Master thesis on Intelligent Interactive Systems

Universitat Pompeu Fabra

A novel message passing approach to spatial air quality prediction in urban areas

Sergio Calo Oliveira

Supervisor: Filippo Bistaffa (IIIA-CSIC)

Co-Supervisor: Vicenç Gómez, Anders Jonsson

July 2022

Master thesis on Intelligent Interactive Systems

Universitat Pompeu Fabra

A novel message passing approach to spatial air quality prediction in urban areas

Sergio Calo Oliveira

Supervisor: Filippo Bistaffa (IIIA-CSIC)

Co-Supervisor: Vicenç Gómez, Anders Jonsson

July 2022

Contents

Acknowledgement

I would like to express my sincere gratitude to my supervisors at UPF, Anders and Vicenç, for their time and advice. Also to my supervisor at IIIA-CSIC, Filippo, for the trust placed on me and for what I have learned in these months of work together. Thanks also to the researcher Pau Ferrer-Cid for his generosity in sharing his work with us.

Abstract

Air pollution in our cities is a very significant cause of death and worsening of the quality of life in current times. Knowing in depth this phenomenon and its details and building tools that help us to mitigate its effects can be key to a better habitability of present and future urban environments. This is precisely the purpose of this work. Using specific data from the city of Barcelona, the problem of spatial prediction of air quality at the urban level has been addressed. The objective is to obtain a resolution at street level starting from the known values of a series of detectors, dispersed throughout the city. Certain particularities of this case study have forced us to carry out novel research in this aspect. The contributions of this work are the following. Firstly, a formalization of the problem based on Markov Random Fields has been carried out, following the recommendations of recent works in this sense, in order to favor the unification of the theoretical frameworks of the Graph Signal Processing field. After this, a novel algorithm has been developed for the resolution of the problem. This algorithm is based on a message passing scheme between nodes. In the proposed method, this algorithm is combined with a Graph Neural Network that refines the obtained result for a better approximation.

Keywords: Air quality prediction, Graph signal reconstruction, Graph Neural Networks.

Chapter 1

Introduction

In today's world, climate change, emissions of polluting gases and other environmental problems are at the center of the agendas of all national and international institutions. These challenges are cross-cutting for the entire population and can be studied from very different points of view. One of the open fronts within this problem is air quality. Air quality refers to the condition of the air in terms of its pollutants, while air pollution is defined as "contamination of the indoor or outdoor environment by any chemical, physical or biological agent that modifies the natural characteristics of the atmosphere" [1]. Specifically in urban environments, due to the number of different pollutant sources we can find (cars, factories, homes, etc.) the danger of this phenomenon becomes more pronounced [2]. According to [3] more than 90% of the global population in 2019 lived in areas where concentrations exceeded 10 µg/m3, the air quality guideline level proposed in [4].

1.1 Motivation and objectives

Given the impact that air quality has on health, it seems clear that knowing the levels of air pollution to which we are exposed daily could be crucial to combat, as far as possible, these harmful effects. In the case of the city of Barcelona, there are a limited number of detectors that provide air quality measurements in very specific points of the city. Taking this into account, we wondered how we could use Artificial

Intelligence methods for the spatial prediction of pollutant concentration from the available detectors.

Seeing that the problem posed is based on a graph structure, we decided to explore the existing techniques on the literature. We found out that there was no techniques based on Graph Neural Networks for this particular problem. Graph Neural Networks (GNNs) [5] are a neural network architecture built to work with data in the form of a graph. In the recent years, the field of GNNs is experiencing exponential growth in terms of published articles and general interest [6]. Its application is resulting in several breakthroughs in many different fields. To cite just a few examples, we have applications in Physics [7], in combinatorial optimization problems [8], in Computer Vision [9] or in traffic forecasting [10]. For this reason, we believe it is necessary to explore these techniques in the framework of a problem of such relevance as the one proposed.

In this case, our initial goal will not be to perfectly solve such a complex problem as urban air pollution prediction, but to explore methods based on GNNs and adapt them to our particular problem. This could potentially lead to an effiecient and simple method to predict the urban air pollutants concentrations at street scale from a set of detectors distributed around the city approximately, for a given density of detectors.

1.2 Related work

Numerous articles can be found in the literature that address the problem of air pollutant prediction, mainly in a spatio-temporal or temporal manner. In some papers, the problem is treated as a time series prediction where the aim is only the prediction of the temporal dimension of a given detector [11, 12, 13]. In these works, well-known Deep Learning techniques are applied to deal with time series such as Long Short-Term Memory (LSTM) recurrent neural networks. Other methods introduce meteorological information to the Deep Learning model in order to improve its performance [14]. In our case, these works will not be of particular relevance since

in our problem we do not deal with a temporal dimension, but only with spatial information.

Deep spatiotemporal learning based on an air quality forecast methods has been applied to analyse spatial and temporal correlations in air pollutants concentrations. These methods usually combine two types of Deep Learning architectures, LSTM for the time dimension and Convolutional Neural Networks (CNN) for the spatial domain [15, 16]. However, this combination of LSTM and CNN is not the only approach to tackle this problem. Others works are based on a stacked autoencoder model, which is used to extract representative spatiotemporal air quality features, and it is trained in a greedy layer-wise manner [17]. Closer to our approach, there are some papers already investigating the use of GNNs to work with these spatiotemporal correlations [18, 19]. In the aforementioned works, the spatial dependencies between the detectors are represented by a graph structure that is modeled by a GNN, with the objective of improving the temporal predictions of the detectors in future times, so the problem does not fit the one we are trying to solve in the present work either.

Analyzing the case of the city of Barcelona, it can be confirmed that there is an interest in the study of air quality. Several recent works analyze different aspects of this phenomenon on health [20, 21] or the urban mobility impact [22]. In addition, the authors Mota-Bertran et. al. [23] conducted a compositional analyis of air pollutants in Barcelona.

1.3 Structure of the Report

The document is arranged as follows. In the current chapter the motivation of the work, along with an overview of the topic are presented. In Chapter 2 the theoretical background needed to build the following steps is presented. This background covers Graph theory, Graph Neural Networks and Probabilistic Graphical Networks. In the same chapter, we list some existing methods for Graph Signal Reconstruction. In Chapter 3 it is discussed the formalization of the problem and the theoretical

framework to follow is developed. In addition, as a key point, the novel method proposed in this work is developed in this chapter. Next, Chapter 4 presents the experimental method together with the results obtained in the experiments, as well as the metrics and data used. Finally, Chapter 5 closes the work with a discussion of the results obtained and the conclusions reached.

Chapter 2

Background

2.1 Graph theory preliminaries

A graph, in general, is nothing more than a data structure that captures different elements and the relationship between them. More formally, a graph G can be defined as (V, E, W), where V is a set of vertices or nodes, E is a set of connections between the vertices and W a weight matrix that represents the weights of those connections. These connections are also called edges.

The graph can be weighted, in case the edges have different weights, or unweighted, if all these edges are equal. We can also distinguish between directed graphs, where the edges have a notion of direction (i.e., from a node A to a node B, but not vice versa), or undirected graphs, where this notion does not exist.

This structure admits different representations. One of them is the Adjacency Matrix. This is an n by n square matrix, where n is the total number of nodes in the network. The elements of this matrix are zeros except in those positions (i, j) where a connection exists between nodes i and j. In that case the value of the matrix element is 1 in the case of unweighted graphs or the value of the connection, $e_{i,j}$, in the case of weighted graphs. Furthermore, if the graph is undirected, the Adjacency matrix is symetric.

Once we know the definition of a graph, we also can define the signal of a given node as a function f, such that

$$f : v \to \mathbb{R}, v \in V$$

Given the node signal, we can then define a graph signal as the set of individual node signals in the following way

$$g = [f(v_1), f(v_2), ..., f(v_N)] \ \mid \ v_1, v_2, ..., v_N \in V$$

From what has been exposed so far in this section, we can then express that Graph Signal Processing (GSP) is an extension of the Signal Processing field which deals with signals living on graph domains. One specific family of problems within GSP is Graph Signal Reconstruction (GSR). We can find a scheme of the field of study of the problem in 1. The GSR process, generally understood, proposes the reconstruction of a complete graph signal from a series of samples extracted from it.

