Master Project title

Final master project

Cristina Gava

July 16, 2018

Contents

1 The emerging field of graph signal processing **5**

1. Mathematical description of the graph structure 6
2. the inverse covariance matrix 7

2 The Graph learning issue **9**

1. Choosing the right dictionary 12
2. The correspondence between the dictionary and the graph signal 14

3 The smoothness assumption **17**

1. Smoothness and sparsity 18
2. Brief state of the art 19
3. The algorithms we started from: Dorina's and Hermina's 19

4 Problem Presentation **21**

1. State of the art 21
2. Problem description 22
    - 2.1 Assumptions 22
    - 2.2 Representation graph 23
    - 2.3 Dictionary learning section 23
3. Problem presentation 24
4. main improvements 24
    - 4.1 first part: the dictionary learning under smoothness constraints 25

4.2 second part: the graph learning part 25

4.3 Merging the two approaches 25

5. further improvements 25

5.1 graph signal clustering 25

5.2 varargin 25

6. conclusions 25

7. acknoledgements 25

Chapter 1

The emerging field of graph signal processing

Nowadays huge amounts of data are collected every day, from engineering sciences to our personal data regarding health monitoring, to financial data or political influences. The analysis and the process of these huge datasets has become a non indifferent challenge, since, in fact, data tends to reside on complex and irregular structures, which progressively required the introduction of innovative approaches to better handle and utilize them. [1] [2]

One of the central entities we will use in this work is the concept of *graph signal*. Graphs are generic data representation structures, which are useful in a great variety of fields and applications for everyday life and technology in general. These structures are composed by two main elements: the nodes and the edges; the former are a set of points, identifying an aspect of the graph structure itself, while the latter are connections between the nodes. Several structures we can encounter in natural entities and abstract constructions can be represented by a graph structure, and when we associate values to their set of nodes and edges we obtain a graph signal. To be specific, a graph signal is seen as a finite collection of samples located at each node in the structure and which are interconnected among themselves through the edges, to which we can associate numerical values representing the weights of the connections. The metric for these weights is not unique, as it depends on which relation between the nodes we are looking at: a typical metric for graph weights may be, for example, inversely proportional to the distance (physical or not) between two different nodes, but other options are possible.

As previously mentioned, graph structures appear to be significantly helpful when they are used to represent signals and their applications are the most varied: we could focus on transportation networks, where we could want to work with data describing human migration patterns or trade goods; we could also apply graph theory over social networks, in which users can be seen, for example, as our nodes while their connections to friends are our edges. [1] Brain imaging is another interesting application of graph signals: through it we could infer the inner structure of some regions of the brain, in a way that permits us to better understand physiological dynamics. [3] We could go on listing a considerable amount of valuable applications, from genetic and biochemical research to fundamental physical experiments; what these big amounts of data have in common is the fact that their complex inner structure makes it difficult to apply most of the well-known tools, like Principal Component Analysis (PCA), spectral analysis or Singular Value Decomposition (SVD), without redefining the signal structure. [2]

1. Mathematical description of the graph structure

In the field of Graph Signal Processing (GSP), spectral graph theory plays an important role, it focuses on analysing, constructing and manipulating graphs and makes uses of well know tools and concepts as frequency spectrum and the graph Fourier transform. In our work, the signals we are considering are defined on an undirected, connected, weighted graph $\mathcal{G} = \{\mathcal{V}, \varepsilon, W\}$ consisting on a finite set of vertices $\mathcal{V}$ with $|\mathcal{V}| = N$, a set of edges ε and a weighted adjacency matrix $\mathcal{W}$. The adjacency matrix contains the information of the connections between two nodes: if an edge e is present between two vertices i and j, then the entry $W_{i,j}$ represents the weight of that edge, while if there is no connection between two nodes, the value of the edge is null ($W_{i,j} = 0$). Moreover, if the graph was not connected and had K connected components, then we could decompose the graph signal over $\mathcal{G}$ into M sections which can be processed each one independently of the other.

