Stationary signal processing on graphs

Nathanaël Perraudin and Pierre Vandergheynst [*]

May 23, 2016

Abstract

Graphs are a central tool in machine learning and information processing as they allow to conveniently capture the structure of complex datasets. In this context, it is of high importance to develop flexible models of signals defined over graphs or networks. In this paper, we generalize the traditional concept of wide sense stationarity to signals defined over the vertices of arbitrary weighted undirected graphs. We show that stationarity is intimately linked to statistical invariance under a localization operator reminiscent of translation. We prove that stationary graph signals are characterized by a well-defined Power Spectral Density that can be efficiently estimated even for large graphs. We leverage this new concept to derive Wiener-type estimation procedures of noisy and partially observed signals and illustrate the performance of this new model for denoising and regression.

Index terms— Stationarity, graphs, spectral graph theory, power spectral density, Wiener filter, covariance, Gaussian random fields

1 Introduction

Stationarity is a traditional hypothesis in signal processing used to represent a special type of statistical relationship between samples of a temporal signal. For instance, wide-sense stationarity assumes that the first two statistical moments are invariant under translation. In other words, the correlation between two samples depends only on their time difference. Stationarity is a corner stone of many signal analysis methods. The expected frequency content of stationary signals, called Power Spectral Density (PSD), provides an essential source of information used to build signal models, generate realistic surrogate data or perform predictions. In Figure 1, we present an example of a stationary process (blue curve) and two predictions (red and green curves). As the blue signal is a realization of a stationary process, the red curve is more probable than the green one because it respects the frequency content of the observed signal. Classical stationarity is a statement of statistical regularity under arbitrary translations and thus requires a regular structure (often "time"). However many signals do not live on such a regular structure. Imagine that instead of having one sensor returning a temporal signal, we have multiple sensors living in a two-dimensional space, each of which delivers only one value. In this case (see Figure 2 left), the

[*]EPFL, Ecole Polytechnique Fédérale de Lausanne, LTS2 Laboratoire de traitement du signal, CH-1015 Lausanne, Switzerland

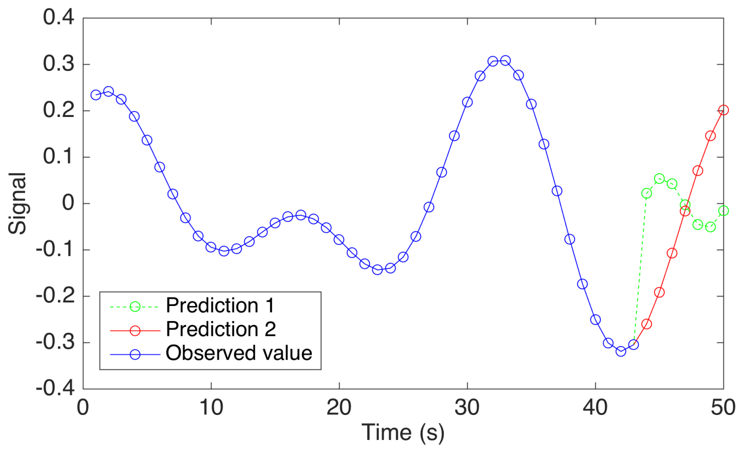


Figure 1: Signal prediction. The red curve is more likely to occur than the green curve because it respects the frequency statistics of the blue curve.

signal support is no longer regular. Since there exists an underlying continuum in this example (2D space), one could assume the existence of a 2D stationary field and use Kriging [1] to interpolate observations to arbitrary locations. This technique thus generalizes stationarity for a regular domain but irregularly spaced samples. The goal of this contribution is to generalize stationarity for an irregular domain that is represented by a graph, without resorting to any underlying regular continuum. Instead we use a weak notion of translation invariance that captures the structure (if any) of the samples set. Whereas classical stationarity means correlations are computed by translating the auto-correlation function, here correlations are given by localising a common graph kernel, which is a generalised notion of translation as detailed in [2]. Graphs are convenient data structures able to capture complicated topologies and are used in numerous domains of applications such as social, transportation or neuronal networks. In this work a graph is composed of vertices connected by weighted undirected edges. As we shall recall, a graph possess a localization operator that generalizes the classical translation. Signals are now scalar values observed at the vertices of the graph.