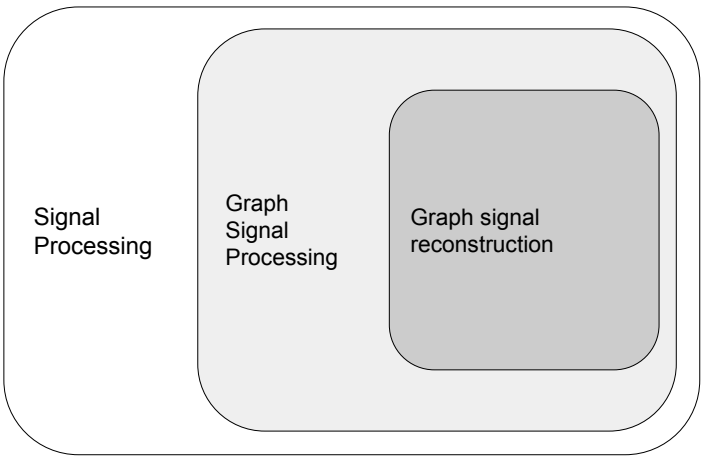


Figure 1: Visual scheme of the location of the problem to be treated within the study areas. As presented in the text, Graph Signal Reconstruction problems are a subtype within Graph Signal Processing, which in itself is included in the field of Signal Processing.

2.2 Graph Neural Networks preliminaries

As previously stated, in this research we work with a very particular type of data, i.e. graphs. These follow a very different structure than other types of data, such as tabular data or time series. This is why traditional neural network architectures or methods are not always as effective in capturing patterns when dealing with graphs. This fact motivated new research to find models that work naturally with graphs as input. Before the model was standardized in a single framework, several independent lines of research had published different approaches under different terms, such us Graph Convolutional Networks, Graph Neural Networks or Spectral Networks [24, 25, 26].

The work of Gilmer et al. [27] unified various graph neural network and graph convolutional network approaches [28]. The term used in this case is Message Passing Neural Networks (MPNNs). This was initially conceived from the field of chemistry, constitutes a framework that systematizes and encompasses a large part of the current approaches. Thus, we can say that the most widely used Graph Neural Network scheme is built on this same MPNN basis. We will base the present work on this framework as well, which will be shown in detail below.

2.2.1 Message Passing Neural Networks

Message Passing Neural Networks (MPNN) are a particular family of algorithms within the Message Passing scheme, which follows an Aggregation and Combination process. The idea behind the MPNN is conceptually simple. Each node of the graph has a hidden state which can be a real value or a vector. For each node v, we aggregate the result of applying a function on the hidden states and (in certain cases) the edges of all neighboring nodes with node v itself. We then update the hidden state of node v using the obtained message and the previous hidden state of that node.

More formally, according to the general approach of the Message Passing Neural Networks [27], hidden states h_v^t at each node in the graph are updated during the

message passing phase based on messages m_v^{t+1} through the following equations (2.1) and (2.2):

$$m_v^{t+1} = \frac{1}{n} \sum_{w \in N(v)} M_t(h_{v,w}^{t\ t}, e_{vw}) \tag{2.1}$$

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \tag{2.2}$$

We can repeat this MP algorithm for a given number of times in an iterative manner.

2.3 Theoretical framework

When it comes to Graph Signal Processing problems, numerous attempts to build an useful theoretical framework for Graph Signal Processing can be observed in recent literature. For instance, the spectral analysis of graph signals based on Fourier analysis, the same way as in one-dimensional signals or grids, is widely used in graphs for numerous applications. We can see this approach in [29] or [30], to name a few recent works.

Among the different approaches present in the literature, we will build the theoretical framework of the present work from the point of view of Probabilistic Graphical Models. As a result, our method, which is based on a message passing scheme, fits naturally into the framework. Another advantage is that we align our work with many recent papers in the field, which also take this approach. From our point of view, we believe it is necessary to find a common framework for the field to facilitate commonality between the different methods. Citing some works that try to link Graph signal theory and PGM, in [31], Zhang et. al defend the idea of an approach to graph signal theory from the probabilistic graphical models point of view and its benefits. The authors also demonstrate the direct mapping between undirected weighted graphs and GMRF. According to [32] a probabilistic interpretation allows us to view graph signal sampling theory as a model based method. Other works that solve Graph Signal problems under this framework include methods based on

the common graph smoothness assumption formulated in a Gaussian random field model [33], or an approach to semi-supervised learning based on a Random fields [34], among others.

2.3.1 Probabilistic graphical models

We can define Probabilistic Graphical Models (PGM) as a graph-based representation for encoding a distribution over a multidimensional space. In this graphical representation the nodes correspond to the variables in our domain, and the edges correspond to direct probabilistic conditional interactions between them. [35].

2.3.2 Markov Random Fields

A Random Field is a family of Probabilistic Graphical Models, where a factored graphical representation of a joint probability distribution over a set of random variables X is built. It consists of an undirected graph $G = (V, E)$ in which the nodes (also called vertices) V represent random variables X and the edges E encode conditional independence relationships between them [36]. A Random Field is called a Markov Random Field (MRF) if, and only if, given any two nodes v_i and v_j with no direct edge between them, they are conditionally independent given the rest of the graph. This property is called the pairwise Markov property. On the other hand, A Random Field is called a Gaussian Random Field (GRF) if, and only if, the random variables X follow a gaussian probability distribution. In the same way, when a Random Field meets these two requirements, it is defined as a Gaussian MArkov Random Field (GMRF). A visual scheme of the hierarchy of the PGM can be seen in Figure 2.

2.4 Existing methods

In this section we will present a set of existing methods in the literature. Each of this methods tries to address the presented problem of Graph Signal Reconstruction through different approaches. A crucial aspect of any of these methods is whether or not it requires training data to build a model. When we perform an adjustment of

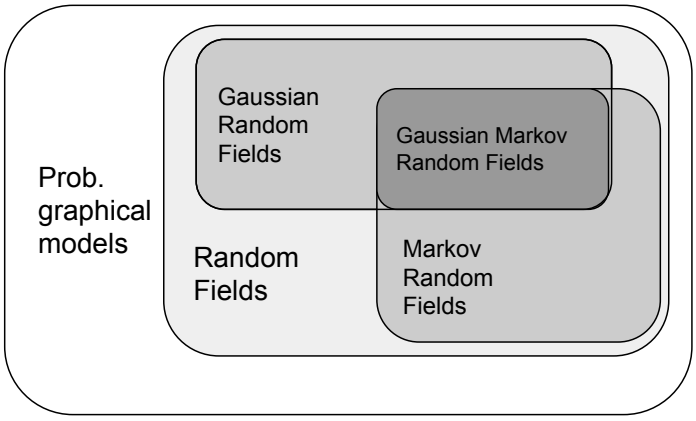


Figure 2: Visual scheme of the

parameters based on training data, we can define this as semi-supervised methods. On the other hand, other methods are not based on training data, but are built on a set of rules or assumptions. Those methods are often called knowledge-driven, model-driven or non-data-driven methods.

Due to the variability in the literature of the field, beyond these categories based on data, we can find very different formalizations and methods. We will address this complexity by framing this methods in three different families. First, we will present those methods that make use of shifts and kernel operators. Secondly, methods based on a Belief Propagation schema, also based on Message Passing but with specific characteristics. Finally, graph regularization methods, based on solving a set of equations to maximize a regularization term.

Kernel based methods on graphs are nothing more than a generalization of the well-known kernel methods for other data geometries (such as images) to graphs. Kondor and Lafferty [37] propose a family of kernels for graphs based on the heat equation, which they call Diffusion kernels. Zhu et. al [34] proposes this type of methods to solve a Graph Signal Reconstruction problem. In this work we will implement the mentioned method, solving by means of a kernel the heat equation with sources. These sources correspond to the nodes with presence of detectors. This method will henceforth be referred to as Heat Diffusion Kernel (HDK). This method is going to

be non-data-driven.

Belkin et al. [38] propose to employ regularization in the graph based on solving a system of equations to solve the proposed problem. One of the algorithms developed in the paper is called Interpolated Regularization (IR). This algorithm does not consist of any parameters, and it is based on finding a function of maximum smoothness.

Belief Propagation was first proposed by Judea Pearl in 1982 [39] as an algorithm for obtaining the marginals of a joint distribution via local message passing between nodes in a graph. Gaussian belief propagation (GBP) [40] is a special case in which the conditional dependent variables are related by Gaussian functions. The aim here is to infer Gaussian-distributed marginals.

A comparison will be made between the new methods proposed in this paper and the following existing methods mentioned above: Heat Diffusion Kernel (HDK), Interpolated Regularization (IR) and Gaussian belief propagation (GBP).

Chapter 3

A novel approach to urban air quallity prediction

3.1 Nature of our data

Some of the works that have been discussed in the Section *Related work* have complete histories of a very small number of nodes and seek to make temporal predictions about them. Other works use datasets with a country level scale and use all the detectors within that area to establish spatio-temporal correlations between the different detectors, or for their calibration. However, in the present work we seek to attack the problem of working at an urban scale with a street level resolution. The objective is then to obtain a graph containing a node-level resolution of pollution concentrations in real time. The starting point we have in the data are these real-time values in low resolution (for a very limited number of nodes). In addition, we have two samples of average data from previous years (in particular for years 2018 and 2019) with this high (node-level) resolution. This nature of the data, which differs from the usual and where real-time data without resolution and past samples without strong temporal correlation are combined largely decides the approach to be executed. For this reason, in the present work a particular and novel approach to the problem is developed, as well as a novel method for its resolution is proposed.