Over this structure, we can then lay **(Cri says: cerca un verbo migliore)** the signal we need to analyse: to be precise, a signal or function $f : \mathcal{V} \to \mathbb{R}$ defined on the vertices of our graph, can be

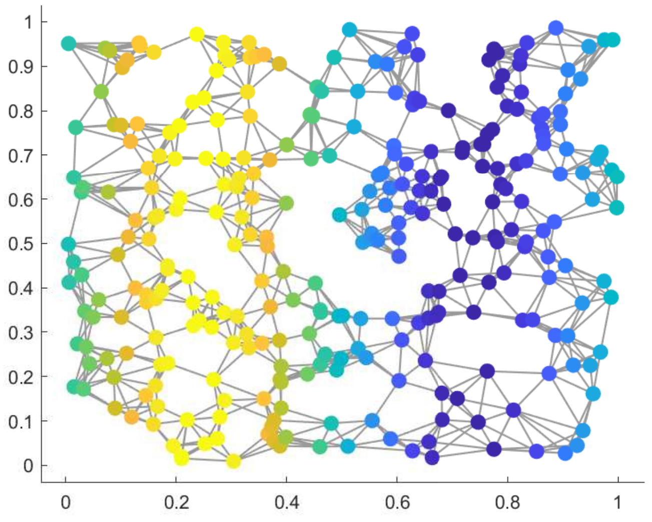

Figure 1.1: Random sensor graph

described as a vector $f \in \mathbb{R}^N$ that has the signal value at the i^{th} vertex in $\mathcal{V}$ as the i^{th} component of the vector f. Figure 1.1 shows an example of graph signal: the colours of the vertices represents the values the signal has on them.

2. the inverse covariance matrix

(Cri says: Vedere se aggiungere o meno questa sezione)

The graph Laplacian

To go from the graph vertex domain (which plays the same role of the time domain in the classical signal processing theory) to the spectral domain the *graph Laplacian operator*, or *Combinatorial graph Laplacian*, has been introduced, which is defined as $L := D - W$, where D represents a diagonal matrix whose i^{th} diagonal element is equal to the sum of weights of all the edges incident to the vertex i. This entity is a difference operator, since it satisfies the condition:

$$((Lf)(i) = \sum_{j \in \mathcal{N}_i} W_{i,j}[f(i) - f(j)] \tag{2.1}$$

The element $\mathcal{N}_i$ represents the set of vertices connected to vertex i by an edge.

From its definition, the graph Laplacian L is a real symmetric matrix, thus it has a complete set of orthonormal eigenvectors, here denoted by $\{u_l\}_{l=0,1,\ldots,N-1}$, to which real and non-negative eigenvalues are associated ($\{\lambda_l\}_{l=0,1,\ldots,N-1}$). Together with the eigenvectors, they satisfy the condition:

$$Lu_l = \lambda_l u_l, \text{ for } l = 0, 1, \ldots, N-1 \tag{2.2}$$

In the eigenvalues set, the one corresponding to 0 has a multiplicity equal to the number of connected components of the graph; from this, since we are considering only connected graphs, we can assume the Laplacian eigenvalues have the distribution: $0 = \lambda_0 < \lambda_1 \leq \lambda_2 \cdots \leq \lambda_{N-1} = \lambda_{max}$ and, of course, the set $\sigma(L) = \{\lambda_0, \lambda_1, \ldots, \lambda_{N-1}\}$ is the entire spectrum of our signal.

Eigenvalues and eigenvectors are then used to build up the graph version of a well known and useful transform, which is the *classical Fourier Transform* and that we recall here being defined as:

$$\hat{f}(\xi) = \langle f, e^{2\pi i \xi t}\rangle = \int_{\mathbb{R}} f(t) e^{-2\pi i \xi t} dt \tag{2.3}$$

and that we can see as the expansion of a function f in terms of the complex exponentials, elements which represent the eigenfunctions of the one-dimensional Laplace operator:

$$-\Delta(e^{2\pi i \xi t}) = -\frac{\partial^2}{\partial t^2} e^{2\pi i \xi t} = (2\pi\xi)^2 e^{2\pi i \xi t} \tag{2.4}$$

From the observation of the classical Fourier Transform, we can analogously define the *Graph Fourier Transform* $\hat{f}$ of any function $f \in \mathbb{R}^N$ on the vertices of $\mathcal{G}$ as the expansion of f in terms of the eigenvectors of the graph Laplacian [3]:

$$\hat{f}(\lambda_l) = \langle f, u_l\rangle = \sum_{i=1}^{N} f(i) u_l * (i) \tag{2.5}$$

while, at the same time, the *inverse Graph Fourier Transform* is given by:

$$f(i) = \sum_{l=0}^{N-1} \hat{f}(\lambda_l) u_l(i) \tag{2.6}$$

Chapter 2

The Graph learning issue

For the most part, the research regarding graph signal processing has been focusing on working with the signals spanned onto a graph manifold, assuming the graph structure to be already known. However, the choice of the underlying graph for the signal not necessarily represents faithfully the ground truth intrinsic manifold over which the signal is structured. In these cases, a more suitable approach to the problem could be trying to learn the graph structure underneath, such that the processing of the signal can rely on assumptions that better capture the intrinsic relationships between the entities or make them clear when otherwise it would not have be so. [4] Clearly the question is not trivial, since in general the action of learning a graph from sample data represents an ill-posed problem, with the consequence that there may be many solutions to the structure associated to a data. [5] To overcome this obstacle several approaches have been designed in the recent years, and to some of them we will give a more detailed description in the next chapters, when we will address the basis of our work.

Representing a graph: graph learning techniques and dictionary representation

In the previous section we presented the main reason why signal processing field expressed the necessity of having new different structures in order to better represent the amount of information we can collect from a phenomena: what we did not focused on yet, is the fact that these amount of data are usually largely redundant, since they are a densely sampled version of the signal and they may represent multiple correlated versions of the same physical event. So normally the significative information regarding the underlying processes is largely reducible in dimensionality with respect of the collected dataset. [6] Thus, we can obtain the data representations starting from the idea that our observations can be described by a sparse subset of elementary signals - so called *atoms* - taken from an *overcomplete dictionary*. When the dictionary forms a basis, then every signal can be univocally represented through the linear combination of the dictionary atoms, moreover the overcompleteness property **(Cri says: definisci i significati delle n. Sei sicura che stai avendo a che fare con dizionari overcomplete? Si si)** implies that atoms are linearly dependent. [6] [7]

With these premises, we consider a dictionary $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_L] \in \mathbb{R}^{N \times L}$, in which the columns represent the dictionary atoms and, for the overcompleteness, $L \geq N$. Through this entity, the signal representations can be done through two main paths, either the *synthesis* path, or the *analysis* path, and the two can significantly differ in the overcomplete case.

In the synthesis path, the signal $\mathbf{x} \in \mathbb{R}^N$ is represented as a linear combination of the dictionary atoms:

$$\mathbf{x} = \mathbf{D}^T \gamma_s \tag{0.1}$$

while in the analysis path it is represented through its inner product with the atoms:

$$\gamma_a = \mathbf{D}^T \mathbf{x} \tag{0.2}$$

The representation in 0.1 has the consequence that, when the dictionary is overcomplete, the set of representations $\gamma_s$satisfying the equation is *infinitely large*, allowing us to look for the most informative representation of the signal with respect of a certain cost function $\mathcal{C}(\gamma)$. In this way we arrive to a first general optimization problem in the form:

$$\gamma_s = \underset{\gamma}{\operatorname{argmin}} \ \mathcal{C}(\gamma) \quad \text{Subject To } \mathbf{x} = D\gamma \tag{0.3}$$

The way we choose the form of the cost function obviously influences the structure of our solution, to be specific, one of our goals is to achieve sparsity in the representation of the signal, such that the signal reconstruction is reduced in dimensionality as we are trying to achieve from the beginning. Problem in 0.3 becomes what is commonly referred as *sparse coding*, and there are different functions we can apply in order to obtain it: these functions have the characteristic of being tolerant to large coefficients and at the same time importantly penalize small non-zero ones. Among these functions, our choice fell on the l norm, which is one of the simplest and most effective function of this type.

1. Choosing the right dictionary

What we did not focused on yet is the choice of the proper Dictionary for out task. In the research there has been so far, different models of Dictionaries have been defined and used for the most different purposes, in the beginning the attention was mainly on traditional dictionaries, such as wavelet and Fourier dictionaries, which are simple to use and perform well for 1-dimensional signals. However, these structures were too simple to properly describe more complex and high-dimensional data, so the focus slowly moved to seeking solutions that better performed in this environment. Dictionaries emerged from this need were coming from two main sources:

- As an *analytical model*, which allows to extract the dictionary straight form the data for a fast implicit implementation that does not require multiplication by the dictionary matrix;
- As a *set of realizations* of the data, which offers an increased flexibility and adaptability to data structure;