Figure 2 (left) presents an example of some data living in a 2-dimensional space. Seen as scattered samples of an underlying 2D process, one would (rightly) conclude it is not stationary. However, under closer inspection, the observed values look stationary *within* the spiral-like structure depicted by the graph in Figure 2 (right). The traditional Kriging interpolation technique would ignore this underlying structure and conclude that there are always rapid two dimensional variations in the underlying continuum process. This problem does not occur in the graph case where the statistical relationships inside the data follow the graph edges resulting in this case in signals oscillating smoothly over the graph. A typical example of a stationary signal on a graph would be the result of a survey performed by the users of a social network. If there is a relationship between a user's answers and those of his neighbours, this relationship is expected to be constant among all users. Using stationarity on the graph, we

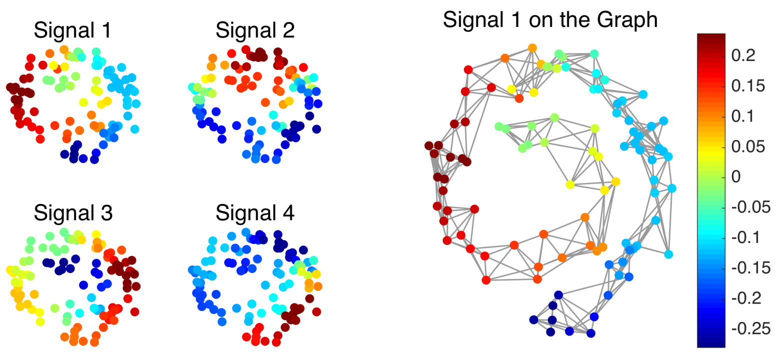


Figure 2: Example of a stationary process on a graph. The graph connections express relationships between the different elements of one signal. In this case, the signal varies smoothly along the snail shape of the graph.

could predict the most probable answer for users that never took the survey.

1.1 Contributions

We use spectral graph theory to extend the notion of stationarity to a broader class of signals. To do so, we establish in Section 3 the theoretical basis of this extension. In order to evaluate the usefulness of our new model, we generalize Wiener filters [3, 4].

Since Wiener filters use the power spectral density (PSD), we generalize the Welch method [5, 6] in Section 4 and obtain a scalable and robust way to estimate the PSD. It improves largely the covariance estimation when the number of signals is limited.

Based on our generalization of Wiener filters, we propose a new regularization term for graph signal optimization instead of the traditional Dirichlet prior. This term depends on the noise level and on the PSD of the signal. The new optimization scheme presented in Section 5 has three main advantages: 1) it allows to deal with an arbitrary regularization parameter, 2) it adapts to the data optimally as we prove that the optimization model is a Maximum A Posteriori (MAP) estimator, and 3) it is more scalable and robust than a traditional Gaussian estimator.

Finally, in Section 6, we show experimentally that common datasets closely follow our stationarity assumption. In section 7, we exploit this fact to perform missing data imputation and we show how stationarity improves over classical graph models and Gaussian MAP estimator.

1.2 Related work

Graphs have been used for regularization in data applications for more than a decade [7, 8, 9, 10] and two of the most used models will be presented in Section A. The idea of graph filtering was hinted at by the machine learning community [11] but developed for the spectral graph wavelets proposed by Hammond et al. [12] and extended by Shuman et al. in [2]. While in most cases, graph filtering is based on the graph Laplacian, Moura et al. [13] have suggested to use the adjacency matrix instead.

A notion of stationarity on graphs has been recently proposed in [14, 15]. However these contributions use a completely different translation operator, promoting energy perservation over localization, and therefore end up with a fairly different framework. Finally, we also note that a probabilistic model using Gaussian random fields has been proposed in [16, 17]. In this model, signals are automatically graph stationary with an imposed covariance matrix. Our model differentiates itself from these contributions because it is based on a much less restrictive hypothesis and uses the point of view of stationarity. A detailed explanation is given at the end of Section 3.