3.2 Framing our problem

Given the PGM framework and the definitions we showed in the previous section, in this section we will try now to translate our particular problem to that specific theoretical framework. We will also discuss the particular properties of the problem we will solve under the proposed framework.

First of all, we will then model our Dataset in the form of a MRF. This MRF will be a graph with two different types of nodes. On the one hand we have *variable* nodes, that correspond to the signals we want to estimate, ie. the positions of the map where we want to predict the air pollution concentration. On the other hand, connected to those positions where we have a detector, we have the *evidence* nodes Y, of which the values are the messures given by those detectors. The graph signal will live then on the *variable* nodes, and the *evidence* nodes will be used as a very useful tool to propose our method in a more clear way. A visual representation of this MRF can be seen in Figure 3

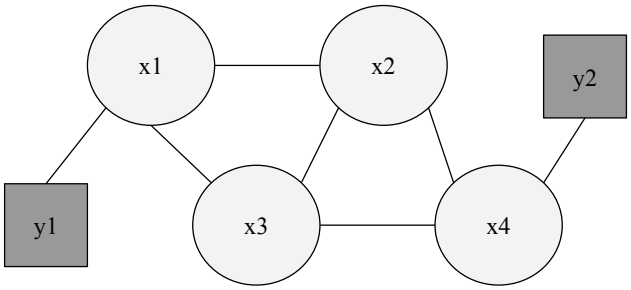


Figure 3: Schematic of the modeling of the problem as MRF. The node variables $\{x1, x2..., xn\}$ correspond to the nodes of the network. On the other hand, the evidence nodes $\{y1, y2..., yd\}$, represented as a square in the diagram, correspond to the evidence obtained from the detectors. For this reason, the value of d will be equal to the number of detectors.

The problem we propose to solve under the Markov Random Fields formalization is then a Maximum a Posteriori (MAP) inference. For a better understanding of MAP

inference, we can start by recalling Bayes' theorem. This theorem provides a way to calculate the conditional probability of an observable given another observable. Bayes' theorem is expressed as follows:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{3.1}$$

where P(B) acts as a normalization quantity. Since in our case we are not trying to obtain the probability, but optimize the given quantity, we can ignore this term for simplicity. The new Bayes' theorem is then

$$P(A|B) = P(B|A)P(A) \tag{3.2}$$

In our particular case, we can asume A as X, the variables, and B as Y, the available information (detectors), resulting in the equation

$$P(X|Y) = P(Y|X)P(X) \tag{3.3}$$

The left side of the equation, $P(X|Y)$, corresponds to the so called Posterior probability, and it can be understood as our belief about X after seeing the available information Y. This Posterior probability can also be expressed in terms of the joint probability $P(X, Y)$. Applying the chain rule for random variables we have $P(X, Y) = P(X|Y)P(Y)$. Rearranging the equation we can define the Posterior probability as $P(X|Y) = P(X, Y)/P(Y)$. In our problem,the inference proceeds by first defining a Probabilistic Graphical Model that describes the relationships between the data and the variables to define this joint probability $P(X, Y)$.

In view of the aforementioned, a MAP inference then aims to obtain the most probable configuration of the values of the X variables given the data Y available to us:

$$X_{MAP} = MAP(X|Y) = argmax_X P(X|Y) \tag{3.4}$$

Under this formalization, we will try to estimate the most probable signal X_{MAP},

given the meassurments of the detectors Y, and the joint probability $P(X, Y)$ given by our MRF. Given the mentioned ingredients we have defined the problem, solving a MAP inference, but not yet the solution. The solution proposed in this work is as follows. MRF can also be defined as energy based models, where an energy function is defined over a subset X_i of X as $E(X_i)$. In this case, finding the most likely variable configuration is equivalent to minimizing the sum of energies along the graph [41].

$$X_{MAP} = argmin_X \sum_i E(X_i) \tag{3.5}$$

Intuitively, given the nature of problem we want to address, we want unknown variables that are nearby in the graph to have similar values. This assumption is widely used in the graph signal processing literature under the name of "homophily assumption", and it is about assuming a smooth signal over the graph. In the present case, the contamination at two points that are very close to each other will be with high probability very similar, due to the continuous character of the concentrations in space. It is true that there may be more or less abrupt changes due, for instance, to puntual pollution sources or discontinuities in the terrain. We are therefore aware that this approach may have limitations. This motivates the choice of the quadratic energy function, also used for a similar type of problem by Zhu et al. [34]. if we define the mentioned subsets X_i as pairs of connected nodes, this energy function takes the form

$$E(x_i, x_j) = \frac{1}{2}(f(x_i) - f(x_j))^2 \tag{3.6}$$

being $f(x_i)$ the signal value at the node x_i. The idea here is then to try to minimize the proposed energy function, and to study if doing so would result in a good reconstruction of the signal in the graph. The proposed energy minimization method is presented in Chapter 3.

3.2.1 Our method: Mean aggregation message passing

In the present work, we have built on this general scheme to construct a non-parametric Message Passing algorithm that aggregates and combines the values of the hidden states, which are now simplified as the value of the contamination at the

node. We will call this algorithm Mean Aggregation Message Passing (MAMP). This method will be based on what has been described in Section 2.2.1. For simplicity and clarity we will keep the original notation h_v^t as the air quality value of node v at iteration t, which would correspond to our *variable* nodes values, X. The functions M_t and U_t that we are going to be using are fundamentally arithmetic averages between the inputs. We choose this functions motivated by one key assumption. This assumption we make in our problem is that air pollution between two very close points in the graph will have a high probability of being similar, due to the nature of the concentrations. Therefore in the graph two nodes that are connected will tend to have similar values. This is also called in the literature as Homophily assumption. A graph that satisfies this rule will host a smooth graph signal, and we will try to exploit this property in our case to perform the reconstruction. Based on the MPNN schema (Equations 2.1 and 2.2) the resulting equations are therefore (3.7) and (3.8):

$$m_v^{t+1} = \frac{1}{n} \sum_{w \in N(v)} h_w^t \tag{3.7}$$

$$h_v^{t+1} = \frac{1}{2}(h_v^t + m_v^{t+1}) \tag{3.8}$$

where n is the size of $N(v)$.

Due to the characteristics of our problem, and given its formalization, we must add a new type of message that occurs not between the *variable* nodes X, but between them and the *evidences* nodes Y. This new message, however, is very simple, and is only intended to keep the node state constant where the evidence allows us to do so. The node status update is therefore the one presented in equation 3.5.

$$h_v^{t+1} = y_u \tag{3.9}$$

Finally, it is necessary for the h_v^t values of the unknown nodes to be initialized to some value. We will study in detail the impact of this initial value at Section 4.6.

As can be seen, no training data is needed for this algorithm, since it does not have any trainable parameters. The training data will be used to adjust the GNN models that will be presented in Section 3.2.3.

3.2.2 Taking edges into account

Under the scheme just mentioned, it can be seen that our method is completely oblivious to the weights of the connections between nodes, all of them having the same weight in the aggregation of the hidden states of neighboring nodes. Since we know this information and it could be useful for the performance of the algorithm, we will compare the method already presented with a new version that will take these weights into account.

This variant will be as follows. The aggregation step will change its function from an arithmetic mean (3.3) to a weighted mean (3.6), where the weighting parameters depend on the weights of the edges. For example, they could be its inverse, or the root of its inverse. Different options will be compared.

$$m_v^{t+1} = \frac{\sum_{w \in N(v)} h_w^t \cdot e_{vw}}{\sum_{w \in N(v)} e_{vw}} \tag{3.10}$$

3.2.3 Introducing GNNs into our method

In addition to the above, a last variant of the method will be developed using a Graph Neural Network (GNN) at the end of the method. The objective of this GNN will be to refine the result obtained in the first step, the MAMP algorithm. The hypothesis behind this decision is that the GNN, thanks to its ability to adapt the weights to optimize the result, will be able to achieve higher expressivity than the MAMP algorithm alone. In order to prove that, we will train various GNN architectures and compare the best performance with the one achieved by other methods. Details about the GNN training process are discussed in Section "GNN training". Figure 4 shows the whole pipeline for the algorithm using presented in this section.

