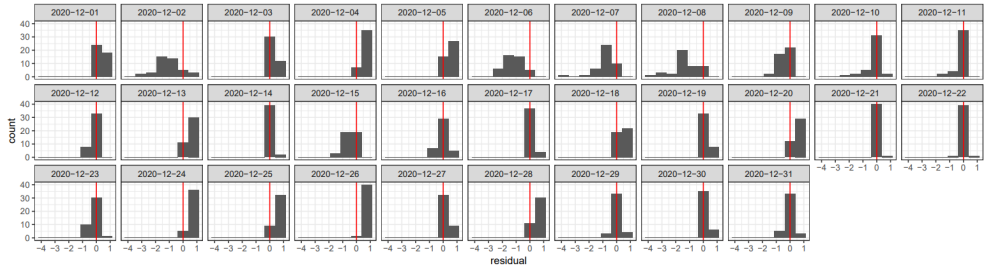


Figure 5.2.24: Prediction residuals (O3)

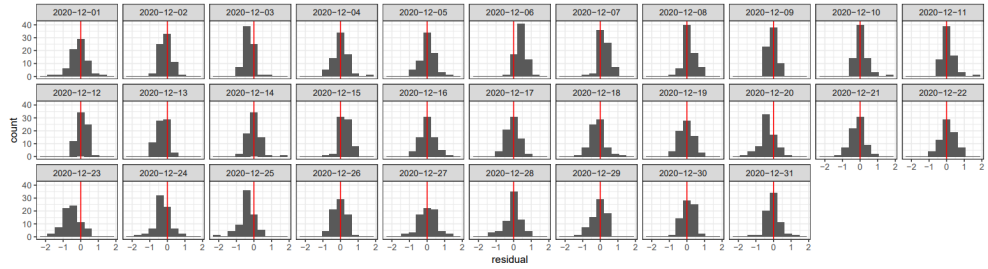
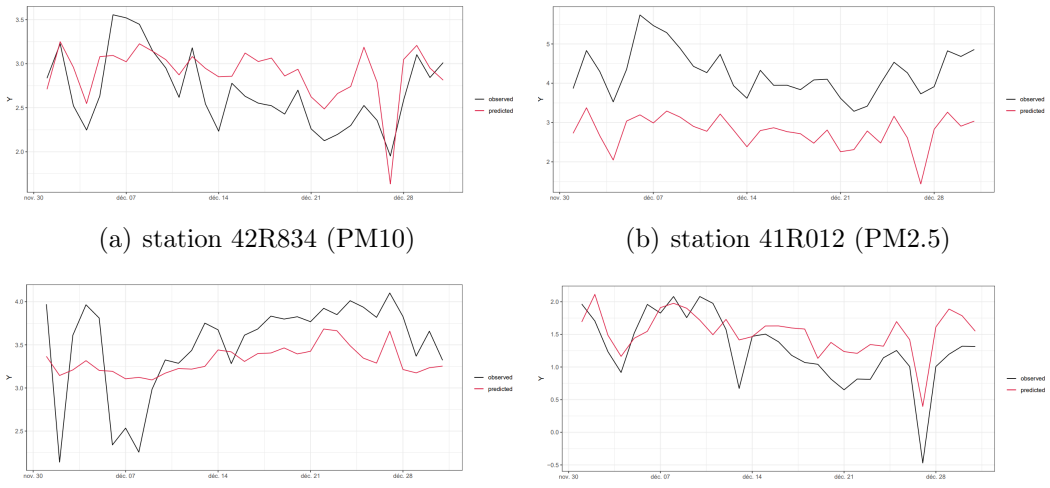


Figure 5.2.25: Prediction residuals (NO2)

Among the 4 pollutants, NO2 is the one that shows the better performance over time. Indeed, residuals from NO2 are distributed more homogeneously around 0 than for the

other pollutants. For some days, there is a strong bias, especially for the pollutant O3. The model struggles to predict specific days. One wants to remove the observed bias by modelling the residuals through a space-time covariance model.

Predictions can be also analyzed for a specific station. The observed and predicted values are shown below.



(a) station 42R834 (PM10)

(b) station 41R012 (PM2.5)

(c) station 42N040 (O3)

(d) station 43N121 (NO2)

Figure 5.2.26: Observed against predicted for 1 randomly chosen station

One can observe that for the station 41R012 (PM2.5), there is a strong bias : although the structure over time is well captured by the model, the concentrations levels are underestimated.

5.2.5.2 Spatio-temporal kriging of the prediction residuals

A spatio-temporal kriging can now be applied to the prediction residuals. The ST kriging requires the use of a dataset that has a regular structure (i.e same number of stations for each time). Hence, stations that do not have observed values for the whole month (December 2020) are ignored. 66 stations are used for PM10, 60 for PM2.5, and 40 for O3.

The predictor of interest is the one that minimizes :

$$E[(Z(s_0,t_0) - \hat{Z}(s_0,t_0))^2] \quad (s_0,t_0) \in S \times T \tag{5.2.34}$$

The kriging predictor is the best linear unbiased estimator for $Z(s_0,t_0)$. Its formula is given below.

$$Z^*(s_0,t_0) = C'_{ST,0} C_{ST}^{-1} \hat{Y} \tag{5.2.35}$$

C_{ST} : ST covariance matrix of size n $\times$ n. Each element of the matrix can be written as :

$$C_{ST}(i,j) = C_{ST}(s_i - s_j, t_i - t_j) \tag{5.2.36}$$

Where $s_i - s_j$ and $t_i - t_j$ correspond respectively to spatial and temporal lags.
$C_{ST,0}$: vector of size n. Each element of the vector can be written as :

$$C_{ST,0}(i) = C_{ST}(s_i - s_0, t_i - t_0) \tag{5.2.37}$$

$\hat{Y}$: vector of size n. Each element can be expressed as :

$$\hat{Y}(s_i, t_i) = Z(s_i, t_i) - \hat{\mu}(s_i, t_i) \tag{5.2.38}$$

Its prediction variance formula is expressed as follows.

$$\sigma_*^2(s_0, t_0) = V[Z^*(s_0, t_0) - Z(s_0, t_0)] = C_{ST}(0,0) - C'_{ST,0} C_{ST}^{-1} C_{ST,0} \tag{5.2.39}$$

The corresponding standard error is $\sigma_*(s_0, t_0)$. This metric has the same unit as $\hat{Z}(s_0, t_0)$. As for the original kriging (spatial only), the ST kriging predictor is an exact interpolator. That is, values estimated at the sample locations are equal to the observed values. Residuals are estimated over the whole spatial domain. The spatial grid represents a set of pixels with equal dimensions. And there is one prediction for each pixel (the center of the square). The center of the pixel is the point where the prediction is performed. The prediction is associated to a point in the space and not to a surface or a volume. Predictions over the whole spatial domain are shown below.

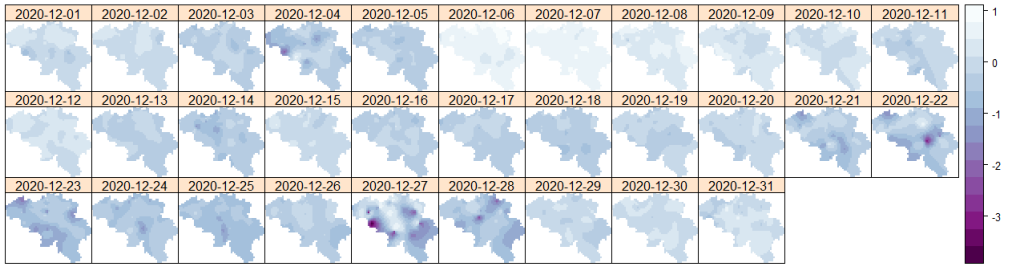


Figure 5.2.27: ST kriging of the prediction residuals (PM10)

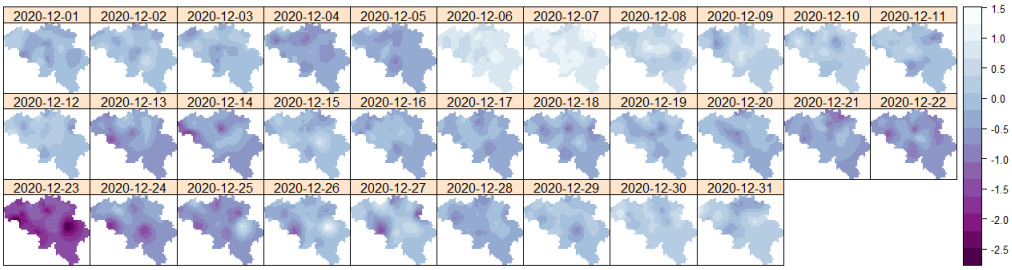


Figure 5.2.28: ST kriging of the prediction residuals (PM2.5)

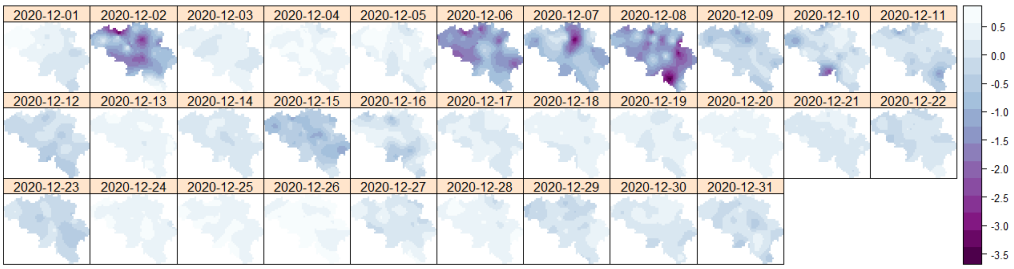


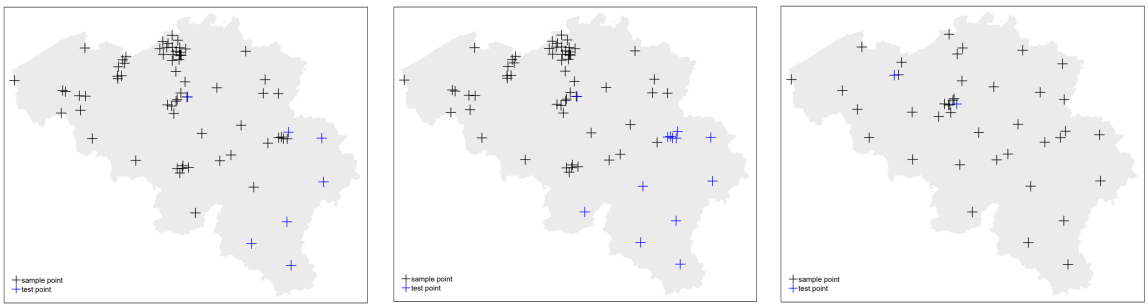
Figure 5.2.29: ST kriging of the prediction residuals (O3)

The performance of the complete model can now be analyzed by comparing observed values and final predicted values (regression + ST kriging). That is, one can study the following component of the model.

$$e = Y(s,t) - \left(\hat{Y}(s,t) + \hat{Z}(s,t)\right) \tag{5.2.40}$$

$\hat{Y}(s,t)$: fitted value obtained from the regression.
$\hat{Z}(s,t)$: estimated residual obtained via ST kriging.

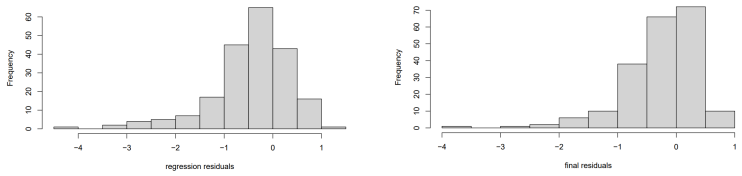
The centers of the pixels or squares usually do not correspond to the exact sample locations. However, there are sample locations that were not used in the ST Kriging. Thus, one can use them to test the performance of the ST kriging. Pixels are relatively small. Therefore, the accuracy level is rather good. One can use the estimate of the pixel that is closest to the sample location. The Euclidean distance is computed between the unused sample location and the grid points. The estimate of the grid point that has the minimum distance with the sample location is taken. Even though it does not correspond to the prediction at the true location, this will give an idea of the performance. 7 stations for PM10, 13 for PM2.5 and 2 for O3 are used. The stations used to test the performance of the ST kriging are represented below in blue. The black crosses are the ones used for the prediction.



(a) PM10 (b) PM2.5 (c) O3

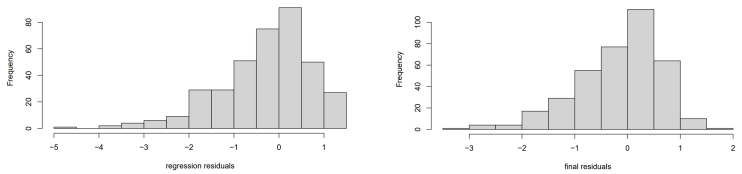
Figure 5.2.30: Test points and sample points

Regression predictions and final prediction obtained via ST kriging are compared with the observed value (see Appendix). 2 residuals can now be compared : residuals obtained via the ST regression kriging (final residuals) and residuals obtained via the regression only (regression residuals).



