Signal processing using graphs

Andres Eduardo Rubiano Martinez
Manuela Alejandra Leal Pelaez

Codification theory

Humberto Sarria, Ph.D
Work line:
Applied Mathematics

Universidad Nacional de Colombia
Facultad de Ciencias
Departamento de Matemáticas
Bogotá, 2022

1 Abstract

In this work we present the basic notions necessary to introduce the signal processing using graphs, also we introduce the extension of the Laplacian operator and Fourier transform to graphs, these two will be useful to present a regularization method known as Tikhonov regularization which allow us to solve the deblurring an image problem, showed as a practical example of what signal processing using graphs can do.

2 Introduction

Graphs as a data structure allow us to represent data in a useful form that describes the geometric structure of data domains in different knowledge areas such as: social, energy, transportation, sensor and neural networks. To use graphs in signal processing we have to consider some attributes for them in addition to the usually components that they have (nodes and edges):

- **Weight:** Associated with each edge, it often represents the similarity between the two vertices that it connects.

- **Signal function:** Associated with the vertices, this function allows to represent the signal in a data set as a finite collection of samples with one sample at each vertex in the graph.

We can find examples of graph signals in many different engineering and science fields. In transportation networks, we may be interested in analyzing epidemiological data describing the spread of disease, census data describing human migration patterns, we can try to describe the behavior of the population during an electoral process, or logistics data describing inventories of trade goods (e.g., gasoline or raw materials). In brain imaging, it is now possible to non-invasive infer the anatomical connectivity of distinct functional regions of the cerebral cortex, and this connectivity can be represented by a weighted graph with the vertices corresponding to the functional regions of interest[3]. Common data processing tasks in these applications include filtering, denoising, inpainting, and compressing graph signals. Suppose you have an image file. But the only problem is it has lots of *noise*. You can't distinguish between that noise and the original image. So how are you supposed to distinguish between the *noise* and the actual signal? In this work we are goinig to study an algorithm know as Tikhonov regularization that allow us to recover information from a noisy image.

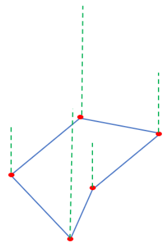

Figure 1: Random positive graph signal represented as vertical lines in each vertex, the edges can also have weights associated

3 The Fourier Transform and its Extension to Graphs

Noise is a random signal which consists of equal intensities or powers at every frequency. In computing, it is statistically defined as a sequence of random variables. So basically, in very simple terms, it is a random thing that may be a part of your signal.

Figure 2: Example of a image with *noise* (left) and the original image (right)

The Fourier Transform is one of deepest insights ever made, and it is a very useful tool for analyzing signals, especially noisy ones. It transforms or converts complex mathematical equations into simpler trigonometric functions in terms of sin or cos. These are used because the signal is easier to analyze in their format. In other terms, Fourier transformation is used to convert time signals into frequency signals and power signals[1].

The main goal of this work is tho show how can we extend the utility of the classical Fourier transform, given a integrable continuous function f

$$\hat{f}(\xi) := \langle f, e^{2\pi i \xi t} \rangle = \int_{\mathbb{R}} f(t) e^{-2\pi i \xi t} dt, \tag{1}$$

to the signal functions that we are going to study in the graph signal processing.

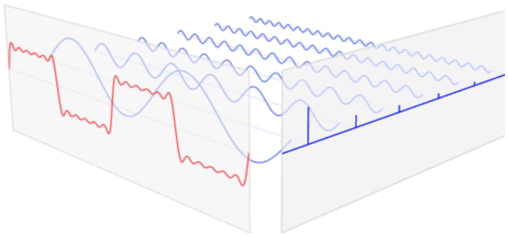

Figure 3: Example of a step function described by a multitude of sin waves

3.1 Weighted graphs

In order to define a correspondent Graph Fourier Transform, considering the mentioned weight and signal function attributes will be needed. Graphs defining signals not only consist of a finite set $\mathcal{V}$ of N vertices and a set of edges $\mathcal{E}$ that connects them, but also of a numerical value associated with each edge. This value corresponds to the *weight* of the edge, which is often positive as it represents cost, distance, capacity, or any other characteristic that can be deduced by the nature of the data or the physics of the problem. Therefore we will be working with labeled graphs where the labels are numerical values.