Algorithm 1 Mean aggregation message passing

Input: S', A, mask, ϵ, GNN ▷ S': input signal, A:Adj. matrix, mask: binary sampling vector, ϵ: convergence condition, GNN: Pretrained Graph Neural Network model (optional)

Output: S ▷ S: reconstructed signal

$improve \leftarrow \inf$
$loss_{(t)}, loss_{(t-1)} \leftarrow \inf, \inf$
$S'[mask] \leftarrow initial\_value$ ▷ Initialize the unknown node values
$S \leftarrow S'$
while $improve \geq \epsilon$ **do**
  for $node$ in S **do**
    $N_{node} \leftarrow$ neighbors of $node$
    $m \leftarrow aggregate(N_{node}, A)$ ▷ Eq. 3.3 or 3.6
    $h \leftarrow combine(node, m)$ ▷ Eq. 3.4
    $node \leftarrow h$
  end for
  $S[mask] \leftarrow S'[mask]$ ▷ Eq. 3.5
  $loss_{(t)} \leftarrow mse(S, S')$
  $improve \leftarrow loss_{(t-1)} - loss_{(t)}$ ▷ Update improvement
  $loss_{(t-1)} \leftarrow loss_{(t)}$
end while
$S \leftarrow GNN(S)$ ▷ GNN for refinement. Optional step

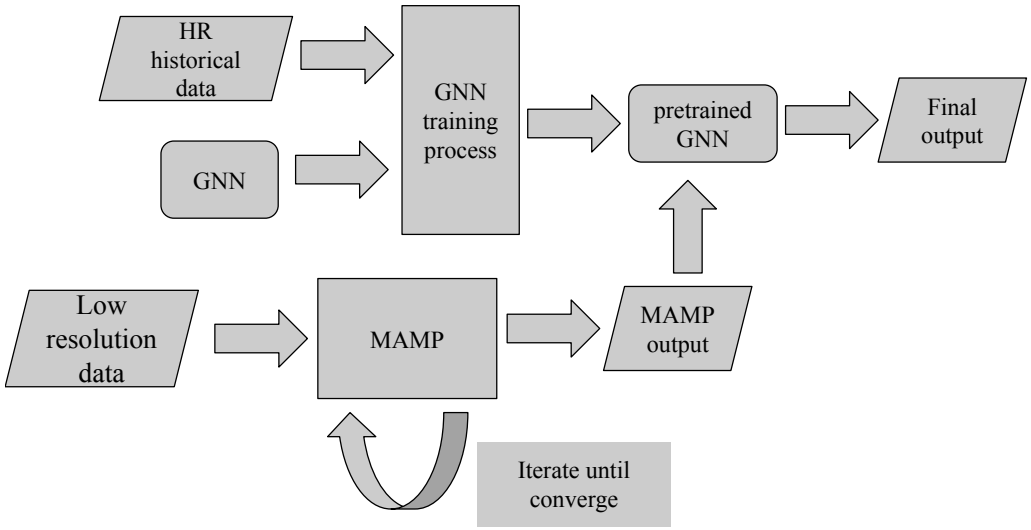


Figure 4: Pipeline of the method presented in this section. The upper part corresponds to the training of the GNN model, where *HR* stands for High resolution, meaning the historical data with street level resolution that we will use to train the GNN. This branch can be used or not. The lower branch is sufficient to understand the method proposed in Sections 3.2.1 and 3.2.2.

Chapter 4

Results

In this section, we will present first the metrics used for the evaluation of the methods. Subsequently, we will discuss in detail the experimental setup and procedure we followed to obtain the results. Those results are then presented. In section 4.3, we will compare the performance of the different methods proposed and other existing methods. In section 4.4 we will study the convergence properties of our method, while in section 4.5 we will analyze the effect of different node initializations. Finally, in section 4.6, we will present the results for the training process of the different GNN models.

4.1 Metrics

The evaluation of the models will proceed as follows. Each time we run a model from a given number of known nodes, we will calculate a series of metrics that will measure the difference between the ground truth (Y) with the result predicted by the algorithm ($\hat{Y}$) at each node of the graph. These metrics will be in our case the Mean Squared Error (MSE) and the Mean Absolute Percentage Error (MAPE).

Mean Square Error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_n - \hat{Y_n})^2 \tag{4.1}$$

Mean Absolute Percentage Error (MAPE):

$$MAPE = \frac{100}{n} \sum_{i=1}^{n} \left| \frac{Y_n - \hat{Y_n}}{Y_n} \right| \tag{4.2}$$

For the evaluation of models based on Graph Neural Networks, the procedure is the same as mentioned above. However, evaluating a model that has been trained with certain data, using the same data, is a clear bias. Therefore, a different training graph, belonging to the previous year's data set (2018), will be used for training the neural models. This allows these models to be evaluated under the same conditions and with the same data as the others. Therefore, it should be noted that none of the data used for the evaluation has been shown to the model during any previous phase.

4.2 Dataset

The dataset with which we will work has been obtained from the official database of the city council of Barcelona, Spain. This database, Open Data BCN [42], is a project designed to open data in many different fields concerning the city of Barcelona. They are accessible to all users at the link here .

In this dataset, we can find the annual average values of different pollutants for the years 2018 and 2019. Specifically, the contaminants present in the dataset are the following: PM2.5, PM10 and NO2. In our case we will keep only the pollutant PM2.5 for simplicity.

It is also necessary to mention that the concentration of pollutants is given by road section in the entire city of Barcelona. To build the graph from this point, we will take the data of road sections, intersections and positions also present in the Open Data BCN platform, accessible through the following link. With this information, we can now construct the graph and assign each road section its contaminant concentration value. In order to distribute the graph signal on the nodes instead of on the edges of the graph, we will take each value of each node as an arithmetic mean

of the concentration values of the road sections that connect to that node. A sample of the final result can be seen in Figure 5.

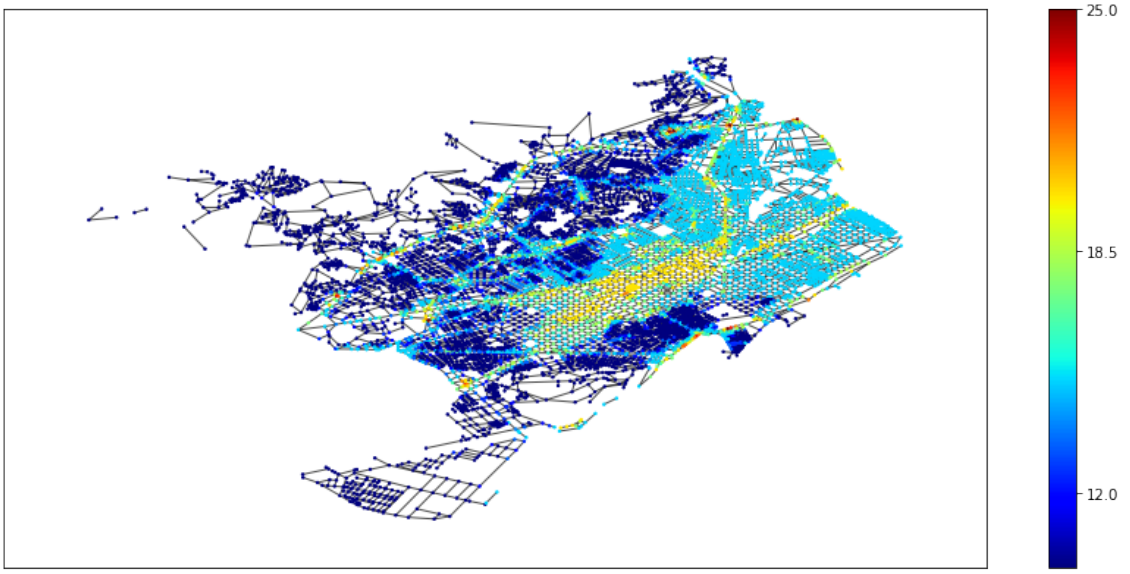


Figure 5: Sample of one of the graphs used in our dataset. Specifically for the pm2.5 pollutant in the year 2018.

4.3 Experiment setup

In order to study the performance of the presented method, we performed a series of experiments. For this purpose, the methods have been implemented and tested using the Dataset previously shown. The language used was Python. We have also made use of the following libraries. Deep graph library (DGL) [43] was used for the construction of the dataset, as well as the implementation of the Graph Neural Networks. The NetworkX library [44] was used to visualize the graphs. Finally, the Heat kernel diffusion method was implemented with the help of the pygsp library [45]. The implementations of the IR method have been carried out thanks to the code provided by the authors of the work Ferrer-Cid et al. [46], to whom we are grateful for their contribution.