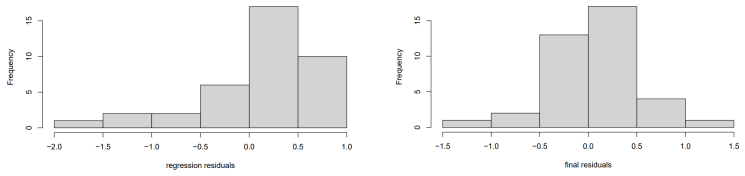
(a) Regression residuals (b) Final residuals

Figure 5.2.31: PM10



(a) Regression residuals (b) Final residuals

Figure 5.2.32: PM2.5



(a) Regression residuals (b) Final residuals

Figure 5.2.33: O3

The bias seems reduced for each pollutant. Final residuals are more distributed around 0. The predictions of each method are investigated over time.

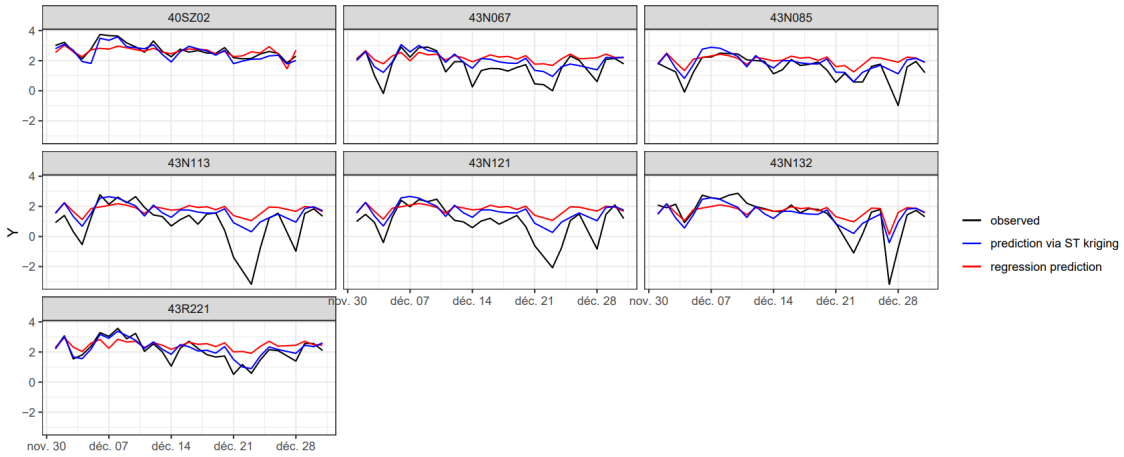


Figure 5.2.34: PM10

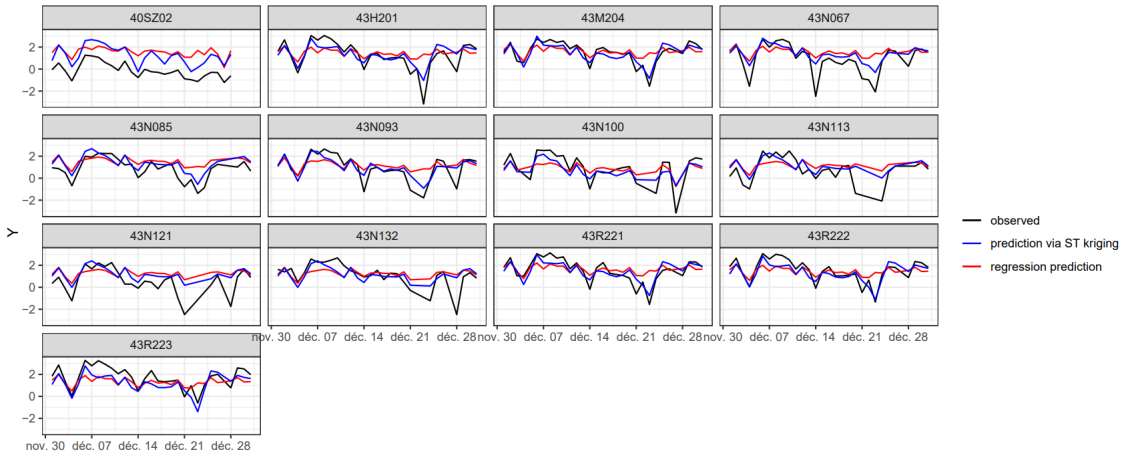


Figure 5.2.35: PM2.5

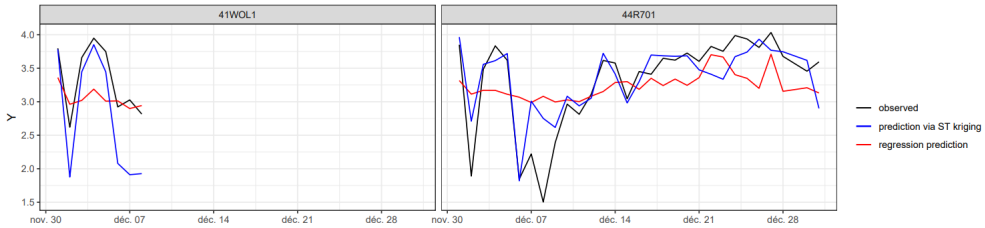


Figure 5.2.36: O3

ST kriging takes more into account fluctuations over time as expected. However, strong bias can remain (e.g PM2.5 : station 40SZ02). Finally, Root mean squared errors are computed to compare both performances.

	PM10	PM2.5	O3
RMSE (regression)	0.94	1.07	0.57
RMSE (ST regression kriging)	0.69	0.81	0.45

Table 5.2.15: Performance of the spatio-temporal kriging

ST kriging provides better predictions than regression predictions.

Conclusion

Working on these data and this topic has been a real challenge for me. The most difficult part was to find the modelling procedure that enables to explain the spatio-temporal variability in the data. Many methods were proposed in the literature. I had to make choices and be able to justify them. Some tasks were particularly tough such as : managing large databases, saving memory usage, code optimisation and time management. This work was a valuable experience and helped me to develop theoretical and practical skills.

Spatio-temporal statistics is a very interesting field as it combines geostatistics and advanced statistical methods. The method used to model the data was one of many. Spatio-temporal clustering, dynamic spatio-temporal models, basis functions are other examples of methods that can be used to model spatio-temporal data. Many relevant books are available in the literature.

Spatio-temporal regression kriging has shown its value through this study by reducing the prediction errors. This method has the advantage to separate the modelling procedure in two parts (regression and ST kriging) and to consider more variability sources in the model. Spatio-temporal interpolation has the potential to provide better predictions than spatial interpolation (Gräler et al., 2016). However, modelling the spatio-temporal variability is much more complex and difficult.

Bibliography

[1] Allard, D. (2016) Géostatistique spatio-temporelle, Atelier Statistique: Introduction aux méthodes spatiales.

[2] Bivand, R., Pebesma, E.J., Gomez-Rubio, V. Applied Spatial Data Analysis with R, Springer, New York, 2nd edition 2013.

[3] European Environment Agency : Europe's air quality status 2022, April 2022 from https://www.eea.europa.eu/publications/status-of-air-quality-in-Europe-2022

[4] Gneiting, T., Genton, M. G., and Guttorp, P. (2006) Geostatistical space-time models, stationarity, separability, and full symmetry. Technical Report no 475.

[5] Gräler, B., Pebesma, E., Heuvelink, G. (2016) The R Journal, 204-218.

[6] Loonis, V. (2018) Manuel d'analyse spatiale : Théorie et mise en oeuvre pratique avec R, Insee méthodes n°131.

[7] RESSTE-Network, Geniaux, G. (Co-dernier auteur) (2017) Analyzing spatio-temporal data with R : Everything you always wanted to know - but were afraid to ask. Journal de la Société Française de Statistique, 158 (3), 124-158.

[8] Spiegelhalter , D. (2019) The Art of Statistics Learning from Data, Pelican Book.

[9] Thunberg, G. (2022) The Climate Book, Kero.

[10] Wikle, C. K., Zammit-Mangion, A., and Cressie, N. (2019), Spatio-Temporal Statistics with R, Boca Raton, FL: Chapman & Hall/CRC. © 2019 Wikle, Zammit-Mangion, Cressie.

[11] WALLER, L.A., Carol Anne, G.C. (2004) Applied spatial statistics for public health data. T. 368. John Wiley & Sons.

Appendix A

Complementary analysis

A.1 Chapter 5

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	2.702e+00	9.888e-02	27.321	< 2e-16 ***
t	-9.325e-05	1.196e-05	-7.794	6.60e-15 ***
X1	2.316e-01	3.862e-03	59.980	< 2e-16 ***
X2	3.444e-01	5.842e-03	58.959	< 2e-16 ***
week_timeweekend	-6.480e-02	4.878e-03	-13.283	< 2e-16 ***
vegetation	-2.523e-03	1.545e-04	-16.330	< 2e-16 ***
denspop	-1.468e-05	1.159e-06	-12.673	< 2e-16 ***
airport_closest	-9.853e-07	1.881e-07	-5.238	1.63e-07 ***
LPS_closest	-1.493e-05	6.645e-07	-22.471	< 2e-16 ***
city_closest	1.694e-06	2.108e-07	8.036	9.51e-16 ***
x	-2.438e-06	5.141e-08	-47.424	< 2e-16 ***
y	3.347e-06	8.010e-08	41.790	< 2e-16 ***
temperature	2.662e-02	6.845e-04	38.889	< 2e-16 ***
precipitation	-1.837e-02	5.179e-04	-35.468	< 2e-16 ***
pressure	3.573e-02	2.893e-03	12.350	< 2e-16 ***
windspeed	-4.371e-02	3.472e-04	-125.899	< 2e-16 ***
humidity	-7.742e-03	2.777e-04	-27.882	< 2e-16 ***

Table A.1.1: Final model (PM10)

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-1.011e+00	1.831e-01	-5.524	3.34e-08 ***
t	-1.816e-04	2.051e-05	-8.854	< 2e-16 ***
X1	3.433e-01	6.631e-03	51.776	< 2e-16 ***
X2	5.449e-01	1.004e-02	54.256	< 2e-16 ***
week_timeweekend	4.353e-02	8.390e-03	5.188	2.13e-07 ***
vegetation	1.440e-03	3.054e-04	4.715	2.43e-06 ***
denspop	8.924e-05	3.442e-06	25.928	< 2e-16 ***
regionBrussels	1.044e+00	2.909e-02	35.873	< 2e-16 ***
regionFlanders	2.421e-01	2.415e-02	10.027	< 2e-16 ***
airport_closest	1.379e-05	3.244e-07	42.517	< 2e-16 ***
LPS_closest	-1.925e-05	1.282e-06	-15.017	< 2e-16 ***
city_closest	-2.641e-06	3.902e-07	-6.768	1.32e-11 ***
x	-9.573e-07	9.295e-08	-10.299	< 2e-16 ***
y	8.822e-06	2.984e-07	29.564	< 2e-16 ***
temperature	2.428e-02	1.173e-03	20.695	< 2e-16 ***
pressure	6.547e-02	5.180e-03	12.640	< 2e-16 ***
windspeed	-7.080e-02	5.675e-04	-124.759	< 2e-16 ***
humidity	-2.572e-03	4.499e-04	-5.716	1.10e-08 ***

Table A.1.2: Final model (PM2.5)

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	4.010e+00	4.562e-02	87.908	< 2e-16 ***
t	8.883e-05	1.438e-05	6.177	6.62e-10 ***
X1	2.683e-01	4.631e-03	57.947	< 2e-16 ***
X2	-6.464e-02	6.884e-03	-9.389	< 2e-16 ***
week_timeweekend	4.259e-02	5.870e-03	7.255	4.12e-13 ***
vegetation	3.935e-03	1.933e-04	20.352	< 2e-16 ***
regionBrussels	4.353e-02	1.137e-02	3.827	0.00013 ***
regionFlanders	-2.981e-02	1.216e-02	-2.451	0.01423 *
airport_closest	1.727e-06	2.307e-07	7.484	7.40e-14 ***
LPS_closest	1.069e-05	7.903e-07	13.522	< 2e-16 ***
city_closest	-1.247e-06	2.460e-07	-5.070	4.00e-07 ***
x	4.983e-07	5.486e-08	9.084	< 2e-16 ***
y	-8.010e-07	1.640e-07	-4.886	1.04e-06 ***
temperature	3.511e-02	8.208e-04	42.776	< 2e-16 ***
humidity	-1.348e-02	3.392e-04	-39.753	< 2e-16 ***
precipitation	2.085e-02	5.599e-04	37.233	< 2e-16 ***

Table A.1.3: Final model (O3)