2 Background theory

2.1 Graph signal processing

Graph nomenclature A graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$ is defined by two sets: $\mathcal{V}, \mathcal{E}$ and a weight function $\mathcal{W}$. $\mathcal{V}$ is the set of vertices representing the nodes of the graph and $\mathcal{E}$ is the set of edges that connect two nodes if there is a particular relation between them. In this work all graphs are undirected. To obtain a finer structure, this relation can be quantified by a weight function $\mathcal{W} : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$ that reflects to what extent two nodes are related to each other. Let us index the nodes from $1, \dots, N = |\mathcal{V}|$ and construct the weight matrix $W \in \mathbb{R}^{N \times N}$ by setting $W[i,j] = \mathcal{W}(v_i, v_j)$ as the weight associated to the edge connecting the node i and the node j. When no edge exists between i and j, the weight is set to 0. For a node $v_i \in \mathcal{V}$, the degree $d[i]$ is defined as $d[i] = \sum_{j=1}^{N} W[i,j]$. In this framework, a signal is defined as a function $f : \mathcal{V} \rightarrow \mathbb{R}$ (or $\mathbb{C}$) assigning a scalar value to each vertex. It is convenient to consider a signal f as a vector of size N with the n^{th} component representing the signal value at the n^{th} vertex.

The most fundamental operator in graph signal-processing is the (combinatorial) graph Laplacian, defined as:

$$L = D - W,$$

where D is the diagonal degree matrix ($D[i,i] = d[i]$).

Spectral theory Since the Laplacian L is always a symmetric positive semi-definite matrix, we know from the spectral theorem that it possesses a complete set of orthonormal eigenvectors. We denote them by $\{u_\ell\}_{\ell=0,1,\dots,N-1}$. For convenience, we order the set of real, non-negative eigenvalues as follows: $0 = \lambda_0 < \lambda_1 \leq \cdots \leq \lambda_{N-1} = \lambda_{\max}$. When the graph is connected[1], there is only one zero eigenvalue. In fact, the multiplicity of the zero eigenvalue is equal to the number of connected components. For more details on spectral graph theory, we refer the reader to [18, 19]. The eigenvectors of the Laplacian are used to define a graph Fourier basis [7, 2] which we denote as U. The eigenvalues are considered as a generalization of squared frequencies. The Laplacian matrix can thus be decomposed as

$$L = U \Lambda U^*,$$

[1]a path connects each pair of nodes in the graph

where U^* denotes the transposed conjugate of U. The graph Fourier transform is written $\hat{f} = \mathcal{F}\{f\} = U^* f$ and its inverse $f = \mathcal{F}^{-1}\{\hat{f}\} = U\hat{f}$. This Graph Fourier Transform possesses interesting properties further studied in [2].

Graph convolutive filters The graph Fourier transform plays a central role in graph signal processing since it allows a natural extension of filtering operations. In the classical setting, applying a filter to a signal is equivalent with a convolution, which is simply a point-wise multiplication in the spectral domain. For a graph signal, where the domain is not regular, filtering is still well defined as a point-wise multiplication in the spectral domain A graph filter g is defined as a continuous function $g : \mathbb{R}_+ \to \mathbb{R}$. In the spectral domain, filtering a signal f with a filter g is therefore written as $\hat{f}'[\ell] = g(\lambda_\ell) \cdot \hat{f}[\ell]$ or in matrix notation

$$f' = Ug(\Lambda)U^* f = g(L)f,$$

where $\hat{f}'$ is the Fourier transform of the filtered signal f', g the filter and $\hat{f}$ the Fourier transform of the signal f. Equivalently, a filter defines the following matrix-valued function:

$$g(L) := Ug(\Lambda)U^*,$$

where $g(\Lambda)$ is a diagonal matrix with entries $g(\lambda_\ell)$. A comprehensive definition and study of these operations can be found in [2]. It is worth noting that these formulas make explicit use of the Laplacian eigenvectors and thus its diagonalization. The complexity of this operation is in general $\mathcal{O}(N^3)$. In order to avoid this cost, there exist fast filtering algorithms based on Chebyshev polynomials or the Lanczos method [12, 20]. These methods scale with the number of edges $|E|$ and reduce the complexity to $\mathcal{O}(|E|)$, which is advantageous in the case of sparse graphs.

Localization operator Because most graphs do not possess a regular structure, we cannot translate a signal around the vertex set. This is problematic since translation plays a central role in stationarity. In order to overcome this issue, we use the localization operator defined in [12, 2], which is simply the convolution with a Kroneker delta. Localizing a filter g onto node i reads:

$$\mathcal{T}_i g[n] = \sum_{\ell=0}^{N-1} g(\lambda_\ell) u_\ell^*[i] u_\ell[n] = (g(L))_{in}, \tag{1}$$

where u_ℓ^* is the (transposed) complex conjugate of u_ℓ. For a sufficiently regular function g, it has been proved in [2] that this operation localizes the filter g around the vertex i. In the classical case, the concept of translation is equivalent to localization since the convolution with a Kroneker delta is precisely a translation. However, for irregular graphs, localization differs from translation because the shape of the localized filter adapts to the graph and varies as a function of its topology. Insights about the localization operator can be found in [2, 12, 21].