The experiments will be carried out as follows: in section 4.3, the methods to be compared are run independently for different percentages of unknown nodes. We call this parameter % *of missing nodes*, and it indicates the percentage of nodes we do not know out of the total number of nodes available in the full signal. It should

be noted that, once the percentage of missing nodes is defined, the avaliable nodes within the network are randomly selected with a uniform distribution until this percentage is reached. This may create a slight variability in the results. For this reason, each experiment is performed more than once in order to know the standard deviation of its performances. This standard deviation will be present both in the graphs in the form of error bars, and in the tables. In section 4.4, we fix a given % *of missing nodes* and run the MAPE algorithm once. The aim is to analyze the behaviour of the metrics throughout the iterations when running our method, to demonstrate experimentally that a point of convergence exists. Lastly, in section 4.5, our goal is to study the effect of node value initialization on the performance of the method. To do this, we run the algorithm for different initial values, and we display the error throughout the iterations for those different values. We will thus try to prove our hypothesis that the initial value of the nodes accelerates convergence when it is closer to the average value of the graph signal.

All the code used in the experiments, as well as the dataset, are available in a GitHub repository, accessible via this link: Code here.

All the experiments, and therefore the computational time measurements, have been carried out on a machine running a processor Intel® Core™ i7-10750H CPU @ 2.60GHz × 12 and an NVIDIA GeForce RTX 2060 GPU with 6GB of memory.

4.4 Method comparison

In this section, we performance a comparison between the three proposed algorithms (MAMP, MAMP weighted by edge features and MAMP refined by a GNN) and some other existing methods in the literature. The methods we choose to perform this comparison are a Kernel method, a belief propagation method, and a laplacian regularization method. The kernel method consist on applying a heat kernel, that solves the heat equation. We did a small change on the algorithm so it uses the detectors as heat sources. The Belief Porpagation method used is Gaussian Belief Propagation. The last method we will compare is called Laplacian Regularization,

	% of missing nodes					
	50	70	80	90	95	99
GBP	-	-	-	-	-	-
IR	-	-	-	-	-	-
HDK	1.180 ± 0.042	1.974 ± 0.041	2.621 ± 0.037	3.81 ± 0.16	4.695 ± 0.084	6.52 ± 0.38
MAMP	0.791 ± 0.043	1.455 ± 0.083	1.977 ± 0.060	3.051 ± 0.080	4.04 ± 0.18	9.36 ± 0.35
MAMP + edges	0.711 ± 0.070	1.377 ± 0.091	1.940 ± 0.083	3.238 ± 0.049	4.284 ± 0.080	10.05 ± 0.056
MAMP + GNN	**0.576±0.015**	**1.054 ±0.084**	**1.388±0.037**	**2.186±0.024**	**3.02±0.14**	**4.627±0.028**

Table 1: Mean Squared Error for the different methods. "-" means the method has intrinsic problems that do not allow it to return a solution

a method extracted from [38]. All those methods are more extensively discussed in Section 2.4.

After conducting the experiments according to the experimental setup, two very similar graphs are presented. Both relate the error achieved by the methods with respect to the percentage of unknown nodes. However, graph 6 shows the MSE error (Eq. 4.1), while graph 6 shows the MAPE error (Eq. 4.2). For the edges weights we used the particular case of the inverse root of the distance between two nodes.

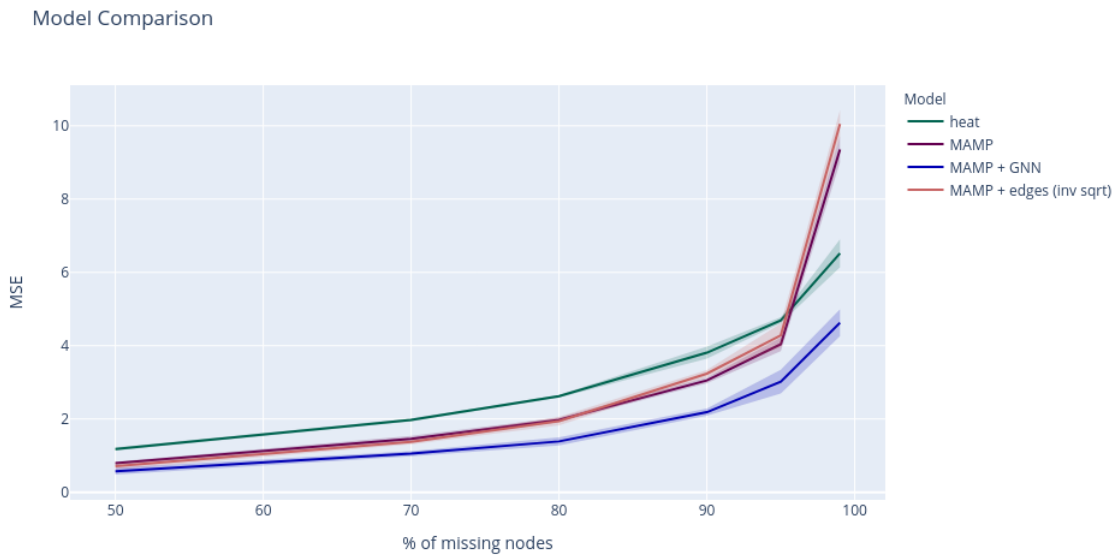


Figure 6: Method comparison. The plot shows MSE vs the % *of missing nodes*

Table 1 presents the mean MSE values and their standard deviations after running the experiments repeatedly. We can see that there are two methods (GBP and IR) that do not contain the results of the experiments. This is because these two methods have intrinsic problems in handling the proposed problem. Firstly GBP,

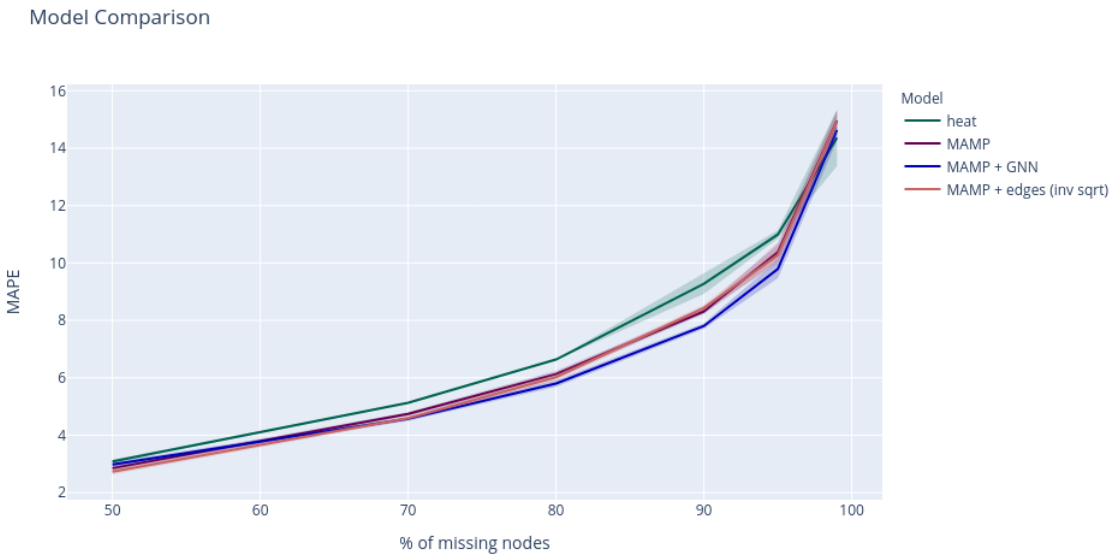


Figure 7: Method comparison. The plot shows MAPE vs the % *of missing nodes*

which stands for Gaussian Belief Propagation, does not reach a convergence point. This is because the convergence of the algorithm is not assured when there are loops in the network, as is the case. The method abbreviated as IR, Interpolated Regularization, requires calculating an inverse of the Laplacian matrix of the subset of unknown nodes. Calculating the inverse of this matrix according to the proposed method involves first calculating its determinant. Thus, when the percentage of unknown nodes is 50 percent (the lowest considered in this work), the absolute number of unknown nodes increases to 4733. In this case, there is a numerical problem when dealing with such a large matrix, since the value of the determinant diverges towards infinity very quickly. We have been able to verify experimentally that the maximum percentage of unknown nodes supported by this method for this problem is around 1% (about 100 nodes). It is important to mention that the graph we work with in this problem is much larger in terms of number of nodes and edges than other scenarios where these methods have been tested. That is why some methods have encountered difficulties specific to their architecture.