The first model prefers speed over adaptability, since its success depends on the chosen underlying model, sometimes resulting in a over-simplistic solution for the proposed problem. From this, the necessity of also focusing on the second type of dictionary, also defined as *trained dictionary*.
Machine learning techniques of the period between 1980's and 1990's allowed to approach this problem under the new assumption that the structure of a natural phenomena can be accurately extracted *straight form the data* with better results than using a mathematical formulation. The most recent training methods focus on the l^0 and l^1 sparsity measures, which have a simpler formulation and at the same time can use the more recent sparsity coding techniques. [8] [9] Among these learning methods, particular relevance acquired the *parametric training methods*, such as *translation-invariant dictionaries*, *multiscale dictionaries* or *sparse dictionaries*. These implementations involve the reduction of parameters' number and the assumption of several desirable properties on the dictionary, in the end leading to an accelerate convergence, reduced density of the local minima and the convergence to a better solution. Moreover, the generalization of the learning process is improved thanks to the smaller number of parameters, as much as the reduction in number of examples needed. Finally, parametric dictionaries bring to a more efficient implementation, due to the fact that parametrization typically has a more compact representation and, not less important, a parametric dictionary may be designed to represent infinite or arbitrary-sized signals. [7] **(Cri says: forse aggiungi parte sugli sparse dictionaries sempre presa dallo stesso paper. Poi: vedi se aggiungere un excursus storico sui dizionari ad apprendimento)**.

The sparse approximation

The intention of sparse approximations to represent a certain signal y of dimension n as the linear combination of a small amount of signals selected from the source database which is the dictionary and in which the elements typically are unit norm functions called *atoms* and can be denoted through ϕ_k with $k = 1, \ldots, N$ and being N the size of the dictionary. When a dictionary is overcomplete, then every signal can be represented with the aforementioned linear combination:

$$\mathbf{y} = \phi \mathbf{a} = \sum_{k=1}^{N} a_k \phi_k \tag{1.4}$$

and the value that **a** can assume is not unique.

To achieve sparse and efficient representations, the requirement for finding the exact representation is in general relaxed: starting from the NP-hard problem which is the minimization of the l^0 norm of **a**, we can approach the issue through convex relaxation methods that solve a problem in the form:

$$\min_{\mathbf{a}} \ (||\mathbf{y} - \phi \mathbf{a}||_2^2 + \lambda ||\mathbf{a}||_1) \tag{1.5}$$

in which the relaxation allowed to replace the nonconvex l^0 norm in the original problem with a more meaningful l^1 norm. [6]

2. The correspondence between the dictionary and the graph signal

In 2. we introduced the Laplacian operator as $L = D - W$, here we focus on a modified version of this operator, that is the *normalized Laplacian* $\mathcal{L} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$, also a real symmetric and positive semidefinite matrix, with a complete set of orthonormal eigenvectors $\chi = [\chi_0, \chi_1, \ldots, \chi_{N-1}]$ and associated non negative eigenvalues $\sigma(\mathcal{L}) := \{0 = \lambda_0, \lambda_1, \ldots, \lambda N - 1 \leq 2\}$ (where the upper bound 2 comes from the normalization). The normalized Laplacian eigenvectors are Fourier basis, this bringing any function y defined on the vertices of the graph to be represented in his Fourier transform version $\hat{y}$ at frequency λ_l as:

$$\hat{y}(\lambda_l) = \langle y, \chi_l \rangle = \sum_{n=1}^{N} y(n) \chi_l^*(n) \tag{2.6}$$

and its relative inverse transform to be:

$$y(n) = \sum_{l=0}^{N-1} \hat{y}(\lambda_l) \chi_l(n), \quad \forall n \in \mathcal{V} \tag{2.7}$$

This transform plays a major role not only in the harmonic analysis, but also in defining the signal translation onto a graph. It is worth to notice here that in the classical signal processing field the translation operator is defined through the change of variable $(T_n f)(t) := f(t - n)$, but when it comes to graph signal processing, this concept looses meaning since there is no significate to $f(\circ - n)$ in the graph environment. However, from transform theory we can recall that the classical translation operator T_n can also be seen as a convolution with a Kronecker delta δ centered in n[3]