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	4.610e+00	4.220e-02	109.255	< 2e-16 ***
t	-4.732e-04	1.185e-05	-39.917	< 2e-16 ***
X1	3.975e-02	3.811e-03	10.430	< 2e-16 ***
X2	4.165e-01	5.778e-03	72.088	< 2e-16 ***
week_timeweekend	-2.694e-01	4.830e-03	-55.783	< 2e-16 ***
vegetation	-6.318e-03	1.586e-04	-39.840	< 2e-16 ***
denspop	2.948e-05	1.513e-06	19.480	< 2e-16 ***
regionBrussels	-4.710e-02	1.413e-02	-3.334	0.000857 ***
regionFlanders	2.143e-02	1.185e-02	1.808	0.070545 .
airport_closest	-3.352e-06	1.946e-07	-17.222	< 2e-16 ***
LPS_closest	-2.398e-05	7.657e-07	-31.322	< 2e-16 ***
city_closest	-3.074e-06	2.276e-07	-13.510	< 2e-16 ***
x	-1.626e-06	6.007e-08	-27.072	< 2e-16 ***
y	1.689e-06	1.601e-07	10.552	< 2e-16 ***
temperature	3.152e-03	6.797e-04	4.637	3.55e-06 ***
precipitation	4.079e-03	5.045e-04	8.085	6.35e-16 ***
windspeed	-4.964e-02	3.498e-04	-141.918	< 2e-16 ***
humidity	-6.461e-03	2.751e-04	-23.484	< 2e-16 ***

Table A.1.4: Final model (NO2)

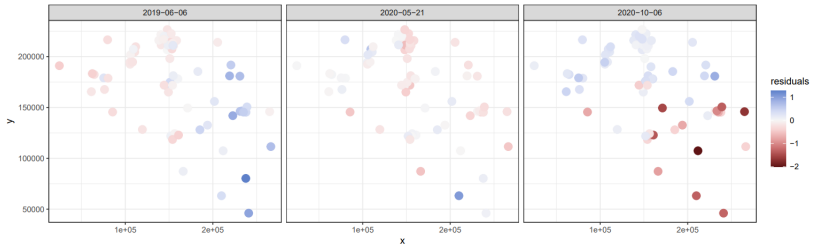


Figure A.1.1: Residuals for 3 randomly chosen dates (PM10)

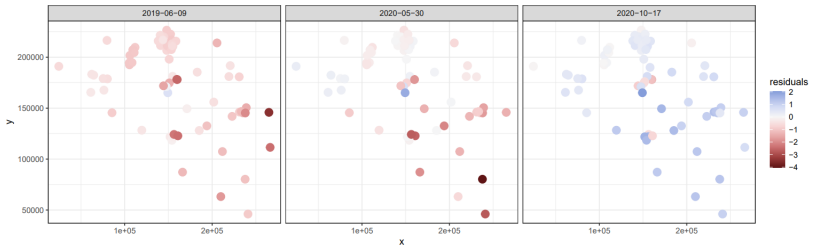


Figure A.1.2: Residuals for 3 randomly chosen dates (PM2.5)

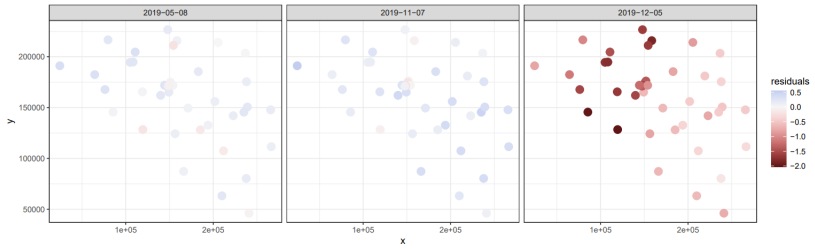


Figure A.1.3: Residuals for 3 randomly chosen dates (O3)

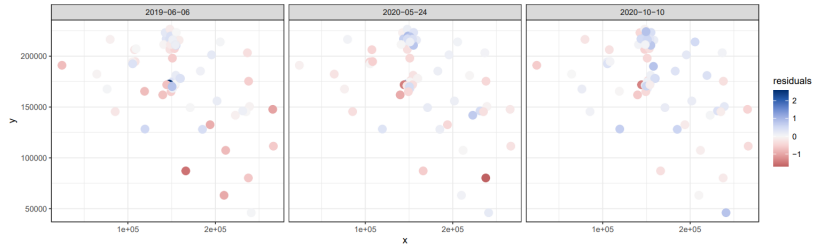
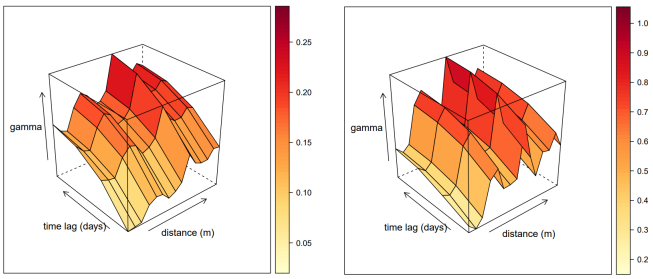
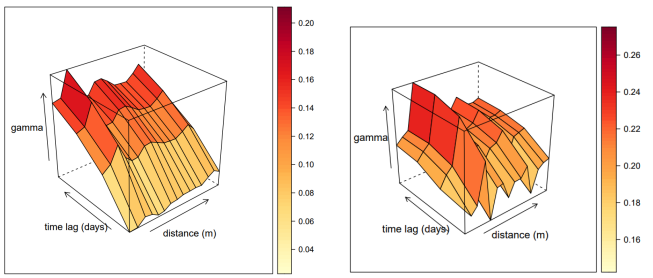


Figure A.1.4: Residuals for 3 randomly chosen dates (NO2)



(a) PM10

(b) PM2.5



(c) O3

(d) NO2

Figure A.1.5: 3D representation of the ST semivariogram

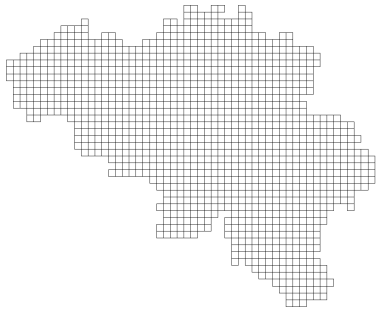


Figure A.1.6: Prediction grid used for spatio temporal kriging

A.2 Chapter 2

Figure A.2.1: Large point sources (LPS)