Graphs can be represented by an adjacency matrix $\mathbf{A}$ where the i, j^{th} entry indicates whether the vertices i and j are connected or not using zeros and ones for example. In other words, any graph can be seen as a weighted graph which weight application indicates if there exists an edge connecting vertices or not. When the edges set is a multi-set this will indicate the number of edges present between each pair of vertices. Accordingly, weighted graphs will be represented by a square adjacency matrix $\mathbf{W}$ where the element $W_{i,j}$ corresponds to the weight associated with the edge that connects vertices i and j, and is therefore null when this pair of vertices are not connected.

The properties of this matrix give indications about the distribution of information represented in the graph, for instance for a non-directed graphs this will be symmetric but not necessarily for a directed one, and the values on the diagonal specify the presence of loops. Also, when considering a graph with different components, it can be partitioned by the adjacency matrices of each. Consequently, we will work with (undirected) weighted graphs represented as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{W}\}$.

3.2 Graph Laplacian Operator

Furthermore, a difference operator can be defined using $\mathbf{W}$. Adding up the weights of those edges incident to a vertex i determines i^{th} element of a diagonal matrix $\mathbf{D}$ such that the graph Laplacian will be defined as $\mathbf{L} = \mathbf{D}$ **-** $\mathbf{W} \in \mathbb{R}^{N \times N}$. When considering a *signal function* associated with the vertices of the graph, which can be represented by a vector in $\mathbb{R}^N$, $\mathbf{L}$ works then as a difference operator. Laplacian is associated with the smoothness of a function, and thus this matrix may give an idea of the variation of the signal from one vertex to another vertex that is connected to it by an edge.

First, when we suggest variations between vertices we hint the idea of a function that maps each vertex into a numerical value (which is precisely the signal function) and the derivative of this function. With respect to graphs the directions in which the signal may vary are thus determined by the edges, so this leads to think about the differences of the signal function evaluated in vertices along the edges. Now this seems as a variation flux trough the edges and is what's reduced to the definition of graph Laplacian. As a linear operator acting over vertices it will hence satisfy

$$(\mathbf{L}f)(i) = \sum_j W_{i,j}\,(f(i) - f(j))$$

for a signal function f and considering j as indexing those vertices connected with vertex i by an edge. When evaluating it as a quadratic form

$$\frac{1}{2}\sum_j W_{i,j}\,(f(i) - f(j))^2 = f^\top \mathbf{L} f$$

it becomes clearer the present relation to the concept of smoothness and also arises the important observation of the eigenvectors of the graph Laplacian as those that minimize the expression.

Notice that, as we are interested in signals represented by undirected graphs, the adjacency matrix will be symmetric and therefore the graph Laplacian is real symmetric. Even more, it can also be verified that the graph Laplacian is positive semidefinite, therefore not only it has N eigenvectors which are ortonormal but it has all non-negative eigenvalues. Again, as for the adjacency matrix, for a disconnected graph the graph Laplacian can be built up as a block diagonal matrix by the graph Laplacian of each connected component. Additionally, it can be observed that the constant vector $\mathbf{1}_N$ has eigenvalue 0 and is the only one too. Therefore, the corresponding eigenvalues $\{\lambda_k\}_{k=0}^{N-1}$ can be written in an ordered manner $0 = \lambda_0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{N-1}$ with a set of N distinct vectors $\{\mathbf{u}_k\}_{k=0}^{N-1}$.

3.3 Graph Fourier Transform

In view of the signal $\mathbf{f} \in \mathbf{R}^N$ of the graph as a function $f : \mathcal{V} \to \mathbb{R}$ related to the vertices of $\mathcal{G}$, we can extend the definition of the classical Fourier Transform (1), which can be understood over a complete ortonormal system of eigenfunctions, to a *graph Fourier Transform* using precisely the graph Laplacian. Hence, the expansion of the function, in terms of the λ_k eigenvalues and corresponding $\mathbf{u}_k$ eigenvectors, will take the form

$$\hat{f}(\lambda_k) := \langle \mathbf{f}, \mathbf{u}_k \rangle = \sum_{i=1}^{N} f(i) u_k^*(i). \tag{2}$$

The eigenvectors of the graph Laplacian subsequently form the graph Fourier basis, and the eigenvalues, similarly to those for the classical Fourier Transform, represent frequencies. As mentioned before this entails a notion of smoothness, for the smaller the eigenvalue is, the lower the frequency and thus a fewer variation of the values related to connected vertices. Different eigenvector act for different frequency modes of the graph signal. In this way, the larger the eigenvalue, the higher the oscillation of the corresponding vertex values.