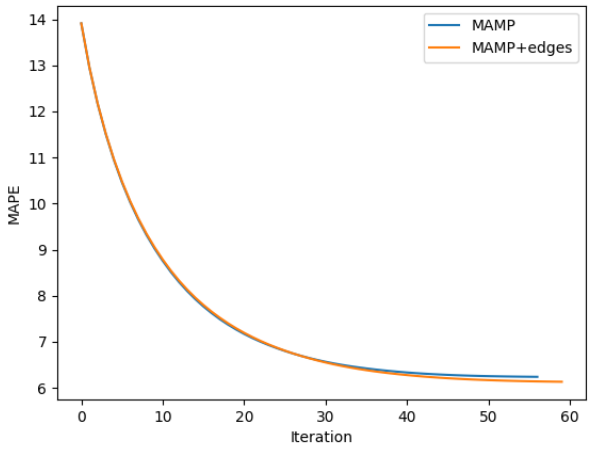
In Table 1 we can also see other methods that have been able to find a convergence point. Specifically we have the HDK method and the three methods proposed in this paper. Of the four methods, the three proposed in this work outperform the

kernel method, except for the case where 99 percent of the nodes are unknown. In this stretch, only the combination of the MAMP algorithm and a GNN achieved to outperform HDK. With respect to the standard deviations, these are not very large with respect to the absolute value obtained. These standard deviations move around the third significant figure, between 2 and 6% of the absolute value of the error. Therefore, we can conclude that both the proposed methods and the HDK comparison method are quite stable in terms of accuracy. As can be seen, the method formed by the combination of the MAMP algorithm and a refinement GNN obtains the best results for all the missing node ranges.

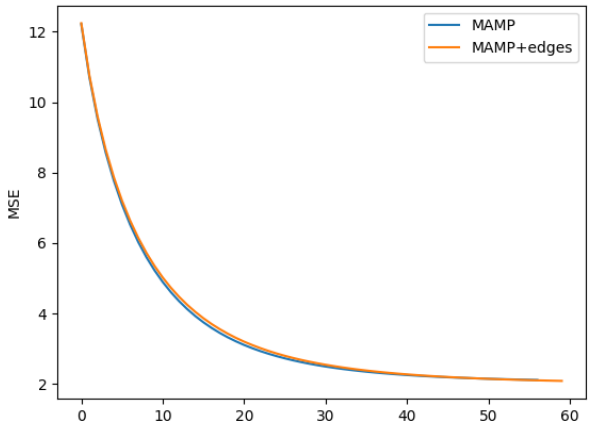
Figure 6 shows the graph for these errors shown in Table 1. On the other hand, Figure 7 shows the equivalent graph but on the Y-axis the MAPE error is shown, which gives a more easily interpretable idea of the amount of error we are making. It can be seen, for example, that when 99% of the nodes are missing, the deviation of the methods from the real value is around 14% on average per node.

4.5 Convergence

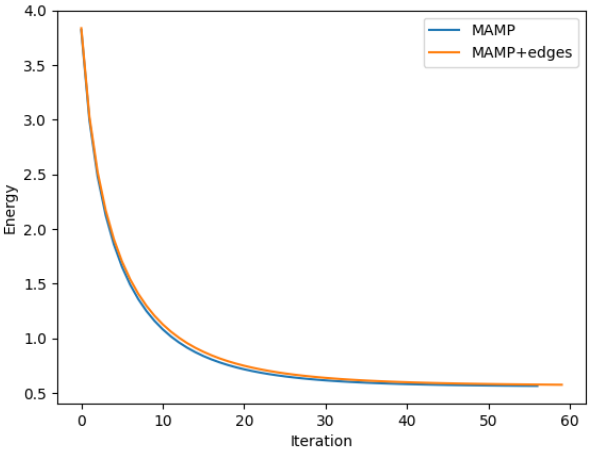
As mentioned above, the method consists of sending and receiving messages iteratively. However, we have not established theoretically the exact number of iterations to be executed for the optimal performance. If the error reached is very dependent on the number of iterations, it could represent a problem when implementing the method in new scenarios. In this section, we experimentally analyze the behavior of the algorithm through its iterative process. The aim of this experiments is to experimentally study the convergence properties of it. To do so, we will visualize how the error between the actual and predicted values varies with respect to the iterations. In particular, we run the algorithm and analyze the relationship between error and iteration, as well as the relationship between the energy (Equation 3.6) and the iteration. Figure 8 shows the results for the mentioned dependency for both errors (MAPE and MSE) and Energy.



(a) MAPE.



(b) MSE.



(c) Energy.

Figure 8: Method convergence. The plot shows differnt metrics vs iteration for a fixed % *of missing nodes* of 80%. Both MAMP and MAMP using edge features are shown.

From Figure 9, the behaviour of the curve that relates the number of iterations to converge given the percentage of missing nodes seems to rise faster than linear relation. Figure 10 represents the same experiment but scaled by the average time per iteration. This value was measured as $(5.41 \pm 0.34)s$.

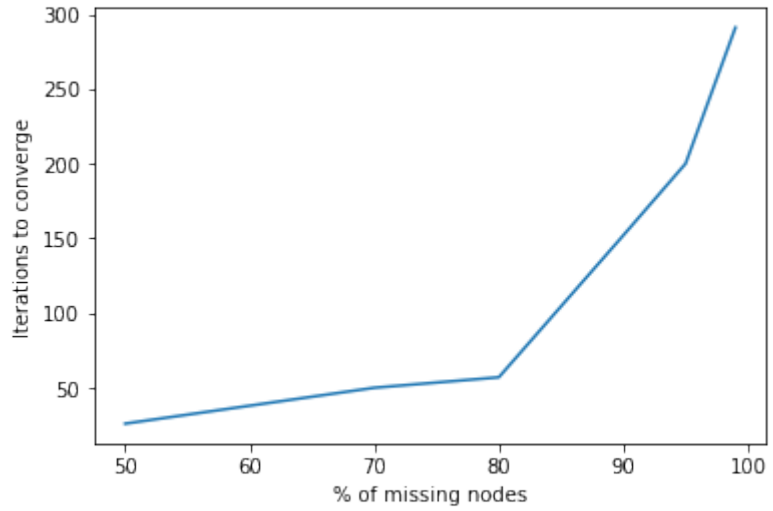


Figure 9: Speed of convergence given the % of missing nodes. The number of iterations required to converge rise with the percentage of missing nodes. The last point corresponds to 99% of missing nodes.

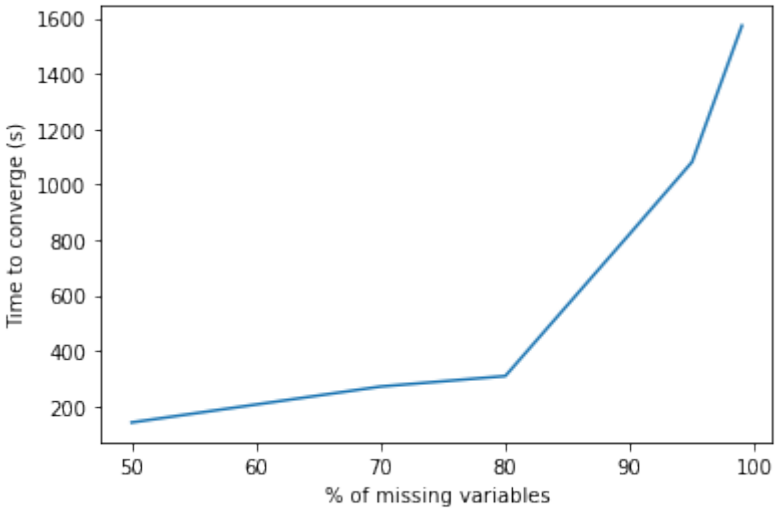


Figure 10: Time of convergence in seconds given the % of missing nodes. The last point corresponds to 99% of missing nodes.

4.6 Node initialization

As mentioned in section 3.0.2, our method requires an initialization of the Graph Signal values at the unknown nodes. We wonder, is then our method dependent on this initialization value in terms of its performance? At the same time, how does it affect its convergence?

We have tried to answer the following questions by comparing different initialization values for the same graph as input. The result obtained can be seen in the figure 11. The value 12.8628 corresponds to the mean value of the Graph Signal. The other values are chosen arbitrarily.

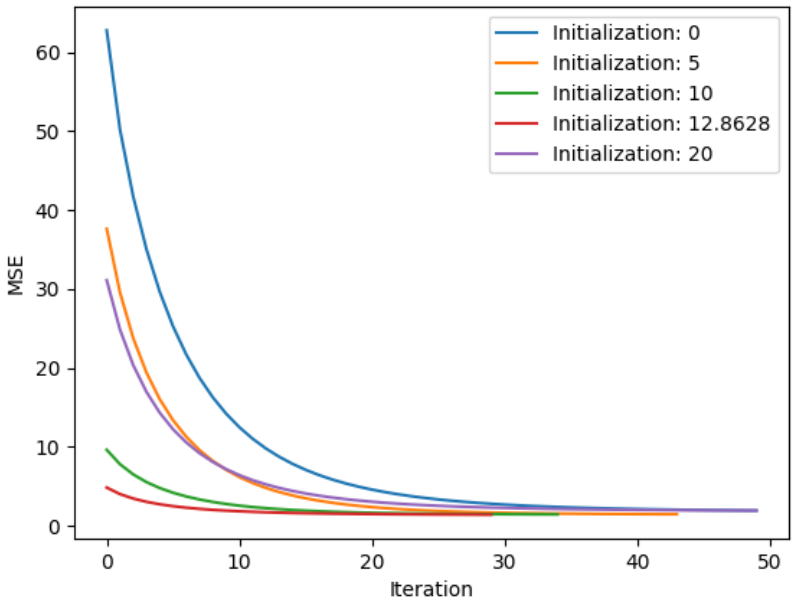


Figure 11: Sample of one of the graphs used in our dataset. Specifically for the pm2.5 pollutant in the year 2018.