Appendix B

Computer code

```
########## FUNCTIONS used for the Rmd file##########
#author : Lucas Bouché

#functions are given in the order of the memoire.Rmd file

#function 1 : compute minimum distance from a station to a LPS or city or airport (euclidean distance used)
min_dist <- function(data, data2, type){

  X_tgt <- subset(data2, TYPE == type)
  vec <- c()
  for (i in 1:nrow(data@data)){
    dists <- sqrt( (data@coords[i,1] - X_tgt@coords[,1])^2 + (data@coords[i,2] - X_tgt@coords[,2])^2)
    vec <- append(vec, min(dists))
  }
  return(vec)
}
#function 2 : change the type of the variables of the dataset
update <- function(dataframe){
  dataframe$station <- factor(dataframe$station)
  dataframe$value <- as.numeric(dataframe$value)
  dataframe$date <- as.character(dataframe$date)
  dataframe$year <- substr(dataframe$date,1, 4)
  dataframe$month <- substr(dataframe$date,5, 6)
  dataframe$day <- substr(dataframe$date,7, 8)
  dataframe$hour <- as.factor(dataframe$hour)
  dataframe$date_time <- strptime(paste(dataframe$date, dataframe$hour, sep = ""), format = "%Y%m%d%H:%M:%S")
  return(dataframe)
}

#function 3 : sum across stations for each day until 31 december 2020
uni_plot <- function(polluant){
  sample <- aggregate(value ~ date, data = get(polluant), FUN = "sum")
  sample <- sample[order(sample$date),]
  plot <- ggplot(sample) + aes( x = as.Date(date, format = "%Y%m%d"), y = value, group = 1) +
    geom_line() + labs( x = "", y = lab1)
  return(plot)
}
#function 4 : plot the mean for each hour
hourly_plot <- function(polluant){

  sample <- aggregate(formula = value ~ hour, data = get(polluant), FUN = "mean")
  plot <- ggplot(sample) + aes( x = substr(hour, 1, 2), y = value, group = 1) +
    stat_summary(geom ="line", fun = "mean") +
    labs(y = lab1, title = "", x = "hour")

  return(plot)
```

```
}
#function 5 : plot the time series of the station for maximum values
extreme_plot <- function(stat_max, polluant){

  plot <- ggplot(subset(get(polluant), station == stat_max)) +
    aes ( x = as.POSIXct(date_time), y = value, group = 1) + geom_line() +
    scale_x_datetime( date_labels =  "%b %Y") +
    labs( x = "", y = lab1, title = paste(polluant, ":", stat_max))
  return(plot)
}
#function 6 : change type of variables
update_2 <- function(dataframe){
  dataframe$region <- factor(dataframe$region)
  dataframe$year <- substr(dataframe$date,1, 4)
  dataframe$year <- factor(dataframe$year)
  dataframe$month <- substr(dataframe$date,5, 6)
  dataframe$month <- factor(dataframe$month)
  dataframe$day <- substr(dataframe$date,7, 8)
  dataframe$day <- factor(dataframe$day)
  return(dataframe)
}
#function 8 : convert a dataframe to a spatial dataframe
conv_spatial <- function(dataframe){
  coordinates(dataframe) <- ~ x + y
  dataframe@proj4string <- CRS(prjs[pos, "prj4"])
  return(dataframe)
}
#function 9 : update time variables
update_3 <- function(sp_obj){

  sp_obj@data$date <- as.Date(sp_obj@data$date, format = "%Y%m%d")
  sp_obj@data$wkd <- weekdays(sp_obj@data$date)
  sp_obj@data$week_time[(sp_obj@data$wkd != "Saturday") & (sp_obj@data$wkd != "Sunday")] <-
"week"
  sp_obj@data$week_time[sp_obj@data$wkd %in% c("Saturday", "Sunday")] <- "weekend"
  sp_obj@data$year <- format(sp_obj@data$date, format = "%Y")
  sp_obj@data$month <- format(sp_obj@data$date, format = "%m")
  sp_obj@data$day <- format(sp_obj@data$date, format = "%d")
  return(sp_obj)
}
#function 10 : return plot of covid
covid_plot <- function(polluant){
  plot <- ggplot(get(polluant)@data) + aes( x = region, y = value, col = covid) +
    stat_summary(fun = "mean", geom = "point", size = 2) +
    stat_summary(fun.data = mean_cl_normal, geom = "errorbar", width = 0.3) +
    labs( x = "", y = lab3, title = "") + scale_color_manual(values = c("yes" = '#009392' , "no" =
'#d18975'))
  return(plot)
```

```
}
#function 11 : time series of 3 randomly chosen stations
spacetime_plot <- function(dataframe){

  sample <- dataframe[order(dataframe$station, dataframe$my),]
  Y <- dcast(data = sample, formula = station  ~ my, value.var ="value")
  Y <- na.omit(Y) #remove na

  set.seed(4)
  #choose randomly 3 stations
  stat <- sample(Y$station,3)

  #transform Y into multiple time series dataframe
  Y <- subset(Y, station %in% stat)
  Y <- t(Y)
  colnames(Y) <- Y[1,]
  Y <- Y[-1,]
  ts <- ts(Y, freq = 12,  start = c(2011,01)) #convert to ts
  plot(ts, main = "", xlab = "")

}
#function 12 : map of 3 randomly chosen dates
spacetime_plot2 <- function(dataframe){

  sample <- subset(dataframe@data, select = c(station, date, log_value, x, y))
  set.seed(123)
  random <- sample(sample$date, 2) #choose randomly 2 times

  p <- ggplot(subset(sample, date %in%random)) +
    aes( x = x , y = y, col = log_value) + geom_point(size = 4)+
    facet_grid( ~ date) + scale_color_gradient(low = "#FDDBC7", high = "#B2182B") +
    labs(col = "log(concentration)")

  return(p)

}
#function 13 : subset for modeling
get_subset <- function(dataframe){
  return(subset(dataframe@data,(year == 2019) | (year == 2020) ))
}
#function 14 : create variables for seasonality
update_4 <- function(dataframe){
  #create dummy variables for seasonnality
  dataframe$jan <- as.factor(ifelse(dataframe$month =="01", 1, 0))
  dataframe$feb <- as.factor(ifelse(dataframe$month =="02", 1, 0))
  dataframe$mar <- as.factor(ifelse(dataframe$month =="03", 1, 0))
  dataframe$apr <- as.factor(ifelse(dataframe$month =="04", 1, 0))
  dataframe$may <- as.factor(ifelse(dataframe$month =="05", 1, 0))
```

```
  dataframe$jun <- as.factor(ifelse(dataframe$month =="06", 1, 0))
  dataframe$jul <- as.factor(ifelse(dataframe$month =="07", 1, 0))
  dataframe$aug <- as.factor(ifelse(dataframe$month =="08", 1, 0))
  dataframe$sep <- as.factor(ifelse(dataframe$month =="09", 1, 0))
  dataframe$oct <- as.factor(ifelse(dataframe$month =="10", 1, 0))
  dataframe$nov<- as.factor(ifelse(dataframe$month =="11", 1, 0))
  dataframe$dec <- as.factor(ifelse(dataframe$month =="12", 1, 0))

  #convert to factor
  dataframe$region <- as.factor(dataframe$region)
  dataframe$month <- as.factor(dataframe$month)
  dataframe$year <- as.factor(dataframe$year)
  dataframe$week_time <- as.factor(dataframe$week_time)

  #create variable month-year
  dataframe$my <- paste(dataframe$month, dataframe$year, sep = "-")

  return(dataframe)
}
#function 15 : create variables for trend
trend_seasonal <- function(dataframe){

  #create time variable t for trend and d for seasonnality
  temps <- data.frame(unique(sub_PM10$date)[order(unique(sub_PM10$date))])
  temps <- data.frame(temps, c(1:731), c(seq(1, 365,1), seq(1, 366,1)))
  colnames(temps) <- c("time", "t", "d")
  dataframe <- merge(dataframe, temps, by.x = "date", by.y = "time", all.x = T)

  #create sinus and cos variable for seasonality
  dataframe$X1 <- sin((2*pi*dataframe$d)/ (365))
  dataframe$X2 <- cos((2*pi*dataframe$d)/ (365))
  return(dataframe)
}
#function 16 : plot observed vs predicted for trainset data
pred_train <- function(data){

  set.seed(123)
  st <- sample(data$station, 1)
  dt <- subset(data, station == st)

  plot <- ggplot(dt) + aes(x = date) +
    geom_line(aes(y = log_value, colour = "observed")) +
    geom_line(aes(y = pred, colour = "predicted")) +
    labs(y = "Y", x = "", title = st) +
    scale_color_manual(name = "", values = c("observed" = "black", "predicted" = "#DC143C"))

  return(plot)
}
```

```
#function 17 : Cross validation function
CV <- function(data, model){

  set.seed(123)
  n <- 700
  t80 <- floor(0.8 * n) #560 days
  t20 <- n - t80 #140 days
  tmp <- numeric(n)#to fill with the mean of the RMSE for the day
  for (i in t80+1:n){
    st.part <- subset(data, t %in% c(1:i-1))
    tmp.model <- lm(formula = model$call$formula, data = st.part)
    pred <- predict(tmp.model, subset(data, t == i))
    obs <- subset(data, t == i, select = c(log_value))
    tmp[i] <- round(sqrt(mean((obs-pred)[,1]^2)),2)
  }

  tmp <- tmp[!is.na(tmp)]
  return(tmp)
}
#function 18 : display residual for each day
get_sp_resday <- function(data, depvar){

  depvar <- enquo(depvar)
  set.seed(2)
  dates <- sample(data$date, 3)

  plot <- ggplot(subset(data, date %in% dates)) +
    geom_point(aes(x, y, colour = eval(depvar)), size = 4) +
    scale_color_continuous_diverging(palette = "Blue-Red 3", rev = T,mid = 0) +
    facet_wrap(~date, nrow = 1) + labs(col = "residuals")

  return(plot)
}
#function 19 : compute metrics from Moran test
get_Moran_day <- function(data){

  n <- 700
  times <- unique(data$date)
  stats <- matrix(0,nrow = n, ncol = 3)
  stats <- data.frame(stats)
  colnames(stats) <- c("IMoran", "pvalue", "time")

  for (i in 1:n){
    xy <- cbind(data$x[data$date == times[i]], data$y[data$date == times[i]])
    matrice.dists <- as.matrix(dist(xy, method = "euclidean"))
    matrice.dists.inv <- 1/matrice.dists
    diag(matrice.dists.inv) <- 0
    Moran <- Moran.I(data$res[data$date == times[i]], matrice.dists.inv)
```

```
   stats[i,] <- c(Moran$observed, Moran$p.value, times[i])
  }
  stats$IMoran <- as.numeric(stats$IMoran)
  stats$pvalue <- as.numeric(stats$pvalue)

  prop <- round(sum(stats$pvalue <= 0.05/n) / n,2)
  return(prop)
}
#function 20 : display the average residuals for each time t
get_temp_resmean_day <- function(data, depvar){

  depvar <- enquo(depvar)
  model_formula <- formula(paste(quo_name(depvar), "~ date"))

  dt <- aggregate(formula = model_formula, data = data, FUN = "mean" )
  ts <- ts(dt[,quo_name(depvar)], frequency = 365, start = c(2019, 01, 01))

  plot_ts <- autoplot(ts) + ylab("residual") +
    xlab("") +
    geom_abline(slope = 0, intercept = 0)

  return(list(ggAcf(ts, lag.max = sqrt(700), main =""), plot_ts))
}
#function 21 : plot observed vs predicted for testsets
pred_test <- function(data){

  set.seed(123)
  st <- sample(data$station, 1)
  dt <- subset(data, station == st)

  plot <- ggplot(dt) + aes(x = date) +
    geom_line(aes(y = log_value, colour = "observed")) +
    geom_line(aes(y = prediction, colour = "predicted")) +
    labs(y = "Y", x = "", title = st) +
    scale_color_manual(name = "", values = c("observed" = "black", "predicted" = "#DC143C"))

  return(plot)
}
#function 22 : compute minimum euclidean distance between kriging point estimate and sample
data location
dist <- function(kriging, sample){

  #create empty matrix
  mat <- matrix(nrow = nrow(sample), ncol = 3)
  colnames(mat) <- c("mindist", "X", "Y")


  for (i in 1:nrow(sample)){
```

```
    dists <- sqrt( (sample$x[i] - kriging@sp$X)^2 + (sample$y[i] - kriging@sp$Y)^2 )
    kriging@sp$dists <- dists

    #fill mat
    mat[i,"mindist"] <- min(dists)
    mat[i,"X"] <- kriging@sp$X[kriging@sp$dists == min(dists)]
    mat[i,"Y"] <- kriging@sp$Y[kriging@sp$dists == min(dists)]

  }
  return(mat)
}
```

```
## ----setup, include=FALSE----------------------------------------------------------------------
knitr::opts_chunk$set(echo = TRUE)


## ----------------------------------------------------------------------------------------------
library(spacetime)
library(adespatial)
library(dplyr)
library(ggplot2)
library(patchwork)
library(rgdal)
library(mapview)
library(cartography)
library(sf)
library(gstat)
library(sp)
library(automap)
library(RColorBrewer)
library(lmtest)
library(Ecdat)
library(orcutt)
library(ape)
library(wesanderson)
library(psych)
library(latex2exp)
library(reshape2)
library(orcutt)
library(ape)
library(lmtest)
library(odbc)
library(DBI)
library(tidyverse)
library(forecast)
library(bamlss)
library(spdep)
library(STRbook)
library(FRK)
library(MASS)
library(car)
library(dplyr)
library(tseries)
library(Hmisc)
library(corrplot)
library(SignifReg)
library(tmap)


## ----------------------------------------------------------------------------------------------
```

```
source("function/functions.R")


## ------------------------------------------------------------------------------------------------
memory.limit(size = 1000000)


## ------------------------------------------------------------------------------------------------
theme_set(theme_bw())
layout <- theme(axis.text.y = element_blank(), axis.ticks.y = element_blank())
lab1 <- TeX("concentration ($\\mu$g/m3)")
lab2 <- TeX("maximum concentration ($\\mu$g/m3)")
lab3 <- TeX("average concentration ($\\mu$g/m3)")


## ------------------------------------------------------------------------------------------------
prjs <- make_EPSG()  #chercher le tableau des projections