From the above, the *inverse graph Fourier transform* turns out as follows:

$$f(i) = \sum_{k=0}^{N-1} \hat{f}(\lambda_k) u_k(i). \tag{3}$$

These generalizations will allow the definition of different operators applied over graph signals.

4 Processing on graphs

When referring to signal processing algorithms operations regarding graphs result fundamental. For the definitions of these the prior extension of the Fourier Transform and its preserving properties come in handy.

4.1 Filtering

When we have a signal that presents some altering information, such as noise that can be seen as specific intensities at some frequencies, one thing that can be done is decomposing the signal and fluctuating the weight or contribution of each component. In this way the noise intensity can be reduced by attenuating its corresponding components within the signal.

To do this a transfer function is used along with the input signal, so the signal will be represented in terms of complex exponentials multiplied in a way such that its contributions change accordingly to what is desired. As the graph Fourier Transform allows us to do with with frequencies, this can be generalized by its graph extension as the output signal $\hat{f}(\lambda_k)\hat{t}(\lambda_k)$ where $\hat{t}$ is the mentioned transfer function. What is done here is a convolution over the time domain given the inverse Fourier Transform when considering multiplication in a frequency domain. Accordingly, in view of filtering a input graph signal into an output graph signal using the inverse graph Fourier Transform (3) the graph filtering can be then defined as

$$f_{out}(i) = \left(\sum_{k=0}^{N-1} \hat{f}_{in}(\lambda_k) u_k(i) \right) \hat{t}(\lambda_k) = \sum_{k=0}^{N-1} \hat{f}_{in}(\lambda_k)\hat{t}(\lambda_k) u_k(i), \tag{4}$$

expressing precisely

$$\hat{f}_{out} = \hat{f}_{in}(\lambda_k)\hat{t}(\lambda_k).$$

This means the implementation of the filter implies finding the eigenvectors $\{\mathbf{u}_k\}$ of the graph Laplacian, adjusting the frequencies in the eigenvector space by means of pointwise multiplication, and projecting the data back from the eigenvector space with the inverse graph Fourier transform where generated values through filtering are evidenced in the vertex space. This reduces filtering to multiplication with graph Laplacian matrix.

5 Tikhonov regularization

Here we illustrate a common inverse problem: deblurring an image in the case where the blur kernel is known, and in the presence of noise. The easiest and simplest method for solving some common inverse problems is to use the Tikhonov regularization method, which is a regularized least squares formulation. We observe a noisy graph signal $y = f_0 + \eta$, where η is uncorre-lated additive Gaussian noise, and we wish to recover f_0. To enforce a priori information that the clean signal f_0 is smooth with respect to the underlying graph, we include a regularization term of the form $f^\top \mathbf{L} f$, and, for a fixed $\gamma > 0$, solve the optimization problem

$$\arg\min_f \{\|f - y\|_2^2 + \gamma f^\top \mathbf{L} f\}. \tag{5}$$

To show this, we use the code from [4] which is adapted in [2], firts we have the input image.

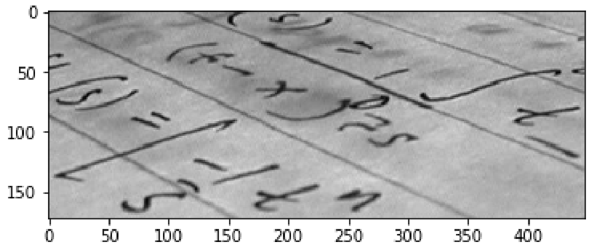

Figure 4: Input image that we are going to use in the algorithm.