2.2 Stationarity for temporal signals

Let $\mathbf{x}(t)$ be a time indexed stochastic process. Throughout this paper, all random variables are written in bold fonts. We use $m_{\mathbf{x}} = \mathbb{E}\{\mathbf{x}\}$ to denote the arithmetic mean of $\mathbf{x}$.

Definition 1 (Time Wide-Sense Stationarity). *A signal is Time Wide-Sense Stationary (WSS) if its first two statistical moments are invariant under translation, i.e:*

1. $m_{\mathbf{x}}(t) = \mathbb{E}\big\{\mathbf{x}(t)\big\} = c \in \mathbb{R}$,
2. $\mathbb{E}\big\{(\mathbf{x}(t) - m_{\mathbf{x}})(\mathbf{x}(s) - m_{\mathbf{x}})^*\big\} = \gamma_{\mathbf{x}}(t-s)$,

where $\gamma_{\mathbf{x}}$ is called the autocorrelation function of $\mathbf{x}$.

For simplicity, we use $m_{\mathbf{x}}$ instead of $m_{\mathbf{x}}(t)$ when $m_{\mathbf{x}}(t)$ is constant. For a WSS signal, the autocorrelation function depends only on one parameter, $t-s$, and is linked to the Power Spectral Density (PSD) through the Wiener-Khintchine Theorem [3]. The latter states that the PSD of the stochastic process $\mathbf{x}$ denoted $S_{\mathbf{x}}(\omega)$ is the Fourier transform of its auto-correlation :

$$S_{\mathbf{x}}(\omega) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \gamma_{\mathbf{x}}(t) e^{-i\omega t} \mathrm{d}t. \tag{2}$$

As a consequence, when a signal is convolved with a filter, its PSD is multiplied by the energy of the filter: for $\mathbf{y} = h * \mathbf{x}$, we have

$$S_{\mathbf{y}}(\omega) = \left|\hat{h}(\omega)\right|^2 S_{\mathbf{x}}(\omega),$$

where $\hat{h}(\omega)$ is the Fourier transform of the filter. For more information about stationarity, we refer the reader to [22].

Note that equivalent definitions exist for continuous, discrete and periodic signals. When generalizing these concepts to graphs, the underlying structure for the stationarity will no longer be time but graph vertices.

3 Stationarity of graph signals

Let $\mathbf{x} \in \mathbb{R}^N$ be a stochastic process with a finite number of variables indexed by the vertices of a weighted undirected graph. The expected value of each variable is written $m_{\mathbf{x}}[i] = \mathbb{E}\big\{\mathbf{x}[i]\big\}$ and the covariance matrix of the process is $\Sigma_{\mathbf{x}} = \mathbb{E}\big\{(\mathbf{x} - m_{\mathbf{x}})(\mathbf{x} - m_{\mathbf{x}})^*\big\})$. From N_s realizations x_n of the process $\mathbf{x}$, the covariance $\Sigma_{\mathbf{x}}$ can be estimated empirically with no information about the graph itself via the sample covariance:

$$\Sigma_{\mathbf{x}}[i,j] \approx \frac{1}{N_s} \sum_{n=1}^{N_s} (x_n[i] - \overline{x}_n)(x_n[j] - \overline{x}_n)^*,$$

where $\overline{x}_n$ is the arithmetic mean of x_n. For discrete time WSS processes, the covariance matrix $\Sigma_{\mathbf{x}}$ is Toeplitz, or circulant for periodic boundary conditions, reflecting translation invariance. In that case, the covariance is diagonalized by the Fourier transform. We now generalize this property to take into account the intricate graph structure.