4.7 Graph Neural Network training

In the present section we will be discussing the training process of the GNNs. First of all, we will talk about the data used for this goal. Our dataset is composed of two different graphs, one with data from the year 2018 and the other from the year

	GCN-S	GCN-M	GCN-L	SAGE_3-S	SAGE_3-M	SAGE_3-L	SAGE_5-S	SAGE_5-M	SAGE_5-L
80% MSE	5.3493	5.3468	5.4108	1.628	1.5338	**1.2535**	1.9824	1.38	1.3733
99% MSE	8.823	8.6541	8.3669	**4.4284**	4.636	4.8877	7.6484	5.1323	5.4785
80% MAPE	13.0467	12.9961	12.9421	6.252	5.8608	5.2645	4.9679	5.8403	**4.6188**
99% MAPE	18.9384	18.8288	18.2218	**14.2787**	14.9617	15.1193	15.2428	15.5539	14.44
Training time (s)	4.736	9.691	26.74	**1.675**	5.392	5.893	2.456	5.304712	12.24661
Parameters	1101	25501	101001	**261**	5301	20601	681	15401	60801

Table 2: Comparison of different GNN architectures. The performance achieved for 80 and 99 percent of missing nodes is shown, as well as the average training time and the number of trainable parameters.

2019, both of them with a size of 9466 nodes and 14684 edges. For training, we split the 2018 graph in two diferent sets, 80% to train and 20% to evaluate. We consider 2019 graph as the test set by itself, and the final scores once we have chosen the best model will be obtained using this test set.

We define the loss function as the Mean Squared Error, to match the metric we where using previously. MSE is a widely used loss function in regression problems like this one. This will be the objective function to minimize by the GNN. For this case, the concrete form of the loss is given in Eq. 4.3, where y_n corresponds to the actual node value and $\hat{y_n}$ the predicted value.

$$loss = \frac{1}{n}\sum_{i=1}^{n}(y_n - \hat{y_n})^2 \tag{4.3}$$

The optimizer used is ADAM [47], with a learning rate parameter, lr, set to 0.001. In Figure 12 we show the results of the training process for three different percentages of missing nodes for a given GNN architecture. The figure shows the validation loss value in logarithmic scale versus the training epoch. As we can see, the initial behaviour of the loss in every case is very noisy. However, the performance of the GNN converges to a given value, which is higher when the number of missing nodes is also higher. This result is in line with the intuition of the problem.

In Table 2 we show the results obtained for different GNN architectures. Due to the infinite variety of architectures that can be build, we have decided to systematize the experiments as much as possible. For this purpose, we have selected two well-known architectures, Graph Convolutional Networks (GCN) and GraphSAGE (or

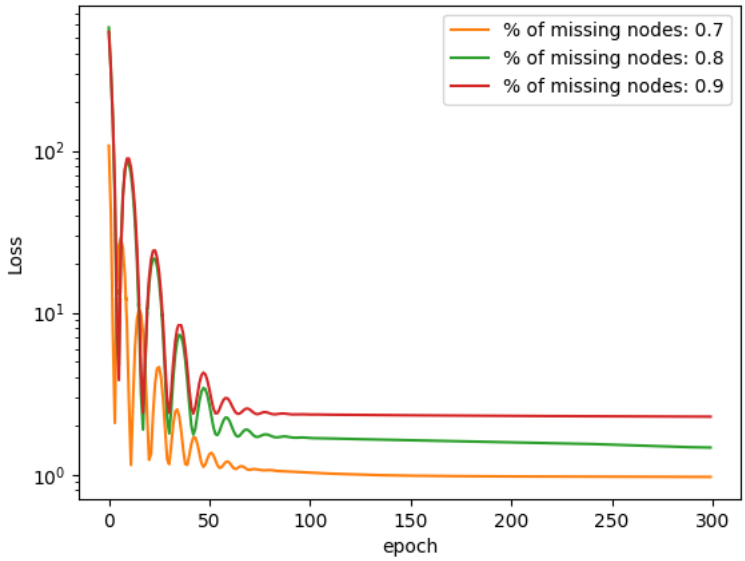


Figure 12: Training results of a GNN model for three different missing node portcentages. The trained model is in this case a SAGE architecture, with 3 layers and 20601 trainable parameters.

SAGE). Under the SAGE architecture we have built two different models, SAGE3 and SAGE5, with 3 and 5 layers respectively. On the other hand, GCN has 7 layers. The letters S, M and L (Small, Medium and Large) correspond to the number of parameters in the hidden layers of the models. This number corresponds to 10, 50 and 100 respectively. As can be seen, the SAGE architectures achieve better performances for all the cases studied with ease. The lowest MSE obtained are found in the SAGE3 architecture, being SAGE3-S the best for 99 percent of missing nodes and SAGE3-L when the percentage is 80.

Chapter 5

Discussion

In this chapter we will discuss the results obtained from the experiments, presented in the previous chapter. From this discussion we will try to draw relevant conclusions on the main aspects of the proposed method, as well as its strengths and weaknesses with respect to existing methods. Finally, the limitations of the method and of the conducted experiments will be discussed, and a possible future line of research to alleviate these limitations will be proposed.

5.1 Discussion

In the present work, we have performed a set of experiments to compare the accuracy achieved by the proposed method in comparison to other existing methods. The combination of MAMP and GNNs produces the best results for any percentage of unknown nodes. Moreover, it does not suffer from intrinsic limitations imposed by the large size of the network. Finally, contrary to the initial hypothesis, weighting by the distances between nodes does not improve the result obtained with the simple MAMP method.

With respect to its convergence properties, it can be observed experimentally, that they follow the monotonic improvement condition, generating smooth curves that go down to a given convergence point. Knowing that this convergence point exists

in the method means that it can be executed an arbitrary number of times without any worsening of the result. However, we have not yet formalized this experimental result theoretically, so we do not know its conditions or generalities. Our starting hypothesis is, however, that one could generalize the monotonic energy minimization result to any arbitrary graph. This would fit with the goal of the algorithm, which is ultimately to reduce the difference between connected nodes. However, we believe that this would not necessarily be the case for the MSE error in an arbitrary graph. If the Graph Signal in that graph is not smooth, it could be that a minimization of the energy does not imply a monotonic reduction in the error. Further studies would be necessary to draw clearer conclusions in this regard.

Experiments have also shown that a lower number of known nodes leads to a higher number of iterations required until convergence. This could be a serious limitation of our method when we have to deal with highly unknown graphs due to the computational time required.

The last feature of the MAMP algorithm we analyzed is node initialization, in Section 4.5. Before running this experiment, we had the hypothesis given by intuition that the best node initialization could be close to the mean value of the signal across the graph. This intuition is given by the fact that we are calculating the average between all neighboring nodes, including the unknown ones. In this sense, if the distance between the initial value and the actual value is smaller, so will be the effort to reach that value. As we can observe from the experiment performed, our hypothesis seems to be experimentally supported.

Finally, in Section 4.6 we took a closer look at the influence that the architecture of the GNN model used can have on the performance of our method. It has been observed that the results of the GNNs are not always better than those obtained by the MAMP algorithm alone. However, with the right GNN architecture, significantly better errors can be achieved. In view of the experiments, the SAGE architecture showed to be promising. Interestingly, deeper models or those with a greater number of parameters are not necessarily the best ones. After these first experiments, we believe that it is necessary to continue in this line with new models. It is possible

that by carrying out experiments with a greater variety of architectures, we may be able to find others that reduce the error even more.