pos <- grep(31370, prjs[, "code"])


## ------------------------------------------------------------------------------------------------
#pollutants
PM10 <- read.csv('data/polluant/PM10.csv', header = T)
PM25 <- read.csv('data/polluant/PM25.csv', header = T)
O3 <- read.csv('data/polluant/O3.csv', header = T)
NO2 <- read.csv('data/polluant/NO2.csv', header = T)


## ------------------------------------------------------------------------------------------------
#stations with localisations
stations_irceline <- read.csv('data/station/stations.csv', header = T, sep =';', dec =",") #stations
IRCELINE
stations_converted <- read.csv('data/station/stations_converted.csv', header= T, sep = ',', dec = ".")
#stationsconverted
stations_irceline <- rbind(stations_irceline, stations_converted)


## ------------------------------------------------------------------------------------------------
X <- read.csv('data/X/add_converted.csv', header= T, sep = ';')
airport <- readOGR("data/X/airport/Aeroports_72.shp")

airport@data$PS <- airport@data$NEWFIELD1
airport@data$TYPE <- "airport"
airport@data <- airport@data[,-c(1,2)]
airport@data$lambert_x <- airport@coords[,1]
airport@data$lambert_y <- airport@coords[,2]

#create saptial object from X
```

```
X <- rbind(X, airport@data)
coordinates(X) <-  ~ lambert_x + lambert_y
X@proj4string <- CRS(prjs[pos, "prj4"])


## ------------------------------------------------------------------------------------------------
X_st <- read.csv(file = "data/X/temp.csv", header = T, dec = ".", sep = ";")
X_st$date <- as.Date(X_st$date, format = "%d/%m/%Y")


## ------------------------------------------------------------------------------------------------
stat_uniq <- data.frame(unique(stations_irceline['ab_local_code']))
stat_uniq <- merge(stations_irceline[,c("ab_local_code","region", "denspop",'vegetation',
"lambert_x", "lambert_y")], stat_uniq, by = 'ab_local_code', all.y = T)
colnames(stat_uniq)[which(names(stat_uniq) == "lambert_x")] <- "x"
colnames(stat_uniq)[which(names(stat_uniq) == "lambert_y")] <- "y"
stat_uniq <- conv_spatial(stat_uniq)
stat_uniq@data$LPS_closest <- min_dist(stat_uniq, X, "LPS")
stat_uniq@data$city_closest <- min_dist(stat_uniq, X, "COMMUNE")
stat_uniq@data$airport_closest <- min_dist(stat_uniq, X, "airport")
stat_uniq@data$x <- stat_uniq@coords[,1]
stat_uniq@data$y <- stat_uniq@coords[,2]


## ------------------------------------------------------------------------------------------------
PM10 <- update(PM10)
PM25 <- update(PM25)
O3 <- update(O3)
NO2 <- update(NO2)


## ----missing_values, eval = FALSE----------------------------------------------------------------
## #idea : regroup values by taking mean of the day
## N <- nrow(PM10) + nrow(PM25) + nrow(O3) + nrow(NO2)
## n_miss <- length(PM10$value[PM10$value == -9999]) + length(PM25$value[PM25$value == -
999.9]) + length(PM25$value[PM25$value == -9999]) + length(O3$value[O3$value == -9999]) +
length(NO2$value[NO2$value == -9999])
##
## #global missing value percentage
## (n_miss / N) * 100
##
## #by polluant
## length(PM10$value[PM10$value == -9999]) / nrow(PM10)
## (length(PM25$value[PM25$value == -999.9]) + length(PM25$value[PM25$value == -9999])) /
(nrow(PM25))
## length(O3$value[O3$value == -9999]) / (nrow(O3))
## length(NO2$value[NO2$value == -9999]) /nrow(NO2)
## #by hour
```

```
## prop.table(xtabs( PM10$value == - 9999 ~ PM10$hour))


## ------------------------------------------------------------------------------------------------
#remove missing values from the datasets
PM10 <- subset(PM10, value != -9999)
PM25 <- subset(PM25, value !=-9999 & value != -999.9)
O3 <- subset(O3, value != -9999)
NO2 <- subset(NO2, value !=-9999)


## ------------------------------------------------------------------------------------------------
uni_plot("PM10")
uni_plot("PM25")
uni_plot("O3")
uni_plot("NO2")

#summary statistics
round(summary(PM10$value),2)
round(summary(PM25$value),2)
round(summary(O3$value),2)
round(summary(NO2$value),2)
#standard deviation
round(sd(PM10$value) * ((length(PM10$value) - 1) / (length(PM10$value))),2)
round(sd(PM25$value) * ((length(PM25$value) - 1) / (length(PM25$value))),2)
round(sd(O3$value) * ((length(O3$value) - 1) / (length(O3$value))),2)
round(sd(NO2$value) * ((length(NO2$value) - 1) / (length(NO2$value))),2)


## ------------------------------------------------------------------------------------------------
hourly_plot("PM10")
hourly_plot("PM25")
hourly_plot("O3")
hourly_plot("NO2")


## ------------------------------------------------------------------------------------------------
#station with maximum values
statmax_PM10 <- PM10$station[PM10$value == max(PM10$value)]
statmax_PM25 <- PM25$station[PM25$value == max(PM25$value)]
statmax_O3 <- O3$station[O3$value == max(O3$value)]
statmax_NO2 <- NO2$station[NO2$value == max(NO2$value)]

extreme_plot(statmax_PM10, "PM10")
extreme_plot(statmax_PM25, "PM25")
extreme_plot(statmax_O3, "O3")
extreme_plot(statmax_NO2, "NO2")
```

```
## ----1st aggregation----------------------------------------------------------------------
#take the mean with respect to the station and the full date
PM10_d <- aggregate( formula = value ~ date + station, data = PM10, FUN = "mean")
PM25_d <- aggregate( formula = value ~ date + station, data = PM25, FUN = "mean")
O3_d <- aggregate( formula = value ~ date + station, data = O3, FUN = "mean")
NO2_d <- aggregate( formula = value ~ date + station, data = NO2, FUN = "mean")

#merge with spatial variables
PM10_d <- merge(PM10_d, stat_uniq@data[,c("ab_local_code", "region", "denspop", "vegetation",
"x", "y", "city_closest", "airport_closest", "LPS_closest")], by.x = "station", by.y = "ab_local_code",
all.x = T)
PM25_d <- merge(PM25_d, stat_uniq@data[,c("ab_local_code", "region", "denspop", "vegetation",
"x", "y", "city_closest", "airport_closest", "LPS_closest")], by.x = "station", by.y = "ab_local_code",
all.x = T)
O3_d <- merge(O3_d, stat_uniq@data[,c("ab_local_code", "region", "denspop", "vegetation", "x",
"y", "city_closest", "airport_closest", "LPS_closest")], by.x = "station", by.y = "ab_local_code", all.x =
T)
NO2_d <- merge(NO2_d, stat_uniq@data[,c("ab_local_code", "region", "denspop", "vegetation", "x",
"y", "city_closest", "airport_closest", "LPS_closest")], by.x = "station", by.y = "ab_local_code", all.x =
T)

#update2
PM10_d <- update_2(PM10_d)
PM25_d <- update_2(PM25_d)
O3_d <- update_2(O3_d)
NO2_d <- update_2(NO2_d)


## ------------------------------------------------------------------------------------------------
#add region variable to initial dataframes
PM10 <- merge(PM10, stat_uniq@data[,c("ab_local_code", "region")], by.x = "station", by.y =
"ab_local_code", all.x = T)
PM25 <- merge(PM25, stat_uniq@data[,c("ab_local_code", "region")], by.x = "station", by.y =
"ab_local_code", all.x = T)
O3 <- merge(O3, stat_uniq@data[,c("ab_local_code", "region")], by.x = "station", by.y =
"ab_local_code", all.x = T)
NO2 <- merge(NO2, stat_uniq@data[,c("ab_local_code", "region")], by.x = "station", by.y =
"ab_local_code", all.x = T)

#nb stations unique pour chaque polluant
stat1 <- unique(PM10$station)
stat2 <- unique(PM25$station)
stat3 <- unique(O3$station)
stat4 <- unique(NO2$station)

pol1 <- data.frame(table(PM10$station, PM10$region))
pol1 <- pol1[pol1$Freq >0,]
```

```
table(pol1$Var2)

stat_common <- Reduce(intersect, list(stat1, stat2, stat3, stat4))#station en commun
stat_all_unique <- unique(c(stat1, stat2, stat3, stat4)) #ensemble des stations différentes

#PM10
summary1 <- as.data.frame(table(PM10$station, PM10$year))
summary1 <- subset(summary1, Freq >0)
table(summary1$Var2) #nb stations pour chaque année


## ------------------------------------------------------------------------------------------
set.seed(123)
merge <- merge(PM10_d@data, O3_d@data[,c("station", "date", "value")], by = c("station", "date"))
merge <- sample_n(merge, 1000)
ggplot(merge) + aes(x = value.x, y = value.y, col = region) +
   geom_point(alpha = 0.7, size = 2) +
   labs( col = "", x = "PM10", y = "O3") + theme(legend.text = element_text(size = 13))


## ----cartography----------------------------------------------------------------------------
#read shp files belgian cities
polyg <- readOGR("data/X/commune/Communes.shp")
polyg@proj4string <- CRS(prjs[pos, "prj4"])

PM10_d <- conv_spatial(PM10_d)
PM25_d <- conv_spatial(PM25_d)
O3_d <- conv_spatial(O3_d)
NO2_d <- conv_spatial(NO2_d)


## ------------------------------------------------------------------------------------------
Sys.setlocale("LC_TIME", "English")
PM10_d <- update_3(PM10_d)
PM25_d <- update_3(PM25_d)
O3_d <- update_3(O3_d)
NO2_d <- update_3(NO2_d)


## ----analyse_time_series--------------------------------------------------------------------
#density
p1 <- ggplot(subset(PM10_d@data, value >0)) +
  aes(x = log(value), fill = week_time) +
  geom_density(alpha = 0.8) +
  labs(x = "log(concentration)", title = "PM10") +
    scale_fill_manual(values = c("week" = '#d18975', "weekend" = '#009392'),
                      labels = c("workdays", "weekend")) +
  theme(legend.position = "none")
```

```
p2 <- ggplot(subset(PM25_d@data, value >0)) +
  aes(x = log(value), fill = week_time) +
  geom_density(alpha = 0.8) +
  labs(x = "log(concentration)", title = "PM2.5") +
  scale_fill_manual(values = c("week" = '#d18975', "weekend" = '#009392'),
                    labels = c("workdays", "weekend")) +
  theme(legend.position = "none")

p3 <- ggplot(subset(O3_d@data, value >0)) +
  aes(x = log(value), fill = week_time) +
  geom_density(alpha = 0.8) +
  labs(x = "log(concentration)", title = "O3") +
  scale_fill_manual(values = c("week" = '#d18975', "weekend" = '#009392'),
                    labels = c("workdays", "weekend")) +
  theme(legend.position = "none")

p4 <- ggplot(subset(NO2_d@data, value >0)) +
  aes(x = log(value), fill = week_time) +
  geom_density(alpha = 0.8) +
  labs(x = "log(concentration)", title = "NO2") +
  scale_fill_manual(values = c("week" = '#d18975', "weekend" = '#009392'),
                    labels = c("workdays", "weekend")) +
  theme(legend.title = element_blank())


p1 + p2 + p3 + p4


##test mean difference
#plot
ggplot(data = subset(PM10_d@data, value > 0 & week_time == "week"), aes(sample = log(value))) +
       stat_qq() + stat_qq_line(col = "red") + labs(title = "PM10_d : week")
ggplot(data = subset(PM10_d@data, value > 0 & week_time == "weekend"), aes(sample =
log(value))) +
       stat_qq() + stat_qq_line(col = "red") + labs(title = "PM10_d : weekend")

#Wilcoxon rank sum test
wilcox.test(log(subset(PM10_d@data, value > 0 & week_time == "week")$value),
         log(subset(PM10_d@data, value > 0 & week_time == "weekend")$value),
         alternative = "less")


## ------------------------------------------------------------------------------------------
#year
y_PM10 <- aggregate(value ~ year, data = PM10, FUN = mean)
y_PM25 <- aggregate(value ~ year, data = PM25, FUN = mean)
y_O3 <- aggregate(value ~ year, data = O3, FUN  = mean)
```

```
y_NO2 <- aggregate(value ~ year, data = NO2, FUN = mean)

y_all <- data.frame(PM10 = y_PM10$value, PM25 = y_PM25$value, O3 = y_O3$value, NO2 =
y_NO2$value, year = y_PM10$year)
y_all <- melt(y_all, id.vars = "year")
colnames(y_all)[2] <- "pollutant"

ggplot(y_all) + aes(x = year, y = value, group = pollutant, col = pollutant) + geom_line(aes(linetype =
pollutant), size = 1) + labs(y = lab1, x = "year") + ylim(c(0,60))

#month
m_PM10 <- aggregate(value ~ month, data = PM10, FUN = mean)
m_PM25 <- aggregate(value ~ month, data = PM25, FUN = mean)
m_O3 <- aggregate(value ~ month, data = O3, FUN  = mean)
m_NO2 <- aggregate(value ~ month, data = NO2, FUN = mean)

m_all <- data.frame(PM10 = m_PM10$value, PM25 = m_PM25$value, O3 = m_O3$value, NO2 =
m_NO2$value, month = m_PM10$month)
m_all <- melt(m_all, id.vars = "month")
colnames(m_all)[2] <- "pollutant"
ggplot(m_all) + aes( x = month, y = value, group = pollutant, col = pollutant) + geom_line(aes(linetype
= pollutant), size = 1)+ labs(y = lab1)+ theme(legend.position = "none") + ylim(c(0,70))



## ------------------------------------------------------------------------------------------
#covid : 1 or 0
PM10_d@data$covid <- ifelse(PM10_d@data$date > as.Date("2020-03-01", format = "%Y-%m-