[10]:

$$T_n g = \sqrt{N}(g * \delta_n) = \sqrt{N} \sum_{l=0}^{N-1} \hat{g}(\lambda_l) \chi_l^*(n) \chi_l \tag{2.8}$$

this leading to think about 2.8 as an operator acting on the kernel $g(\cdot)$ directly defined in the spectral domain and from which we can obtain the effective translation to kernel n through inverse graph Fourier transform. Moreover the term $\sqrt{N}$ in the Equation 2.8 is a guarantee for the translation operator to preserve the mean of the signal, while the $g(\cdot)$ kernel smoothness is a measure to control the localization of $T_n g$ centered at vertex n (meaning that the magnitude ($T_n g$)(i) of the translated kernel at vertex i decays with the increasing distance from the center of the kernel). We can thus compose our atoms $T_n g$ as smooth entities, coming from the assumption that the kernel function in 2.8 is a smooth polynomial function of degree K:

$$\hat{g}(\lambda_l) = \sum_{k=0}^{K} \alpha_k \lambda_l^k, \quad l = 0, \dots, N-1 \tag{2.9}$$

From this fact, we can so obtain a global expression for a translation operator as:

$$T_n g = \sqrt{N}(g * \delta_n) = \sqrt{N} \sum_{l=0}^{N-1} \sum_{k=0}^{K} \alpha_k \lambda_l^k \chi_l^*(n) \chi_l \tag{2.10}$$

$$= \sqrt{N} \sum_{k=0}^{K} \alpha_k \sum_{l=0}^{N-1} \lambda_l^k \chi_l^*(n) \chi_l = \sqrt{N} \sum_{k=0}^{K} \alpha_k (\mathcal{L}^k)_n \tag{2.11}$$

where $(\mathcal{L})_n$ represents the n^{th} column of the k^{th} power of the Laplacian matrix $\mathcal{L}^k$. The final concatenation of N such columns brings us to generate a set of N atoms, corresponding to the columns of

$$Tg = \sqrt{N}\hat{g}(\mathcal{L}) = \sqrt{N}\chi\hat{g}(\Lambda)\chi^T = \sqrt{N} \sum_{k=0}^{K} \alpha_k \mathcal{L}^k \tag{2.12}$$

Where the quantity Λ corresponds to the diagonal matrix of the eigenvalues in such a way that the following relation holds: $\mathcal{L} = \chi\Lambda\chi^T$ [5]. This brief section has the intrinsic meaning that if the kernel $g(\cdot)$ is a polynomial of degree K, then the translation operator is 0 in all the vertices i that are more than K hops far away from the center vertex n, which means that the vertex domain support of the translated vertex is contained in a sphere of radius K and center in n and its magnitude decades with the increasing distance from n to i. In Figure 2.1 there is an example of an atom translation in three different ways.

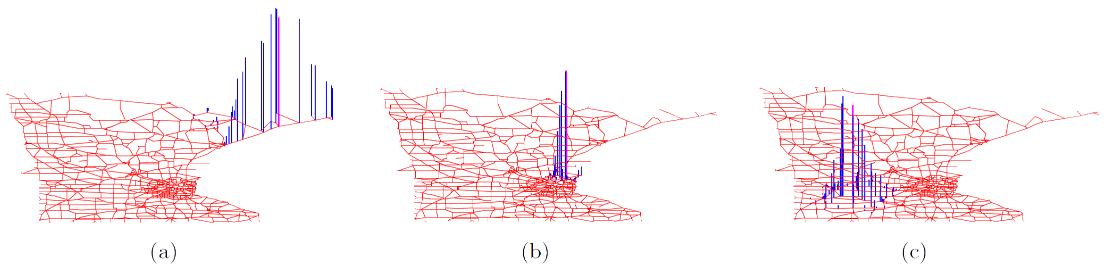

Figure 2.1: Example of graph signal translation: translation for $T_{200}g$ (a), $T_{1000}g$ (b) and $T_{2000}g$ (c)

Chapter 3

The smoothness assumption

Going further with the spectral analysis, some other intuitions about the graph spectrum can be conveyed from the from the classical setting to the graph setting.

One of them is that in the classical setting the eigenvalues carry a specific notion of frequency: complex exponentials related to lower frequencies are relatively smooth, slowly oscillating functions whereas for high frequencies the associated complex exponentials oscillate more rapidly. Reporting this onto the graph spectral settings, we see that the Laplacian eigenfunctions associated with lower eigenvalues can be seen as smooth, while the ones related to higher eigenvalues tend to oscillate more intensively. [11] In the specific case of the null eigenvalue the related eigenvector (χ_0) is constant and equal to $\frac{1}{\sqrt{N}}$ [3].