The matrix form of the image can be used as the signal f_0, in that order deblurring an image is an ill-posed problem, in the sense that the matrix inversion or least-square approach is very sensitive to noise. To propose a less noise-sensitive approach to the problem, we consider the following regularized least-squares problem

$$\arg\min_x \{\|Hx - y\|_2^2 + \|\Gamma x\|_2^2\}. \tag{6}$$

where x is the sought vector, y the corrupted, observed vector, H the degradation matrix, and Γ the regularisation prior. Γ is most commonly chosen to be I, the identity matrix, but can be set to another, more favourable one, like one corresponding to the gradient operator ∇ or the Laplacian Δ.
This problem is convex quadratic and so an explicit solution exist, given by the regu-

larized normal equation

$$(H^\top H + \Gamma^\top \Gamma)x = H^\top y. \tag{7}$$

The code in [2] solves the regularized normal equation using a conjugate gradient algorithm. This is only an acceptable solution in the case where H and Γ are both mostly diagonal. In this case $H^\top H + \Gamma^\top \Gamma$ remains sparse, and is definite positive. In any other case, an algorithm that does not require computing this matrix is much preferable.

We can attempt to deblur the image without regularization. This only makes sense if there is a tiny amount of noise. Further, this is only possible on tiny images, because solving ill-conditioned problems is much more costly than well-conditioned ones.

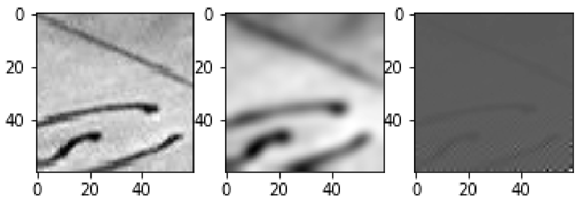

Figure 5: Debluring of a zoomed image of the input without regularization, in the left we can see the input zoomed image, in the middle the blured imaged and in the right the deblured image using (7).

Clearly, the result is not fantastic. If we add more noise, the resulting deblurred result becomes completely dominated by noise. This is due to the fact that the blur matrix is ill-conditioned.

Now we test with another portion of the input image, with a lot more noise, and the Tikhonov regularization.

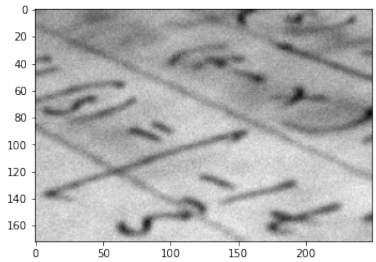

Figure 6: Blurred and noisy image. The noise level is significant and more than 10^5. We use the Laplacian prior $\Gamma = \lambda\Delta$. The value 0.025 is the value of λ.

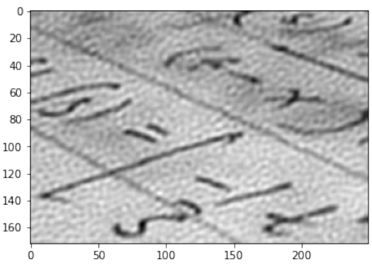

Figure 7: Deblurred image by Tikhonov regularization.

The result is not perfect, in particular there is some ringing in the flat areas of the image. However, the result is sharper than the blurred image, and most importantly, it is stable.

6 Conclusions

We can see that the emerging field of signal processing using graphs has a lot of applications in the industry and study how we can extend theory that we use in other fields may improve this theory and give the possibility to extend the applications and the utility of it.
Deblurring images is a problem that appears a lot and have more alternatives to solve that kind of problems is very important, stability allow us to strengthen the solutions that we can obtain and this theory give to us the necessary tools to do that.

References

[1] T. Khare, *Introduction of Fast Fourier Transformation (FFT).* https://ai-pool.com/a/s/introduction-of-fast-fourier-transformation–fft.

[2] A. Rubiano, *Tikhonov-regularization.* https://github.com/aerubianoma/Codification-theory/Final-project, 2022.

[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, 30 (2013), pp. 83–98.

[4] H. Talbot, *Image deblurring with Tikhonov regularization.* http://news.zahlt.info/en/optimization/image-deconvolution-using-tikhonov-regularization/.