Definition 2. *A stochastic process $\mathbf{x}$ defined on the vertices of a graph $\mathcal{G}$ is called Graph Wide-Sense (or second order) Stationary (GWSS), if and only if it satisfies the following properties :*

1. *its first moment is constant over the vertex set, i.e* $m_{\mathbf{x}}[i] = \mathbb{E}\{\mathbf{x}[i]\} = c \in \mathbb{R}$ *and*

2. *its covariance matrix* $\Sigma_{\mathbf{x}}[i,j] = \mathbb{E}\{(\mathbf{x}[i] - m_{\mathbf{x}})(\mathbf{x}[j] - m_{\mathbf{x}})\}$ *is jointly diagonalizable with the Laplacian of* $\mathcal{G}$*, i.e:* $\Sigma_{\mathbf{x}} = U\Gamma_{\mathbf{x}}U^*$*, where* $\Gamma_{\mathbf{x}}$ *is a diagonal matrix.*

The first part of the above definition is equivalent to the first property of time WSS signal. The requirement for the second moment is a natural generalization where we are imposing the use of the graph Fourier transform but it seems to obscure any notion of translation invariance. We shall now see that this condition is equivalent to imposing that the covariance is invariant under the localization operator. The easiest way to observe this fact is to generalize the Wiener-Khintchine Theorem 2, which makes the link between the localization operator and the covariance matrix.

Theorem 1. *If a signal is GWSS, its covariance matrix is given by graph localization of a kernel* $\gamma_{\mathbf{x}}$*, i.e:*

$$\Sigma_{\mathbf{x}}[i,j] = \mathcal{T}_i\gamma_{\mathbf{x}}(j) \tag{3}$$

Proof. By Definition 2, the covariance matrix of a GWSS signal can be written as:

$$\Sigma_{\mathbf{x}} = U\Gamma_{\mathbf{x}}U^* \tag{4}$$

where $\Gamma_{\mathbf{x}}$ is a diagonal matrix. For any filter $\gamma_{\mathbf{x}}$ satisfying $\gamma_{\mathbf{x}}(\lambda_\ell) = \Gamma_{\mathbf{x}}(\ell,\ell)$, we observe that $\Sigma_{\mathbf{x}} = \gamma_{\mathbf{x}}(L)$ is the graph localized version of $\gamma_{\mathbf{x}}$. □

The choice of $\gamma_{\mathbf{x}}$ in this result is somewhat arbitrary but we shall soon see that we are interested in localized kernels. In that case, $\gamma_{\mathbf{x}}$ will be typically be the lowest degree polynomial satisfying the constraints and can be constructed using Lagrange interpolation for instance.

Theorem 1 provides a fundamental property of the covariance. The size of the correlation (distance over the graph) depends on the support of localized the kernel $\mathcal{T}_i\gamma_{\mathbf{x}}$. In [2, Theorem 1 and Corollary 2], it has been proved that the concentration of $\mathcal{T}_i\gamma_{\mathbf{x}}$ around i depends on the regularity[2] of $\gamma_{\mathbf{x}}$. For example, if $\gamma_{\mathbf{x}}$ is polynomial of degree K, it is exactly localized in a ball of radius K. Hence we will be mostly interested in such low degree polynomial kernels.

The graph spectral covariance matrix of a stochastic process is given by $\Gamma_{\mathbf{x}} = U^*\Sigma_{\mathbf{x}}U$. For a GWSS signal this matrix is diagonal and the graph power spectral density (PSD) of $\mathbf{x}$ becomes:

$$\gamma_{\mathbf{x}}(\lambda_\ell) = (U^*\Sigma_{\mathbf{x}}U)_{\ell,\ell}\,. \tag{5}$$

Table 1 present the difference and the similarities between the classical and the graph case. For a regular cyclic graph (ring), the localization operator is equivalent to the traditional translation and we recover the classical cyclo-stationarity results. Our framework is thus a generalization of stationarity to irregular domains.

[2]A regular kernel can be well approximated by a smooth function, for instance a low order polynomial, over spectrum of the laplacian.

Example 1 (Gaussian i.i.d. noise)**.** *Normalized Gaussian i.i.d. noise is GWSS for any graph. Indeed, the first moment is $\mathbb{E}\big(\mathbf{x}[k]\big) = 0$. Moreover, the covariance matrix can be written as $I = \Sigma_{\mathbf{x}} = UIU^*$ with any orthonormal matrix U and thus is diagonalizable with any graph Laplacian. We also observe that the PSD is constant, which implies that similar to the classical case, white noise contains all "graph frequencies".*

When $\gamma_{\mathbf{x}}$ is a bijective function, the covariance matrix contains an important part of the graph structure: the laplacian eigenvectors. On the contrary, if $\gamma_{\mathbf{x}}$ is not bijective, some of the graph structure is lost as it is not possible to recover all eigenvectors. This is for instance the case when the covariance matrix is low-rank. As another example, let us consider completely uncorrelated samples. In this case, the covariance matrix becomes $\Sigma_{\mathbf{x}} = I$ and loses all graph information, even if by definition the process remains stationary on the graph.