Taking a step back, one of the common simple assumptions about data residing on graphs, is that the signal changes smoothly between connected nodes; in this, a simple way to quantify how smooth is a set of vectors residing on a weighted undirected graph, is through the function:

$$\frac{1}{2}\sum_{i,j} W_{i,j}||\chi_i - \chi_j||^2 \tag{0.1}$$

with W_ij representing the weight between node i and node j, as usual, and χ_i and χ_j representing two vectors in $\chi_1, \chi_2, \ldots, \chi_N \in \mathbb{R}$. This means that if two vectors χ_i and χ_j reside on two nodes connected by a high W_{ij}, then they are expected to have a small distance. Equation 0.1 can be thus rewritten in matrix form as [12] :

$$\text{tr}(\chi^T L \chi) \tag{0.2}$$

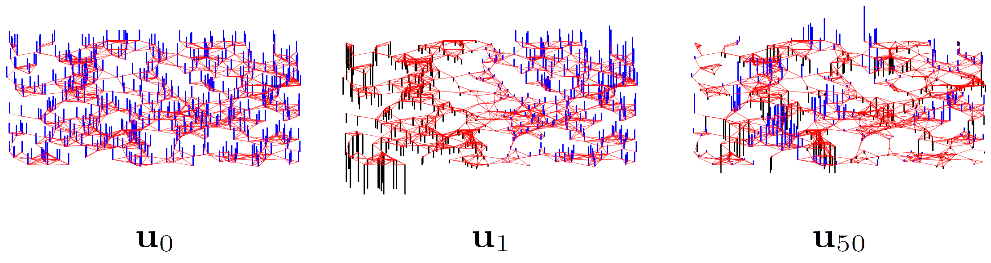

Figure 3.1: three graph Laplacian eigenvectors for a random sensor network graph. It can be seen how χ_{50} contains definitely more zero crossing than the other two vectors χ_0 and χ_1

and, recalling that the Laplacian matrix can be expressed as:

$$L = \chi \Lambda \chi \tag{0.3}$$

with Λ being the diagonal matrix containing the associated eigenvalues of the Laplacian, then we can insert 0.3 in 0.2, obtaining:

$$\mathrm{tr}(\chi^T L \chi) = \mathrm{tr}(\chi^T \chi \Lambda \chi^T \chi) = \mathrm{tr}(\Lambda) \tag{0.4}$$

This result clearly explains the relation between the signal smoothness and the eigenvalues of the Laplacian that we presented at the beginning of the paragraph. Figure 3.1 shows different graph Laplacian eigenvectors for a random graph. The eigenvectors associated to larger eigenvalues tend to oscillate more rapidly and to have dissimilar values on vertices connected by a high weight edge, this moreover bringing to higher occurrences of zero crossing associated to the graph Laplacian. [3].

1. Smoothness and sparsity

In [12] the previous notion of smoothness is directly connected with the one of graph sparsity: in fact they show that the smoothness term can be seen equivalently as a weighted l^1 norm of the adjacency matrix, which being minimized can lead to a graph with a sparse set of edges, that prefers only the ones associated to small distances in the *pairwise distances matrix* $\mathbb{Z} \in \mathbb{R}_+^{N \times N}$ defined as $\mathbb{Z}_{i,j} = ||\chi_i - \chi_j||^2$.

This concept is strictly connected with the compressibility property of a signal, since previous studies validate the hypothesis that smooth signals are likely to be compressible in the frequency domain. In [13] it has been shown that the liner approximation error is upper bounded by the overall variation of the signal, this bringing to the conclusion that if the total variation of a signal is small, then the approximation error is small. This resulting in the fact that the signal can be well approximated by a small portion of the Fourier coefficients.

2. Brief state of the art

(Cri says: vedi dopo se scrivere qualcosa)

3. The algorithms we started from: Dorina's and Hermina's

Chapter 4

Problem Presentation

Given the premises presented in the previous chapters, our work mainly focused on two open issues regarding both the dictionary learning and the graph learning problem. The starting point were the works of Dorina Thanou- regarding the dictionary learning approach for graph signals - ant the work of Hermina Petric Maretic - regarding the graph learning approach.
In the related work, Thanou et al. proposed a learning algorithm which was able to retrieve the kernel functions for a certain dictionary, alternately optimizing over the kernel coefficients α and the sparsity matrix X, while Maretic et al. started from the complementary assumption that the entities know in their case where exactly the kernel coefficients, while they alternatively optimized over the sparsity matrix and the graph weight matrix.