5.2 Conclusions

In the present work we have attacked the problem of spatial prediction of air quality. We have seen that, in spite of being a well-studied problem, certain particularities of our case study have forced us to carry out novel research in this aspect. First of all, a formalization of the problem based on Markov Random Fields has been carried out, following the recommendations of recent works in this sense, in order to favor the unification of the theoretical frameworks of the Graph Signal Processing field. After this, a novel method for solving the problem has been developed. This method is based on a message passing scheme, where the values of the nodes are updated by performing the arithmetic averages of the neighboring nodes. This message passing system performs a homogenization of the graph signal, which produces a good approximation to the real value under a homophily graph assumption. It has also been proved that, after training a Graph Neural Network on an external training set, the performance of this method can be significantly improved. Once the method has been built, it has been compared with other existing methods in the literature. It has been observed that the proposed method combining message passing and GNNs produces the best result among the compared methods. In addition to its efficiency, the proposed method is conceptually and mathematically simple, which would be an advantage over other methods that attack similar problems.

Despite these advantages, it is important to note that many lines of work remain open with respect to the proposed method. In the first place, the data with which we have worked are, for reasons beyond our control, very limited. This causes the results obtained in this work to be taken carefully. Therefore, more research is needed to test this method in new cases. In addition, in this work, experiments have been carried out to measure its efficacy experimentally, but the theoretical foundations are still unknown. We believe it would be interesting to work in this line in the future to construct conditions necessary for the correct functioning of the

method and to develop theorems that expose its generality or properties.

List of Figures

List of Tables

Bibliography

[1] World health organization. (n.d.). air pollution. world health organization. retrieved june 5, 2022, from. URL `https://www.who.int/health-topics/air-pollution`.

[2] Khomenko, S. *et al.* Premature mortality due to air pollution in european cities: a health impact assessment. *The Lancet. Planetary health* (2021).

[3] Organization, W. H. *WHO global air quality guidelines: particulate matter (PM2.5 and PM10), ozone, nitrogen dioxide, sulfur dioxide and carbon monoxide* (World Health Organization, 2021).

[4] for Europe, W. H. O. R. O. *Air quality guidelines: global update 2005: particulate matter, ozone, nitrogen dioxide and sulfur dioxide* (World Health Organization. Regional Office for Europe, 2006).

[5] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M. & Monfardini, G. The graph neural network model. *IEEE transactions on neural networks* **20**, 61–80 (2008).

[6] Zhou, J. *et al.* Graph neural networks: A review of methods and applications. *AI Open* **1**, 57–81 (2020).

[7] Sanchez-Gonzalez, A. *et al.* Graph networks as learnable physics engines for inference and control. In *International Conference on Machine Learning*, 4470–4479 (PMLR, 2018).

[8] Khalil, E., Dai, H., Zhang, Y., Dilkina, B. & Song, L. Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems* **30** (2017).

[9] Garcia, V. & Bruna, J. Few-shot learning with graph neural networks. *arXiv preprint arXiv:1711.04043* (2017).

[10] Rico, J., Barateiro, J. & Oliveira, A. L. Graph neural networks for traffic forecasting. *ArXiv* **abs/2104.13096** (2021).

[11] Reddy, V. N. & Mohanty, S. Deep air : Forecasting air pollution in beijing , china (2017).

[12] Zeinalnezhad, M., Chofreh, A. G., Goni, F. A. & Klemeš, J. J. Air pollution prediction using semi-experimental regression model and adaptive neuro-fuzzy inference system. *Journal of Cleaner Production* **261**, 121218 (2020).

[13] Xayasouk, T., Lee, H. & Lee, G. Air pollution prediction using long short-term memory (lstm) and deep autoencoder (dae) models. *Sustainability* **12**, 2570 (2020).

[14] Teng, Y., Huang, X., Ye, S. & Li, Y. Prediction of particulate matter concentration in chengdu based on improved differential evolution algorithm and bp neural network model. *2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)* 100–106 (2018).

[15] Bekkar, A., Hssina, B., Douzi, S. & Douzi, K. Air-pollution prediction in smart city, deep learning approach. *Journal of big Data* **8**, 1–21 (2021).

[16] Mao, W. *et al.* A hybrid integrated deep learning model for predicting various air pollutants. *GIScience & Remote Sensing* **58**, 1395–1412 (2021).

[17] Li, X., Peng, L., Hu, Y., Shao, J. & Chi, T. Deep learning architecture for air quality predictions. *Environmental Science and Pollution Research* **23**, 22408–22417 (2016).

[18] Lu, Y.-J. & te Li, C. Agstn: Learning attention-adjusted graph spatio-temporal networks for short-term urban sensor value forecasting. *2020 IEEE International Conference on Data Mining (ICDM)* 1148–1153 (2020).

[19] Ouyang, X., Yang, Y., Zhang, Y. & Zhou, W. Spatial-temporal dynamic graph convolution neural network for air quality prediction. *2021 International Joint Conference on Neural Networks (IJCNN)* 1–8 (2021).

[20] Pierangeli, I., Nieuwenhuijsen, M. J., Cirach, M. & Rojas-Rueda, D. Health equity and burden of childhood asthma - related to air pollution in barcelona. *Environmental research* 109067 (2020).

[21] de Bont, J. *et al.* Ambient air pollution and overweight and obesity in school-aged children in barcelona, spain. *Environment International* **125**, 58 – 64 (2019).

[22] Rodríguez-Rey, D., Guevara, M. & Garcia, J. C. An integrated model system tool to evaluate the impact of urban mobility policies on air pollution: Barcelona case study (2020).

[23] Mota-Bertran, A., Saez, M. & Coenders, G. Compositional and bayesian inference analysis of the concentrations of air pollutants in catalonia, spain. *Environmental Research* **204**, 112388 (2022).

[24] Bruna, J., Zaremba, W., Szlam, A. D. & LeCun, Y. Spectral networks and locally connected networks on graphs. *CoRR* **abs/1312.6203** (2014).

[25] Kipf, T. & Welling, M. Semi-supervised classification with graph convolutional networks. *ArXiv* **abs/1609.02907** (2017).

[26] Henaff, M., Bruna, J. & LeCun, Y. Deep convolutional networks on graph-structured data. *ArXiv* **abs/1506.05163** (2015).

[27] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O. & Dahl, G. E. Neural message passing for quantum chemistry. In *International conference on machine learning*, 1263–1272 (PMLR, 2017).

[28] Battaglia, P. W. *et al.* Relational inductive biases, deep learning, and graph networks. *ArXiv* **abs/1806.01261** (2018).

[29] Tseng, C. C. & Lee, S.-L. Frequency selective filtering of graph signal in directed graph fourier transform domain. *2021 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)* 1–2 (2021).

[30] Srivastava, D., Bagler, G. & Kumar, V. Graph signal processing on protein residue networks helps in studying its biophysical properties. *bioRxiv* (2021).

[31] Zhang, C., Florêncio, D. A. F. & Chou, P. A. Graph signal processing – a probabilistic framework (2016).

[32] Gadde, A. & Ortega, A. A probabilistic interpretation of sampling theory of graph signals. In *2015 IEEE international conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3257–3261 (IEEE, 2015).

[33] Ji, M. & Han, J. A variance minimization criterion to active learning on graphs. In *AISTATS* (2012).

[34] Zhu, X., Ghahramani, Z. & Lafferty, J. D. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML* (2003).

[35] Koller, D. & Friedman, N. *Probabilistic graphical models: principles and techniques* (MIT press, 2009).

[36] Hernández-Lemus, E. Random fields in physics, biology and data science. *Frontiers in Physics* **9**, 641859 (2021).

[37] Kondor, R. & Lafferty, J. D. Diffusion kernels on graphs and other discrete input spaces. In *ICML* (2002).

[38] Belkin, M., Matveeva, I. & Niyogi, P. Regularization and semi-supervised learning on large graphs. In *COLT* (2004).

[39] Pearl, J. Reverend bayes on inference engines: A distributed hierarchical approach. *Probabilistic and Causal Inference* (1982).

[40] Davison, A. J. & Ortiz, J. Futuremapping 2: Gaussian belief propagation for spatial ai. *ArXiv* **abs/1910.14139** (2019).

[41] LeCun, Y., Chopra, S., Hadsell, R., Ranzato, A. & Huang, F. J. A tutorial on energy-based learning (2006).

[42] Open Data BCN | Servicio de datos abiertos del Ajuntament de Barcelona. URL `https://opendata-ajuntament.barcelona.cat/es`.

[43] Wang, M. *et al.* Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315* (2019).

[44] Hagberg, A., Swart, P. & S Chult, D. Exploring network structure, dynamics, and function using networkx. Tech. Rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States) (2008).

[45] Defferrard, M., Martin, L., Pena, R. & Perraudin, N. Pygsp: Graph signal processing in python. URL `https://github.com/epfl-lts2/pygsp/`.

[46] Ferrer-Cid, P., Barceló-Ordinas, J. M. & García-Vidal, J. Graph signal reconstruction techniques for iot air pollution monitoring platforms. *ArXiv* **abs/2201.00378** (2022).

[47] Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. *CoRR* **abs/1412.6980** (2015).