%d"),"yes","no")
PM10_d@data$covid <- as.factor(PM10_d@data$covid)
PM25_d@data$covid <- ifelse(PM25_d@data$date > as.Date("2020-03-01", format = "%Y-%m-
%d"),"yes","no")
PM25_d@data$covid <- as.factor(PM25_d@data$covid)
O3_d@data$covid <- ifelse( O3_d@data$date > as.Date("2020-03-01", format = "%Y-%m-
%d"),"yes","no")
O3_d@data$covid <- as.factor(O3_d@data$covid)
NO2_d@data$covid <- ifelse( NO2_d@data$date > as.Date("2020-03-01", format = "%Y-%m-
%d"),"yes","no")
NO2_d@data$covid <- as.factor(NO2_d@data$covid)

covid_plot("PM10_d")

p1 <- ggplot(PM10_d@data) + aes( x = region, y = value, col = covid) +
   stat_summary(fun = "mean", geom = "point", size = 2) +
   stat_summary(fun.data = mean_cl_normal, geom = "errorbar", width = 0.3, size = 0.5) +
   labs( x = "", y = lab1, title = "PM10") + scale_color_manual(values = c("yes" = '#009392' , "no" =
'#d18975')) +
   theme(legend.position = "none")
```

```
p2 <- ggplot(PM25_d@data) + aes( x = region, y = value, col = covid) +
   stat_summary(fun = "mean", geom = "point", size = 2) +
   stat_summary(fun.data = mean_cl_normal, geom = "errorbar", width = 0.3, size = 0.5) +
   labs( x = "", y = lab1, title = "PM2.5") + scale_color_manual(values = c("yes" = '#009392' , "no" =
'#d18975'))

p3 <- ggplot(O3_d@data) + aes( x = region, y = value, col = covid) +
   stat_summary(fun = "mean", geom = "point", size = 2) +
   stat_summary(fun.data = mean_cl_normal, geom = "errorbar", width = 0.3, size = 0.5) +
   labs( x = "", y = lab1, title = "O3") + scale_color_manual(values = c("yes" = '#009392' , "no" =
'#d18975'))+
   theme(legend.position = "none")

p4 <- ggplot(NO2_d@data) + aes( x = region, y = value, col = covid) +
   stat_summary(fun = "mean", geom = "point", size = 2) +
   stat_summary(fun.data = mean_cl_normal, geom = "errorbar", width = 0.3, size = 0.5) +
   labs( x = "", y = lab1, title = "NO2") + scale_color_manual(values = c("yes" = '#009392' , "no" =
'#d18975'))+
   theme(legend.position = "none")

p1 + p2 + p3 + p4


## ----TS----------------------------------------------------------------------------------------
#Time series of O3 monthly observations (mean) from 2011 to 2020
ts3 <- ts(aggregate(value ~ month + year, data = O3_d, FUN = mean)$value, freq = 12, start = 2011)
autoplot(ts3) + ylab(lab1) + xlab("")

#statistical summary
summary(ts3)

#seasonality
ggseasonplot(ts3, continuous = T, year.labels = T)+ labs(title="", x = "")

#differentation
ts.1 <- diff(ts3, lag = 12)
autoplot(ts.1) + ylab(TeX("$Y_t")) + xlab("")



#stationary test
#Dickey-Fuller test
adf.test(ts.1, alternative = "stationary") # stationarity not rejected

#ACF and PACF
AF <- ggAcf(ts.1, lag.max = 70) + labs (title = "")
PAF <- ggPacf(ts.1, lag.max = 70) + labs(title ="")
AF + PAF
```

```
##model 1
md1 <- Arima(ts3, order = c(0,0,0), seasonal = c(0,1,1) )
summary(md1)
coeftest(md1)
AIC(md1)

#diagnostics
tsdiag(md1, gof.lag = sqrt(length(ts3)))

#prediction model 1
pred.ts <- predict(md1, n.ahead = 12, prediction.interval = T, level = 0.95)
plot(ts3, xlim = c(2011,2022), ylab = lab1, xlab ="")
lines(pred.ts$pred, col = "red")
lines(pred.ts$pred + 1.96*pred.ts$se, lty = 2, col = "blue")
lines(pred.ts$pred - 1.96*pred.ts$se, lty = 2, col = "blue")


## ------------------------------------------------------------------------------------------
#mdl : dataset for modeling with only values greater than 0 to apply log
PM10_mdl <- subset(PM10_d, value > 0)
PM25_mdl <- subset(PM25_d, value > 0)
O3_mdl <- subset(O3_d, value > 0)
NO2_mdl <- subset(NO2_d, value > 0)

#create log_value
PM10_mdl@data$log_value <- log(PM10_mdl@data$value)
PM25_mdl@data$log_value <- log(PM25_mdl@data$value)
O3_mdl@data$log_value <- log(O3_mdl@data$value)
NO2_mdl@data$log_value <- log(NO2_mdl@data$value)

#x and y coordinates
PM10_mdl@data$x <- PM10_mdl@coords[,1]
PM10_mdl@data$y <- PM10_mdl@coords[,2]
PM25_mdl@data$x <- PM25_mdl@coords[,1]
PM25_mdl@data$y <- PM25_mdl@coords[,2]
O3_mdl@data$x <- O3_mdl@coords[,1]
O3_mdl@data$y <- O3_mdl@coords[,2]
NO2_mdl@data$x <- NO2_mdl@coords[,1]
NO2_mdl@data$y <- NO2_mdl@coords[,2]


## ------------------------------------------------------------------------------------------
#PM10
ts1 <- ts(aggregate(value ~ month + year, data = PM10, FUN = mean)$value, freq = 12, start = 2011)
#PM25
ts2 <- ts(aggregate(value ~ month + year, data = PM25, FUN = mean)$value, freq = 12, start = 2011)
#O3
#NO2
```

```
ts4 <- ts(aggregate(value ~ month + year, data = NO2, FUN = mean)$value, freq = 12, start = 2011)

#viz
autoplot(ts4) + ylab(lab1) + xlab("")


## ------------------------------------------------------------------------------------------
#data processing
my1 <- aggregate( value ~ station + month + year, data = PM10, FUN = "mean")
my2 <- aggregate( value ~ station + month + year, data = PM25, FUN = "mean")
my3 <- aggregate( value ~ station + month + year, data = O3, FUN = "mean")
my4 <- aggregate( value ~ station + month + year, data = NO2, FUN = "mean")

my1$my <- as.yearmon(paste(my1$year, my1$month, sep = "-"))
my2$my <- as.yearmon(paste(my2$year, my2$month, sep = "-"))
my3$my <- as.yearmon(paste(my3$year, my3$month, sep = "-"))
my4$my <- as.yearmon(paste(my4$year, my4$month, sep = "-"))

#viz
spacetime_plot(my1)
spacetime_plot(my2)
spacetime_plot(my3)
spacetime_plot(my4)

spacetime_plot2(PM10_mdl)
spacetime_plot2(PM25_mdl)
spacetime_plot2(O3_mdl)
spacetime_plot2(NO2_mdl)


## ------------------------------------------------------------------------------------------
gridK <- makegrid(polyg, cellsize = c(5000,5000))
names(gridK) <- c("X", "Y")
coordinates(gridK) <- c("X", "Y")

grd_pts <- SpatialPoints(
  coords      = gridK,
  proj4string = CRS(prjs[pos, "prj4"])
)
grd_pts_in <- grd_pts[polyg, ]
ggplot(as.data.frame(coordinates(grd_pts_in))) +
  geom_point(aes(X, Y))

gridded(grd_pts_in) <- T
plot(grd_pts_in)


## ------------------------------------------------------------------------------------------
```

```
sub_PM10 <- get_subset(PM10_mdl)
sub_PM25 <- get_subset(PM25_mdl)
sub_O3 <- get_subset(O3_mdl)
sub_NO2 <- get_subset(NO2_mdl)

#merge with space time variant variables
sub_PM10 <- merge(sub_PM10, X_st, by.x = c("station", "date"), by.y = c("ab_local_code", "date"),
all.x = T)
sub_PM25 <- merge(sub_PM25, X_st, by.x = c("station", "date"), by.y = c("ab_local_code", "date"),
all.x = T)
sub_O3 <- merge(sub_O3, X_st, by.x = c("station", "date"), by.y = c("ab_local_code", "date"), all.x = T)
sub_NO2 <- merge(sub_NO2, X_st, by.x = c("station", "date"), by.y = c("ab_local_code", "date"), all.x
= T)

#update 4
sub_PM10 <- update_4(sub_PM10)
sub_PM25 <- update_4(sub_PM25)
sub_O3 <- update_4(sub_O3)
sub_NO2 <- update_4(sub_NO2)

#add trend and seasonal variables
sub_PM10 <- trend_seasonal(sub_PM10)
sub_PM25 <- trend_seasonal(sub_PM25)
sub_O3 <- trend_seasonal(sub_O3)
sub_NO2 <- trend_seasonal(sub_NO2)

#trainsets
trainset1 <- subset(sub_PM10, t %in% c(1:700))
trainset2 <- subset(sub_PM25, t %in% c(1:700))
trainset3 <- subset(sub_O3, t %in% c(1:700))
trainset4 <- subset(sub_NO2, t %in% c(1:700))

#testsets
testset1 <- subset(sub_PM10, t %in% c(701:731))
testset2 <- subset(sub_PM25, t %in% c(701:731))
testset3 <- subset(sub_O3, t %in% c(701:731))
testset4 <- subset(sub_NO2, t %in% c(701:731))

#set reference levels
trainset1$region <- relevel(trainset1$region, ref = "Wallonia")
trainset2$region <- relevel(trainset2$region, ref = "Wallonia")
trainset3$region <- relevel(trainset3$region, ref = "Wallonia")
trainset4$region <- relevel(trainset4$region, ref = "Wallonia")



## ------------------------------------------------------------------------------------------
cols <- c("X1 (sinus)" = '#009392', "X2 (cosinus)" = '#eeb479')
```

```
ggplot(trainset1) + aes(x = d) + geom_point(aes(y = X1, colour = "X1 (sinus)")) + geom_point(aes(y = X2, colour = "X2 (cosinus)")) + labs(colour = "variable") + scale_color_manual(values = cols) + labs(y = "")


## ------------------------------------------------------------------------------------------
#variables decomposition
temp <- c("t", "feb", "mar", "apr", "may", "jun", "jul", "aug", "sep", "oct", "nov", "dec",  "week_time")
temp2 <- c("t", "X1", "X2", "week_time")
space <- c(" + vegetation" , "denspop", "region", "airport_closest", "LPS_closest", "city_closest", "x", "y")
temp_space <- c(" + temperature", "precipitation", "pressure", "windspeed", "humidity")

#model 1
formula1 <- formula(paste("log_value ~ ", paste(temp, collapse=" + "), paste(space, collapse = " + "), paste(temp_space, collapse = " + ") ))

#model 2
formula2 <- formula(paste("log_value ~ ", paste(temp2, collapse=" + "), paste(space, collapse = " + "), paste(temp_space, collapse = " + ") ))

#models
#model 1
lm1_PM10 <- lm(formula1, data = trainset1)
lm1_PM25 <- lm(formula1, data = trainset2)
lm1_O3 <- lm(formula1, data = trainset3)
lm1_NO2 <- lm(formula1, data = trainset4)

#model 2
lm2_PM10 <- lm(formula2, data = trainset1)
lm2_PM25 <- lm(formula2, data = trainset2)
lm2_O3 <- lm(formula2, data = trainset3)
lm2_NO2 <- lm(formula2, data = trainset4)


## ------------------------------------------------------------------------------------------
#RMSE
round(sqrt(mean(lm1_PM10$residuals^2)),2)
round(sqrt(mean(lm1_PM25$residuals^2)),2)
round(sqrt(mean(lm1_O3$residuals^2)),2)
round(sqrt(mean(lm1_NO2$residuals^2)),2)

round(sqrt(mean(lm2_PM10$residuals^2)),2)
round(sqrt(mean(lm2_PM25$residuals^2)),2)
round(sqrt(mean(lm2_O3$residuals^2)),2)
round(sqrt(mean(lm2_NO2$residuals^2)),2)

#AIC
```

```
extractAIC(lm1_PM10)
extractAIC(lm1_PM25)
extractAIC(lm1_O3)
extractAIC(lm1_NO2)

extractAIC(lm2_PM10)
extractAIC(lm2_PM25)
extractAIC(lm2_O3)
extractAIC(lm2_NO2)

#BIC
BIC(lm1_PM10)
BIC(lm1_PM25)
BIC(lm1_O3)
BIC(lm1_NO2)

BIC(lm2_PM10)
BIC(lm2_PM25)
BIC(lm2_O3)
BIC(lm2_NO2)

#Adjusted R squared
round(summary(lm1_PM10)$adj.r.squared,2)
round(summary(lm1_PM25)$adj.r.squared,2)
round(summary(lm1_O3)$adj.r.squared,2)
round(summary(lm1_NO2)$adj.r.squared,2)

round(summary(lm2_PM10)$adj.r.squared,2)
round(summary(lm2_PM25)$adj.r.squared,2)
round(summary(lm2_O3)$adj.r.squared,2)
round(summary(lm2_NO2)$adj.r.squared,2)


## ------------------------------------------------------------------------------------------
trainset1$pollutant <- "PM10"
trainset2$pollutant <- "PM25"
trainset3$pollutant <- "O3"
trainset4$pollutant <- "NO2"
trainset_all <- rbind(trainset1, trainset2)
trainset_all <- rbind(trainset_all, trainset3)
trainset_all <- rbind(trainset_all, trainset4)

qt <- subset(trainset_all, select = c(log_value, temperature, pressure, windspeed, humidity,
precipitation, denspop, vegetation, LPS_closest, city_closest, airport_closest, x, y, X1, X2, t))
qt <- distinct(qt, temperature, pressure, windspeed, humidity, precipitation, denspop, vegetation,
LPS_closest, city_closest, airport_closest, x, y, X1, X2,t, .keep_all = TRUE)
colnames(qt)[which(names(qt) == "log_value")] <- "Y"
```

```
c <- rcorr(as.matrix(qt))
corrplot(c$r, type = "full", method = "color", tl.col = "black", tl.srt = 45, diag = T)

#plot correlations greater than
ggplot(qt) + aes(x = temperature, y = X2) + geom_point()
ggplot(qt) + aes(x = vegetation, y = denspop) + geom_point()
ggplot(qt) + aes(x = vegetation, y = airport_closest) + geom_point()
ggplot(qt) + aes(x = vegetation, y = city_closest) + geom_point()
ggplot(qt) + aes(x = vegetation, y = LPS_closest) + geom_point()
ggplot(qt) + aes( x = airport_closest, y = city_closest) + geom_point()

#VIF
reg1 <- lm(X2 ~ ., data = subset(qt, select = - Y))
round((1) / (1- summary(reg1)$r.squared),2)

#correlations
round(cor(qt$LPS_closest, qt$vegetation), 2)


## ------------------------------------------------------------------------------------------
set.seed(123)
# Stepwise regression model
step1 <- SignifReg(fit = lm2_PM10, scope = list(lower = lm(log_value ~ t + X1 + X2, data = trainset1),
upper = lm2_PM10), alpha = 0.05, direction = "both", criterion = "p-value", adjust.method =
"bonferroni", trace = F)

step2 <- SignifReg(fit = lm2_PM25, scope = list(lower = lm(log_value ~ t + X1 + X2, data = trainset2),
upper = lm2_PM25), alpha = 0.05, direction = "both", criterion = "p-value", adjust.method =
"bonferroni", trace = F)

step3 <- SignifReg(fit = lm2_O3, scope = list(lower = lm(log_value ~ t + X1 + X2, data = trainset3),
upper = lm2_O3), alpha = 0.05, direction = "both", criterion = "p-value", adjust.method =