1. State of the art

There has already been a discrete amount of work around graph inference for signals that are expected to be smooth on the graph. In [5] Dong et al. addressed the problem adopting a factor analysis model for graph signals and imposing a Gaussian probabilistic prior on the independent variables which model the signal. Their results showed that smoothness property of a graph signal is favoured by the efficient representation obtained through these priors. In particular, in the algorithm they proposed they deployed the use of l^1 and Frobenius norm, the former is expressed by a constraint on the Trace of the Laplacian, and eventually accounts for a sparsity term, while the latter is added as a penalty term in the objective function in order to control the distribution of the off-diagonal entries in L.

At the same time, in [12] Kalofolias proposes another framework to learn a graph structure under the smoothness assumption. They used as well the minimization of the trace term for the Laplacian matrix, clinching that the minimizations of it brings to naturally sparse solutions.

However, this assumption is not always exhaustively descriptive of the real problem we are considering, so we might want to add other reinforcements to the priors in order to better describe the situation. For example, we could imagine a real world network to show a localized behavior or a more organized one having a certain pattern repeating over the graph. In [4], this was an assumption for the signal, bringing it to be a sparse combination of a small number of atoms in a polynomial graph dictionary. This in the end was creating atoms localized in every node on the graph, repeating the same pattern over it with respect to every node as a source, in the same way it is described in [10]. The limitation Thanou's work is mainly the fact that, as we already mentioned, it assumes the kernels to be know function while it concentrates only on the graph dictionary and thus fixing the spectral behaviour of the atoms.

2. Problem description

As previously mentioned, this work deals with the general class of signals that can be represented in a sparse way through atoms of a polynomial dictionary [10], such that it can give a natural segregation of each signal into localised components, as shown in Figure 4.1. The figure shows how each of these components derive from one atom directly dependent of the graph dictionary and this is well explained if, for polynomial kernels of degree k, we see our atoms as an entities spanning only the k-hop distance from the node representing the center of the atom (source).

2.1 Assumptions

(Cri says: Verifica la faccenda della community structure che stai per scrivere) In opposition to the prior work, here we assume the sources of our atoms to be known and the same for all signals, meaning that the network has a certain amount of nodes acting like sources an thus controlling the behavior of the network. Another strictly related assumption, is that the graph has a strong

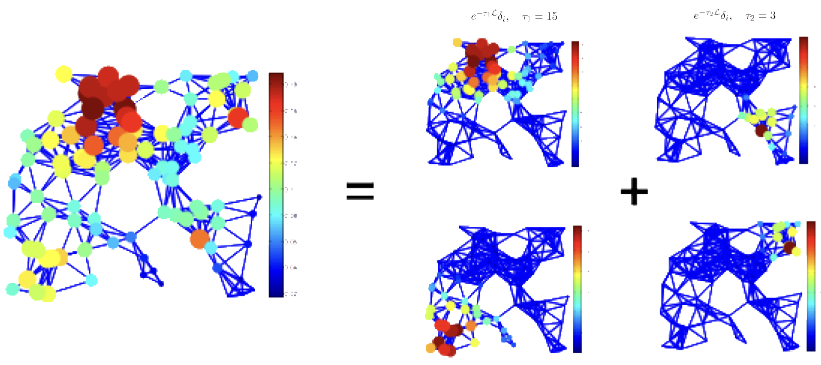
$e^{-\tau_1 \mathcal{L}}\delta_i, \quad \tau_1 = 15$
$e^{-\tau_2 \mathcal{L}}\delta_i, \quad \tau_2 = 3$
=
+

Figure 4.1: Example of signal decomposition using the polynomial dictionary

community structure, meaning that the graph can be separated into communities which are poorly related among themselves, but hold strong connections inside them. These two assumptions are somewhat exchangeable and the reduction of one of them could be a future interesting aspect to better develop.