One of the crucial benefits of stationarity is that it is preserved by filtering, while the PSD is simply reshaped by the filter. The same property holds on graphs.

Theorem 2. *When a graph filter g is applied to a GWSS process, the result remains GWSS and the PSD satisfies:*

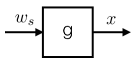
$$\gamma_{g(L)\mathbf{x}}[\ell] = g^2(\lambda_\ell) \cdot \gamma_{\mathbf{x}}[\ell]. \tag{6}$$

Proof. The output of a filter g can be written as $\mathbf{x}' = g(L)\mathbf{x}$. If the input signal $\mathbf{x}$ is GWSS, we can check easily that the first moment of the filter's output is constant, $\mathbb{E}(g(L)\mathbf{x}[i]) = g(0)m_{\mathbf{x}}$. The computation of the second moment gives:

$$\begin{aligned} \mathbb{E}\left(g(L)x\big(g(L)x\big)^*\right) &= g(L)\mathbb{E}\left(xx^*\right)g(L) \\ &= g(L)\Sigma_{\mathbf{x}}g(L) \\ &= Ug^2(\Lambda)\gamma_{\mathbf{x}}(\Lambda)U^*. \end{aligned}$$

□

Theorem 2 provides a simple way to artificially produce stationary signals with a prescribed PSD by simply filtering white noise :



The resulting signal will be stationary with PSD g^2. In the sequel, we assume for simplicity that the signal is centered at 0, i.e: $m_{\mathbf{x}} = 0$. Note that the input white noise could well be non-Gaussian.

Gaussian random field interpretation The framework of stationary signals on graphs can be interpreted using Gaussian Random Field (GRF). Let us assume that the signal $\mathbf{x}$ is drawn from a distribution

$$\mathbb{P}(\mathbf{x}) = \frac{1}{Z_p}\exp\left(-\mathbf{x}^T p(L)\mathbf{x}\right), \tag{7}$$

where $Z_p = \int_{\mathbb{R}^N} \exp\left(-\mathbf{x}^T p(L)\mathbf{x}\right) \mathrm{d}\mathbf{x}$. If we assume that $p(L)$ is invertible, drawing from this distribution will generate a stationary $\mathbf{x}$ with covariance matrix given by:

$$\Sigma_{\mathbf{x}} = (p(L))^{-1} = p^{-1}(L).$$

	Classical	Graph
Stationary with respect to	Translation	The localization operator
First Moment	$\mathbb{E}\big(\mathbf{x}[i]\big) = m_{\mathbf{x}} = c \in \mathbb{R}$	$\mathbb{E}\big(\mathbf{x}[i]\big) = m_{\mathbf{x}} = c \in \mathbb{R}$
Second Moment	$\Sigma_{\mathbf{x}}[i,n] = \mathbb{E}\big(\tilde{\mathbf{x}}[i])\tilde{\mathbf{x}}^*[n]\big) = \gamma_{\mathbf{x}}[t-s]$	$\Sigma_{\mathbf{x}}[i,n] = \mathbb{E}\big(\tilde{\mathbf{x}}[i])\tilde{\mathbf{x}}^*[n]\big) = \gamma_{\mathbf{x}}(L)_{i,n}$
(We use $\tilde{\mathbf{x}} = \mathbf{x} - m_{\mathbf{x}}$)	$\Sigma_{\mathbf{x}}$ Toeplitz	$\Sigma_{\mathbf{x}}$ diagonalizable with L
Wiener Khintchine	$S_{\mathbf{x}}[\ell] = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} \gamma_{\mathbf{x}}[n] e^{-j2\pi \frac{n\ell}{N}}$	$\gamma_{\mathbf{x}}(\lambda_\ell) = (\Gamma_{\mathbf{x}})_{\ell,\ell} = (U^* \Sigma_{\mathbf{x}} U)_{\ell,\ell}$
Result of filtering	$\gamma_{g*\mathbf{x}}[\ell] = g^2(\lambda_\ell) \cdot \gamma_{\mathbf{x}}[\ell]$	$\gamma_{g(L)\mathbf{x}}[\ell] = g^2(\lambda_\ell) \cdot \gamma_{\mathbf{x}}[\ell]$

Table 1: Comparison between classical and graph stationarity. In the classical case, we work with a N periodic discrete signal.