"bonferroni", trace = F)

step3_2 <- SignifReg(fit = lm2_O3, alpha = 0.05, direction = "both", criterion = "p-value",
adjust.method = "bonferroni", trace = F)

step4 <- SignifReg(fit = lm2_NO2, scope = list(lower = lm(log_value ~ t + X1 + X2, data = trainset4),
upper = lm2_NO2), alpha = 0.05,  direction = "both", criterion = "p-value", adjust.method =
"bonferroni", trace = F)


## ------------------------------------------------------------------------------------------
#add residuals and pred to learning dataframe

#PM10
trainset1$res <- step1$residuals
trainset1$pred <- step1$fitted.values
```

```
#PM25
trainset2$res <- step2$residuals
trainset2$pred <- step2$fitted.values
#O3
trainset3$res <- step3$residuals
trainset3$pred <- step3$fitted.values
#NO2
trainset4$res <- step4$residuals
trainset4$pred <- step4$fitted.values

#observed vs predicted : time series for one randomly station
pred_train(trainset1)
pred_train(trainset2)
pred_train(trainset3)
pred_train(trainset4)

#RMSE
round(sqrt(mean(trainset1$res^2)),2) #0.49
round(sqrt(mean(trainset2$res^2)),2) #0.84
round(sqrt(mean(trainset3$res^2)),2) # 0.44
round(sqrt(mean(trainset4$res^2)),2) #0.48

#AIC
extractAIC(step1)
extractAIC(step2)
extractAIC(step3)
extractAIC(step4)

#BIC
BIC(step1)
BIC(step2)
BIC(step3)
BIC(step4)

#Adjusted R squared
round(summary(step1)$adj.r.squared,2)
round(summary(step2)$adj.r.squared,2)
round(summary(step3)$adj.r.squared,2)
round(summary(step4)$adj.r.squared,2)


#plots obs against predicted
ggplot(trainset4) + aes(x = pred, y = log_value) + geom_point() +labs(y = "Y", x = "prediction") +
geom_abline(slope = 1)
round(cor(trainset1$pred, trainset1$log_value),2)


## ------------------------------------------------------------------------------------------
```

```
cv_PM10 <- CV(trainset1, step1)
cv_PM25 <- CV(trainset2, step2)
cv_O3 <- CV(trainset3, step3)
cv_NO2 <- CV(trainset4, step4)

t20 <- 140  #20% left observations

#mean of RMSE
round(sum(cv_PM10)/t20,2)
round(sum(cv_PM25)/t20,2)
round(sum(cv_O3)/t20,2)
round(sum(cv_NO2)/t20,2)

#standard deviation of RMSE
round(sd(cv_PM10[561:700])*((t20-1)/(t20)),2)
round(sd(cv_PM25[561:700])*((t20-1)/(t20)),2)
round(sd(cv_O3[561:700])*((t20-1)/(t20)),2)
round(sd(cv_NO2[561:700])*((t20-1)/(t20)),2)


## ------------------------------------------------------------------------------------------
#map resday
get_sp_resday(trainset1, res)
get_sp_resday(trainset2, res)
get_sp_resday(trainset3, res)
get_sp_resday(trainset4, res)

#proportion I de Moran
get_Moran_day(trainset1) #0.46
get_Moran_day(trainset2) #0.44
get_Moran_day(trainset3) #0.28
get_Moran_day(trainset4) #0.06

#Temporal
get_temp_resmean_day(trainset1, res)
get_temp_resmean_day(trainset2, res)
get_temp_resmean_day(trainset3, res)
get_temp_resmean_day(trainset4, res)


## ------------------------------------------------------------------------------------------
tlags <- c(0:4)
#PM10
STIDF_PM10 <- STIDF(sp = SpatialPoints(trainset1[,c("x","y")], proj4string = CRS(prjs[pos, "prj4"])),
time = trainset1$date,
data = trainset1)
#PM25
STIDF_PM25 <- STIDF(sp = SpatialPoints(trainset2[,c("x","y")], proj4string = CRS(prjs[pos, "prj4"])),
```

```
time = trainset2$date,
data = trainset2)
#O3
STIDF_O3 <- STIDF(sp = SpatialPoints(trainset3[,c("x","y")], proj4string = CRS(prjs[pos, "prj4"])),
time = trainset3$date,
data = trainset3)
#NO2
STIDF_NO2 <- STIDF(sp = SpatialPoints(trainset4[,c("x","y")], proj4string = CRS(prjs[pos, "prj4"])),
time = trainset4$date,
data = trainset4)

#to run during the night
varST <- variogramST(res ~ 1, STIDF_PM10 , na.omit = T, tlags = tlags, tunit = "days")
varST2 <- variogramST(res ~ 1, STIDF_PM25 , na.omit = T, tlags = tlags, tunit = "days")
varST3 <- variogramST(res ~ 1, STIDF_O3 , na.omit = T, tlags = tlags, tunit = "days")
varST4 <- variogramST(res ~ 1, STIDF_NO2 , na.omit = T, tlags = tlags, tunit = "days")

#viz
color_pal2 <- colorRampPalette(brewer.pal(9, "YlOrRd"))(16) #st_2D
color_pal3 <- colorRampPalette(brewer.pal(9, "YlOrRd"))(75) #st_3D
plot(varST, col = color_pal2)
varST$timelag <- as.factor(varST$timelag)
ggplot(varST) +
  aes(x = dist, y = gamma, group = timelag, col = timelag) +
  geom_line(size = 1) +
  labs(x = "distance(m)")+
  ylim(0,0.29) +
  scale_color_discrete(name = "timelag", labels = c("1", "2", "3", "4"))



## ------------------------------------------------------------------------------------------
model_formula <- formula("res ~ date")
dt <- aggregate(formula = model_formula, data = trainset4, FUN = "mean" )
TS <- ts(dt["res"], frequency = 365, start = c(2019, 01, 01))
autoplot(TS) + ylab("average residual") + xlab("") + geom_abline(slope = 0, intercept = 0)

#statistical summary
summary(TS)

#stationary test
#Dickey-Fuller test
adf.test(TS, k = 12, alternative = "stationary") # stationarity not rejected

#ACF and PACF
ggAcf(TS, lag.max = sqrt(700)) + labs (title = "")
ggPacf(TS, lag.max = sqrt(700)) + labs(title ="")

##model 1
```

```
md1 <- Arima(TS, order = c(1,0,0), seasonal = c(0,0,0), include.mean = F)
summary(md1)
coeftest(md1)

#diagnostics
tsdiag(md1, gof.lag = sqrt(length(TS)))

#prediction model 1
pred.ts <- predict(md1, n.ahead = 31, prediction.interval = T, level = 0.95)
plot(TS, xlim = c(2019,2021), ylab = "average residual", xlab ="")
lines(pred.ts$pred, col = "red")
lines(pred.ts$pred + 1.96*pred.ts$se, lty = 2, col = "blue")
lines(pred.ts$pred - 1.96*pred.ts$se, lty = 2, col = "blue")


## ------------------------------------------------------------------------------------------
color_pal <- rev(colorRampPalette(brewer.pal(16, "Spectral"))(16)) #diff plots


## ------------------------------------------------------------------------------------------
#metric
metricVgm <- vgmST(stModel = "metric",
joint = vgm(0.15, "Exp", 60000, nugget = 0.05),
sill = 0.15,
stAni = 30000, temporalUnit = "days")
metricVgm <- fit.StVariogram(varST, metricVgm)
attr(metricVgm, "optim")$value
#sum metric
smetricVgm <- vgmST(stModel = "sumMetric",
                space = vgm(psill=0.15,"Exp", range=60000, nugget=0.05),
                time = vgm(psill= 0.15,"Exp", range= 2, nugget=0.05),
                joint = vgm(0.15, "Exp", 60000, nugget = 0.05),
                sill = 0.15,
                stAni = 30000)
smetricVgm <- fit.StVariogram(varST, smetricVgm)
attr(smetricVgm, "optim")$value
#productsum
prodSumVgm <- vgmST("productSum",
                space = vgm(0.15, "Exp", 60000, 0.05),
                time = vgm(0.15, "Exp", 2, 0.05), k = 2)
prodSumVgm <- fit.StVariogram(varST, prodSumVgm)
attr(prodSumVgm, "optim")$value

#viz
plot(varST, list(smetricVgm, prodSumVgm, metricVgm), diff = T, col = color_pal )


## ------------------------------------------------------------------------------------------
```

```
#metric
metricVgm <- vgmST(stModel = "metric",
joint = vgm(0.70, "Sph", 60000, nugget = 0.2),
sill = 0.7,
stAni = 30000)
metricVgm <- fit.StVariogram(varST2, metricVgm)
attr(metricVgm, "optim")$value
#sum metric
smetricVgm <- vgmST(stModel = "sumMetric",
                space = vgm(psill=0.70,"Sph", range=60000, nugget=0.2),
                time = vgm(psill= 0.70,"Sph", range= 2, nugget=0.2),
                joint = vgm(0.70, "Sph", 60000, nugget = 0.2),
                sill = 0.70,
                stAni = 30000)
smetricVgm <- fit.StVariogram(varST2, smetricVgm)
attr(smetricVgm, "optim")$value
#productsum
prodSumVgm <- vgmST("productSum",
                space = vgm(0.70, "Sph", 60000, 0.2),
                time = vgm(0.70, "Sph", 2, 0.2), k = 2)

prodSumVgm <- fit.StVariogram(varST2, prodSumVgm)
attr(prodSumVgm, "optim")$value

#viz
plot(varST2, list(smetricVgm, prodSumVgm, metricVgm), diff = T, col = color_pal)


## ------------------------------------------------------------------------------------------
#metric
metricVgm <- vgmST(stModel = "metric",
joint = vgm(0.10, "Exp", 20000, nugget = 0.10),
sill = 0.10,
stAni = 20000)
metricVgm <- fit.StVariogram(varST3, metricVgm)
attr(metricVgm, "optim")$value
#sum metric
smetricVgm <- vgmST(stModel = "sumMetric",
                space = vgm(psill=0.10,"Exp", range= 20000, nugget=0.10),
                time = vgm(psill= 0.10,"Exp", range= 2, nugget=0.10),
                joint = vgm(0.10, "Exp", 20000, nugget = 0.10),
                sill = 0.10,
                stAni = 20000)
smetricVgm <- fit.StVariogram(varST3, smetricVgm)
attr(smetricVgm, "optim")$value
#productsum
prodSumVgm <- vgmST("productSum",
                space = vgm(0.10, "Exp", 20000, 0.10),
```

```
            time = vgm(0.10, "Exp", 2, 0.05), k = 2)

prodSumVgm <- fit.StVariogram(varST3, prodSumVgm)
attr(prodSumVgm, "optim")$value

#viz
plot(varST3, list(smetricVgm, prodSumVgm, metricVgm), diff = T, col = color_pal)


## ------------------------------------------------------------------------------------------
#prediction
testset1$prediction <- predict(step1, testset1)
testset1$residual <- testset1$log_value - testset1$prediction

testset2$prediction <- predict(step2, testset2)
testset2$residual <- testset2$log_value - testset2$prediction

testset3$prediction <- predict(step3, testset3)
testset3$residual <- testset3$log_value - testset3$prediction

testset4$prediction <- predict(step4, testset4)
testset4$residual <- testset4$log_value - testset4$prediction

#all the residuals for the whole month
ggplot(testset1) + aes(x = residual) + geom_histogram(bins = nclass.Sturges(testset1$residual))
#residuals for each day
ggplot(testset4) + aes(x = residual) + geom_histogram(bins = sqrt(100)) + geom_vline(xintercept = 0,
col = "red") + facet_wrap(~date, nrow = 3)


## ------------------------------------------------------------------------------------------
#RMSE
round(sqrt(mean(testset1$residual^2)),2)
round(sqrt(mean(testset2$residual^2)),2)
round(sqrt(mean(testset3$residual^2)),2)
round(sqrt(mean(testset4$residual^2)),2)

#average bias
round(mean(testset1$residual),2)
round(mean(testset2$residual),2)
round(mean(testset3$residual),2)
round(mean(testset4$residual),2)

#observed vs predicted
pred_test(testset1)
pred_test(testset2)
pred_test(testset3)
pred_test(testset4)
```

```
## ------------------------------------------------------------------------------------------
sample <- subset(testset3, select = c(station, x, y, date, log_value, residual, prediction, day))

freq <- data.frame(table(sample$station)) #frequency of each station

#stations used for performance
st_miss <- freq$Var1[freq$Freq >0 & freq$Freq <31]
sample_miss <- subset(sample, station %in% st_miss)

#stations used for ST kriging
st <- freq$Var1[freq$Freq == 31]
data <- subset(sample, (station %in% st))

#build space-time object from testset
ST_test <- STFDF(sp = SpatialPoints(unique(data[,c("x","y")]), proj4string = CRS(prjs[pos, "prj4"])),
time = unique(data$date), data = data)

#grid
predGrid <- STF(grd_pts_in, ST_test@time)

#ST kriging prediction
krig1 <- krigeST(formula = residual ~ 1, data = ST_test,
                 newdata = predGrid,
                 modelList = smetricVgm,
                 computeVar = T)

#ST kriging prediction error
stplot(krig1, main ="", col.regions = rev(color_pal4))
#ST kriging prediction error
krig1$se <- sqrt(krig1$var1.var)
stplot(krig1[, , "se"],main = "")


## ------------------------------------------------------------------------------------------
test <- dist(krig1, sample_miss)
test <-data.frame(test)
merge <- cbind(sample_miss, test) #add X, Y and mindist to sample_miss


## ------------------------------------------------------------------------------------------
#add time to merge
time <- data.frame(unique(testset1$date), c(1:31))
colnames(time) <- c("date", "t")
merge <- merge(merge, time, by = "date", all.x = T)

#add time to data_merge
```

```
time.vec <- rep(c(1:31), each = 1230) #1230 points in the grid
data_merge <- cbind(krig1@data$var1.pred, krig1@sp$X, krig1@sp$Y, time.vec)
data_merge <- data.frame(data_merge)
colnames(data_merge) <- c("pred", "X", "Y","t")

#final merge by coordinates and time
merge2 <- merge(merge, data_merge, by = c("X","Y", "t"), all.x = T)
merge2$prediction_final <- merge2$prediction + merge2$pred
merge2$residual_final <- merge2$log_value - merge2$prediction_final

#viz
ggplot(merge2) + aes(x = date) +
  geom_line(aes(y = log_value, colour = "observed")) +
  geom_line(aes(y = prediction, colour = "regression prediction")) +