2.2 Representation graph

2.3 Dictionary learning section

If we considered the former formulation for our problem, we would ignore the fact that atoms in D_small actually are coming from a graph dictionary, while this is on the contrary something we would like to include in our problem. In fact, we did assume these atoms to be vectors describing the signal, but we kept this part separated from the graph notion. We start from stating that we want that our atoms are representative of the portion of the graph they are covering, and so they should spread with non trivial values over all nodes of the subgraph. This reason mad us constraint our dictionary kernels in a way that they have mostly smooth support in the surroundings of the source node. This information is something different form the smoothness assumption we widely examined in the previous sections, since it adds some information strictly related to the behavior of the kernel itself. In fact, if we learned a smaller order polynomial with constraints in high frequencies, we would not have a large spread of our atoms, things we would like to have since in this way our atoms can represent a large portion of the graph. **(Cri says: rivedila ti prego)**

To have this new smoothness constraint, we look at the kernel polynomial:

$$g(\lambda) = \sum_{k=0}^{K} \alpha_k \lambda_k \tag{2.1}$$

and we try to constraint the roots to correspond to the highest eigenvalues. This would change the expression of the kernel and turn our learning problem into a simpler one, since now we are trying to learn a smaller number of coefficients:

$$g(\lambda) = h(\lambda)(\lambda - \lambda_n)(\lambda - \lambda_{n-1}) \cdot \cdots \cdot (\lambda - \lambda n - l + 1) \tag{2.2}$$

$$where \qquad h(\lambda) = \sum_{k=0} K - l\gamma_k \lambda^k \tag{2.3}$$

This formula them can bring to the following dictionary problem:

$$\arg \min_{\gamma_0,\ldots,\gamma_{K-l} X_{small}} ||Y - D_{small} X_{small}||_F^2 \tag{2.4}$$

$$with \qquad g(\lambda) = h(\lambda)(\lambda - \lambda_n)(\lambda - \lambda_{n-1}) \cdot \cdots \cdot (\lambda - \lambda n - l + 1) \tag{2.5}$$

$$h(\lambda) = \sum_{k=0} K - l\gamma_k \lambda^k \tag{2.6}$$

$$D_i = [g(L_i)]_{\text{source}_i} \tag{2.7}$$

$$0 \leq g(\lambda) \leq c \tag{2.8}$$

$$(c - \epsilon)I \prec \sum_{s=1}^{S} D_s \prec (c + \epsilon)I \tag{2.9}$$

(Cri says: Cambia gli operatori per le matrix ineq. con qualcosa che inclusa l'uguaglianza) Where the positions of the non-trivial values of X_{small} are known, but the values are to be learned. Moreover, the last two constraints in **??** are taken from [10] and ensure the kernels are bounded and span the entire spectrum.

3. Problem presentation

4. main improvements

4.1 first part: the dictionary learning under smoothness constraints

4.2 second part: the graph learning part

4.3 Merging the two approaches

5. further improvements

5.1 graph signal clustering

5.2 varargin

6. conclusions

7. acknoledgements

Bibliography

[1] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, “Graph Signal Processing: Overview, Challenges and Applications,” pp. 1–18, 2017.

[2] A. Sandryhaila, “Signal Processing,” no. September, pp. 80–90, 2014.

[3] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains,” *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[4] S. Processing, “Graph learning under sparsity priors Hermina Petric Maretic , Dorina Thanou and Pascal Frossard,” pp. 6523–6527, 2017.

[5] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, “Learning Laplacian Matrix in Smooth Graph Signal Representations,” *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6160–6173, 2016.

[6] I. Tosic and P. Frossard, “Dictionary Learning, What is the right representation for my signal?,” *Signal Processing Magazine, IEEE*, vol. 28, no. 2, pp. 27–38, 2011.

[7] B. R. Rubinstein, A. M. Bruckstein, and M. Elad, “Dictionaries for Sparse Representation Modeling.pdf,” vol. 98, no. 6, pp. 1045–1057, 2010.

[8] I. F. Gorodnitsky and B. D. Rao, “Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm,” *IEEE Transactions on Signal Processing*, vol. 45, no. 3, pp. 600–616, 1997.

[9] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, “Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition,” *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, pp. 40–44, 1993.

[10] D. Thanou, D. I. Shuman, and P. Frossard, “Learning parametric dictionaries for signals on graphs,” *IEEE Transactions on Signal Processing*, vol. 62, no. 15, pp. 3849–3862, 2014.

[11] D. I. Shuman, B. Ricaud, and P. Vandergheynst, “Vertex-frequency analysis on graphs,” *Applied and Computational Harmonic Analysis*, vol. 40, no. 2, pp. 260–291, 2016.

[12] V. Kalofolias, “How to learn a graph from smooth signals,” vol. 2, 2016.

[13] “"Approximating signals supported on graphs",” *Signals, Department of Electrical and Computer Engineering , McGill University 3480 University St , Montreal QC , Canada H3A 2A7*, vol. 3, pp. 3921–3924, 2012.