In other words, assuming a GRF probabilistic model with inverse covariance matrix $p(L)$ leads to a stationary process with a PSD $= p^{-1}$. However a stationary process is not necessarily a GRF. Indeed, stationarity assumes statistical properties on the signal that are not necessarily based on Gaussian distribution.

In Section 3 of [16], Gadde and Ortega have presented a GRF model for graph signals. But they restrict themselves to the case where $p(L) = L + \delta I$. Following a similar approach [17] links the inverse covariance matrix of a Gaussian random process with the Laplacian. Our approach is much broader than these two contributions since we do not make any assumption on the function $p(L)$. Finally we exploit properties of stationary signals, such as the characterization of the PSD, to explicitly solve signal processing problems in Section 5.

4 Estimation of the signal PSD

As the PSD is central in our method, we need a reliable and scalable way to compute it. Equation (5) suggests a direct estimation method using the Fourier transform of the covariance matrix. Unfortunately, when the number of nodes is considerable, this method requires the diagonalization of the Laplacian, an operation whose complexity in the general case scales as $O(N^3)$ for the number of operations and $O(N^2)$ for memory requirements. Additionally, when the number of available realizations of the process is small, it is not possible to obtain a good estimate of the covariance matrix. To overcome these issues, inspired by Bartlett [6] and Welch [5], we propose to use a graph generalization of the Short Time Fourier transform [2] to construct a scalable estimation method.

Bartlett's method can be summarized as follows. The signal is first cut into equally sized segments without overlap. Then, the Fourier transform of each segment is computed. Finally, the PSD is obtained by averaging over segments the squared amplitude of the Fourier coefficients. Welch's method is a generalization that works with overlapping segments.

A direct generalization from the time domain to the vertex domain of these methods is not trivial for two reasons. First it needs to partition the signal in the vertex domain, and second, it requires the Fourier transform of the signal, that is not scalable.

On the other hand, we can see the PSD estimation of both methods as the averaging over time of the squared coefficients of a short Fourier transform. Our method is based on this idea, using the windowed graph Fourier transform [2]. Instead of a translated rectangular window in time, we use a kernel g, localized at each individual node of the graph. This kernel is then shifted by multiples of a step τ in the spectral domain, i.e.

$$g_m(\lambda_\ell) = g(\lambda_\ell - m\tau), \qquad m = 1 \dots M, \quad \tau = \frac{\lambda_{\max}}{M}.$$

The coefficients of the graph windowed Fourier transform can be seen as a matrix with elements

$$C[i,m] = \left[g_m(L)x\right]_i.$$

Our algorithm consists in averaging the squared coefficients of this transform over the vertex set. Because graphs have an irregular spectrum, we additionally need a normalization factor which is given by the norm of the window g_m: $\|g_m\|_2^2 = \sum_\ell g(\lambda_\ell - m\tau)^2$. Note that this norm will vary for the different m. Our final estimator reads :

$$\dot{\gamma_x}(m\tau) = \frac{\|g_m(L)x\|_2^2}{\|g_m\|_2^2} = \frac{\sum_{i=1}^N C[i,m]^2}{\|g_m\|_2^2},, \tag{8}$$

where x is a single realization of the stationary process $\mathbf{x}$. Studying the bias of (8) reveals its interest :

$$\frac{\mathbb{E}\left(\|g_m(L)\mathbf{x}\|_2^2\right)}{\|g_m\|_2^2} = \frac{\sum_{\ell=0}^{N-1}\left(g(\lambda_\ell - m\tau)\right)^2 \gamma_{\mathbf{x}}(\lambda_\ell)}{\sum_{\ell=0}^{N-1}\left(g(\lambda_\ell - m\tau)\right)^2}, \tag{9}$$