  geom_line(aes(y = prediction_final, colour = "prediction via ST kriging")) +
  facet_wrap(~station) +
  labs(y = "Y", x = "", title = "") +
  scale_color_manual(name = "", values = c("observed" = "black", "regression prediction" =
"red","prediction via ST kriging" = "blue" ))

#RMSE regression
round(sqrt(mean(merge2$residual^2)),2)
#RMSE ST kriging
round(sqrt(mean(merge2$residual_final^2)),2)



## ------------------------------------------------------------------------------------------
#create maps with points
coords2 <- unique(data[,c("x","y")])
coords2$type <- "sample point"
coords3 <- unique(sample_miss[,c("x","y")])
coords3$type <- "test point"
coords <- rbind(coords2, coords3)
coords <- conv_spatial(coords)

tm_shape(polyg) + tm_fill(col = "grey92") + tm_shape(coords) + tm_symbols(col = "type", shape = 3,
size = 0.5, palette = c("sample point" = "black", "test point" = "blue"))



## ------------------------------------------------------------------------------------------
hist(merge2$residual_final, main = "", xlab = "final residuals")
hist(merge2$residual, main = "", xlab = "regression residuals")
```

```
## ------------------------------------------------------------------------------------------
library(tmap)
library(latex2exp) #latex
library(ggplot2) #graphs
library(rgdal) #read shp files
library(gstat) #variogram
library(wesanderson)


## ------------------------------------------------------------------------------------------
source("function/functions.R")


## ------------------------------------------------------------------------------------------
theme_set(theme_bw())
layout <- theme(axis.text.y = element_blank(), axis.ticks.y = element_blank())
concent <- TeX("concentration ($\\mu$g/m3)")
concent3 <- TeX("Average concentration ($\\mu$g/m3)")
dist <- "distance (m)"


## ------------------------------------------------------------------------------------------
prjs <- make_EPSG()  #chercher le tableau des projections
pos <- grep(31370, prjs[, "code"])

#read shp files belgian cities
polyg <- readOGR("data/X/commune/Communes.shp")
polyg@proj4string <- CRS(prjs[pos, "prj4"])


## ------------------------------------------------------------------------------------------
gridK <- makegrid(polyg, cellsize = c(5000,5000))
names(gridK) <- c("X", "Y")
coordinates(gridK) <- c("X", "Y")

grd_pts <- SpatialPoints(
  coords      = gridK,
  proj4string = CRS(prjs[pos, "prj4"])
)
grd_pts_in <- grd_pts[polyg, ]
ggplot(as.data.frame(coordinates(grd_pts_in))) +
  geom_point(aes(X, Y))

gridded(grd_pts_in) <- T
plot(grd_pts_in)


## ------------------------------------------------------------------------------------------
```

```
#stations with localisations
stations_irceline <- read.csv('data/station/stations.csv', header = T, sep =';', dec =",") #stations
IRCELINE
stations_converted <- read.csv('data/station/stations_converted.csv', header= T, sep = ',', dec = ".")
#stationsconverted
stations_irceline <- rbind(stations_irceline, stations_converted)


## ------------------------------------------------------------------------------------------
stat_uniq <- data.frame(unique(stations_irceline['ab_local_code']))
stat_uniq <- merge(stations_irceline[,c("ab_local_code", "lambert_x", "lambert_y")], stat_uniq, by =
'ab_local_code', all.y = T)
colnames(stat_uniq)[which(names(stat_uniq) == "lambert_x")] <- "x"
colnames(stat_uniq)[which(names(stat_uniq) == "lambert_y")] <- "y"
stat_uniq <- conv_spatial(stat_uniq)
stat_uniq@data$x <- stat_uniq@coords[,1]
stat_uniq@data$y <- stat_uniq@coords[,2]


## ------------------------------------------------------------------------------------------
PM10 <- read.csv('data/polluant/PM10.csv', header = T)
PM10 <- update(PM10)
PM10 <- subset(PM10, value != -9999) #remove missing values
#take the mean with respect to the station and the full date
PM10 <- aggregate(formula = value ~ date + station, data = PM10, FUN = "mean")
#!! il y a des 0 en moyennes, vérifier que c'est parce que c'était une valeur négatives remplacées par
0


## ------------------------------------------------------------------------------------------
#merge with spatial variables
PM10 <- merge(PM10, stat_uniq@data, by.x = "station", by.y = "ab_local_code", all.x = T)


## ------------------------------------------------------------------------------------------
#Plots data for date = 2020-12-31
#map
sub <- subset(PM10, (date == "20200101"))
sub <- conv_spatial(sub)
belg <- tm_shape(polyg) + tm_fill(col = "lightgrey") #belgium original map

#maps
belg + tm_shape(sub) + tm_symbols(col = "value", shape = 20, border.col ="white", size = 0.5, title.col
= expression(paste("", mu, "/m3")))


## ----krigeage-------------------------------------------------------------------------------
#variogramme
```

```
var.estimation <- variogram(value ~ 1, data = sub, cutoff = 100000)
ggplot(var.estimation) + aes( x = dist, y = gamma) + geom_point(size = 2) +
  labs(x = dist, y = "semivariance")

#bootstrap
bootstrap <- function(n){
  #init plot
  plot(1, type = "n", ylim = c(0, 160),
       xlim =  c(0, max(var.estimation$dist) ), xlab = dist, ylab ="semivariance")

  for (i in 1:n){
    set.seed(i)
    sub@data$value.bts <- sample(sub@data$value)
    var.est.bts <- variogram(sub@data$value.bts~1, data = sub)
    lines(var.est.bts$dist, var.est.bts$gamma, col = "lightgrey")
  }

  lines(var.estimation$dist, var.estimation$gamma, col ="blue", lwd = 3)
}
bootstrap(1000)

#Gaussian
vgm_g <- vgm(40,"Gau", 40000, 5)
model.g <- fit.variogram(var.estimation, vgm_g, fit.sills = F, fit.ranges = F)
plot(var.estimation, model.g, xlab = dist, pch = 3, col = "black")
#spheric
vgm_s <- vgm(40,"Sph", 40000, 5)
model.s <- fit.variogram(var.estimation, vgm_s, fit.sills = F, fit.ranges = F)
plot(var.estimation, model.s, xlab = dist, pch = 3, col = "black")
#exponential
vgm_e <- vgm(45,"Exp", 40000, 5)
model.e <- fit.variogram(var.estimation, vgm_e, fit.sills = F, fit.ranges = F)
plot(var.estimation, model.e, xlab = dist, pch = 3, col = "black")
#pentaspherical
vgm_p <- vgm(35,"Pen", 45000, 5)
model.p <- fit.variogram(var.estimation, vgm_p, fit.sills = F, fit.ranges = F)
plot(var.estimation, model.p, xlab = dist, pch = 3, col = "black")



## ------------------------------------------------------------------------------------------------
#Gaussian
set.seed(2)
krige.cv <- krige.cv(value ~ 1, sub, grd_pts_in, model = vgm_g, nfold = 4)
coordinates(krige.cv) <- c("x", "y")
proj4string(krige.cv) <- CRS(prjs[pos, "prj4"])
round(sqrt(mean(krige.cv$residual^2)),2)
#spheric
krige.cv <- krige.cv(value ~ 1, sub, grd_pts_in, model = vgm_s, nfold = 4)
```

```
coordinates(krige.cv) <- c("x", "y")
proj4string(krige.cv) <- CRS(prjs[pos, "prj4"])
round(sqrt(mean(krige.cv$residual^2)),2)
#exponential
krige.cv <- krige.cv(value ~ 1, sub, grd_pts_in, model = vgm_e, nfold = 4)
coordinates(krige.cv) <- c("x", "y")
proj4string(krige.cv) <- CRS(prjs[pos, "prj4"])
round(sqrt(mean(krige.cv$residual^2)),2)
#pentaspherical
krige.cv <- krige.cv(value ~ 1, sub, grd_pts_in, model = vgm_p, nfold = 4)
coordinates(krige.cv) <- c("x", "y")
proj4string(krige.cv) <- CRS(prjs[pos, "prj4"])
round(sqrt(mean(krige.cv$residual^2)),2)


## ------------------------------------------------------------------------------------------
ko <- krige(value ~ 1, location = sub, newdata = grd_pts_in, model = vgm_g )
#pred
spplot(ko, "var1.pred", col.regions = wes_palette("Zissou1", 16, type = "continuous"))
#var
spplot(ko, "var1.var", col.regions = wes_palette("Zissou1", 16, type = "continuous")) # variance
erreur


## ------------------------------------------------------------------------------------------
# residual analysis
krige.cv$absolute.value <- abs(krige.cv$residual)
krige.cv$sign[krige.cv$residual <= 0] <- "negative"
krige.cv$sign[krige.cv$residual > 0] <- "positive"

ggplot(krige.cv@data) + aes(x = krige.cv$var1.pred , y = residual) + geom_point() +
geom_hline(yintercept = 0) + labs(x = "prediction")

belg + tm_shape(krige.cv) + tm_symbols( size = "absolute.value", col = "sign", palette =
c('#009392',"#CC3300"))


## ------------------------------------------------------------------------------------------
library(ape)
xy <- cbind(krige.cv@coords[,1], krige.cv@coords[,2])
matrice.dists <- as.matrix(dist(xy, method = "euclidean"))
matrice.dists.inv <- 1/matrice.dists
diag(matrice.dists.inv) <- 0
Moran <- Moran.I(krige.cv$residual, matrice.dists.inv)
```

```
##librairies
import pandas as pd
import numpy as np

# split with respect to the pollutant
df = pd.read_csv('data.csv',sep =';', header = None, low_memory = False)
df1 = df[1:87673] # PM10
df2 = df[87674: 175346] # PM2.5
df2 = df2.iloc[:,:-4]
df3 = df[175347:263019] # O3
df3 = df3.iloc[:,:-44]
df4 = df[263020: len(df)] # NO2

df1 = df1.reset_index(drop = True)
df2 = df2.reset_index(drop = True)
df3 = df3.reset_index(drop = True)
df4 = df4.reset_index(drop = True)

# list of station
stations = pd.read_csv('station.csv', sep = ";")

# rename variables
var = ['date', 'hour']
df1.columns = var + list(stations.ST_PM10.values)
df2.columns = var + list(stations.ST_PM25.values[:-4])
df3.columns = var + list(stations.ST_O3.values[:-44])
df4.columns = var + list(stations.ST_NO2.values)

#convert to numeric all stations
for i in range(2,len(df1.columns)):
    df1.iloc[:,i] = pd.to_numeric(df1.iloc[:,i])
for i in range(2,len(df2.columns)):
    df2.iloc[:,i] = pd.to_numeric(df2.iloc[:,i])
for i in range(2,len(df3.columns)):
    df3.iloc[:,i] = pd.to_numeric(df3.iloc[:,i])
for i in range(2,len(df4.columns)):
    df4.iloc[:,i] = pd.to_numeric(df4.iloc[:,i])

# missing values
miss1 = round((df1.isin([-9999]).sum().sum()/ ((len(df1.columns)-2)*len(df1))*100),1)#27.5%
miss2 = round((df2.isin([-9999]).sum().sum()/ ((len(df2.columns)-2)*len(df2))*100),1)#35.6%
miss3 = round((df3.isin([-9999]).sum().sum()/ ((len(df3.columns)-2)*len(df3))*100),1)#17.6%
miss4 = round((df4.isin([-9999]).sum().sum()/ ((len(df4.columns)-2)*len(df4))*100),1)#26.1%

# divide by 10 PM25
df2[["40SZ01", "40SZ02", "41B011", "41MEU1", "41N043", "45R502", "45R510"]] = df2[["40SZ01",
"40SZ02", "41B011", "41MEU1", "41N043", "45R502", "45R510"]].div(10)
```

```
# dataset reshape
df1_final = df1.melt(id_vars = ['date', 'hour'], var_name = 'station', value_name ='value')
df2_final = df2.melt(id_vars = ['date', 'hour'], var_name = 'station', value_name ='value')
df3_final = df3.melt(id_vars = ['date', 'hour'], var_name = 'station', value_name ='value')
df4_final = df4.melt(id_vars = ['date', 'hour'], var_name = 'station', value_name ='value')

# negative values
neg1 = sum((df1_final['value']<0) & (df1_final['value'] != -9999))/len(df1_final) # less than 1%
neg2 = sum((df2_final['value']<0) & (df2_final['value'] != -9999) & (df2_final['value'] != -999.9))/
len(df2_final) #less than 1 %
neg3 = sum((df3_final['value']<0) & (df3_final['value'] != -9999)) / len(df3_final) # 0.3%
neg4 = sum((df4_final['value']<0) & (df4_final['value'] != -9999)) / len(df4_final) # less than 1 %
print(neg1, neg2, neg3, neg4)

# replace neg values by 0
df1_final.loc[(df1_final['value']<0) & (df1_final['value'] != -9999) , 'value'] = 0
df2_final.loc[(df2_final['value']<0) & (df2_final['value'] != -9999) & (df2_final['value'] != -999.9) ,
'value'] = 0
df3_final.loc[(df3_final['value']<0) & (df3_final['value'] != -9999), 'value'] = 0
df4_final.loc[(df4_final['value']<0) & (df4_final['value'] != -9999), 'value'] = 0

# reset index
df1_final = df1_final.reset_index(drop = True)
df2_final = df2_final.reset_index(drop = True)
df3_final = df3_final.reset_index(drop = True)
df4_final = df4_final.reset_index(drop = True)

#export
df1_final.to_csv('C:\\Users\\lucas\\Documents\\BLOC 2\\Q2\\Mémoire\\mémoire
UCLouvain\\Mémoire_analyse\\PM10.csv', index = False)
df2_final.to_csv('C:\\Users\\lucas\\Documents\\BLOC 2\\Q2\\Mémoire\\mémoire
UCLouvain\\Mémoire_analyse\\PM25.csv', index = False)
df3_final.to_csv('C:\\Users\\lucas\\Documents\\BLOC 2\\Q2\\Mémoire\\mémoire
UCLouvain\\Mémoire_analyse\\O3.csv', index = False)
df4_final.to_csv('C:\\Users\\lucas\\Documents\\BLOC 2\\Q2\\Mémoire\\mémoire
UCLouvain\\Mémoire_analyse\\NO2.csv', index = False)
```

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
Faculté des sciences
Place des Sciences, 2 bte L6.06.01, 1348 Louvain-la-Neuve, Belgique | www.uclouvain.be/sc