where $\mathbf{x}$ is the graph stationary process. For a filter g well concentrated at the origin, (9) gives a smoothed estimate of $\gamma_{\mathbf{x}}(m\tau)$. This smoothing corresponds to the windowing operation in the vertex domain: the less localized the kernel g in the spectral domain, the more pronounced the smoothing effect in (9) and the more concentrated the window in the vertex domain. It is very interesting to note we recover the traditional trade-off between bias and variance in non-parametric spectral estimation. Indeed, if g is very sharply localized on the spectrum, ultimately a Dirac delta, the estimator (8) is unbiased. Let us now study the variance. Intuitively, if the signal is correlated only over small regions of the vertex set, we could isolate them with localized windows of a small size and averaging those uncorrelated estimates together would reduce the variance. These small size windows on the vertex set correspond to wide filters g_m and therefore large bias. However, if those correlated regions are large, and this happens when the PSD is localized in low-frequencies, we cannot hope to benefit from vertex-domain averaging since the graph is finite. Indeed the corresponding windows g_m on the vertex set are so large that a single window spans the whole graph and there is no averaging effect: the variance increases precisely when we try to suppress the bias.

Our complete estimation procedure is as follows. First, we design a filterbank by choosing a mother function g (for example a Gaussian $g(\lambda) = e^{-\lambda^2/\sigma^2}$). A frame is then created by shifting uniformly M times g in the spectral domain: $g_m(\lambda) = g(\lambda - m\tau) = e^{-(\lambda - m\tau)^2/\sigma^2}$. Second, we compute the estimator $\widehat{\gamma_x}(m\tau)$

from the stationary signal $\mathbf{x}$. Note that if we have access to K realizations of the stationary process, we can of course average them to further reduce the variance using $\mathbb{E}\left(\|g_m(L)\mathbf{x}\|_2^2\right) \approx 1/K \sum_k \|g_m(L)x_k\|_2^2$. Third we use the following trick to quickly approximate $\|g_m\|_2^2$. Using K_2 randomly-generated Gaussian white signals, we estimate

$$\mathbb{E}\left(\|g_m(L)\mathbf{w}\|_2^2\right) = \|g_m\|_2^2.$$

Finally, the last step consists in computing the ratio between the two quantities and interpolating the discrete points $(k\tau, \big(g * \gamma\big)(m\tau))$.

Experimental assessment of the method Figure 3 shows the results of our PSD-estimation algorithm on a 10-nearest neighbors graph of 20′000 nodes (sensor type) and only $K = 1$ realization of the stationary process. We compare the estimation using frames of 10, 30, 100, 500 Gaussian filters. The parameters σ and τ are adapted to the number of filters ($\tau = \sigma^2 = \frac{(m+1)\lambda_{\max}}{m^2}$). For this experiment K_2 is set to 4 and the Chebysheff polynomial order is 30 (Except for $m = 500$ where we took 100). The estimated curves are smoothed versions of the PSD. Note that with 500 filters, the windows are very concentrated in the spectral domain and broad in the vertex domain. Thus, we loose the averaging effect of the algorithm resulting in a PSD looking like the Fourier transform of the original signal.

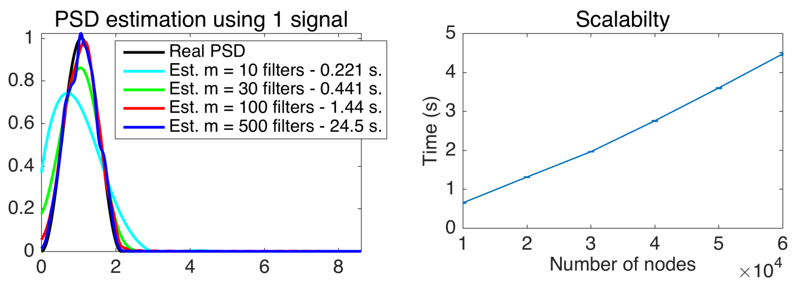


Figure 3: Left: PSD estimation on a graph of 20′000 nodes with $K = 1$ measurements. Our algorithm is able to successively estimate the PSD of a signal. Right: Computation time versus size of the graph. We use $m = 30$ filters. The algorithm scales linearly with the number of edges.

Complexity analysis The approximation scales with the number of edges of the graph: $\mathcal{O}(|\mathcal{E}|)$, (which is proportional to N in many graphs). Precisely, our PSD estimation method necessitates $(K + K_2)M$ filtering operations (with M the number of shifts of g). A filtering operation costs approximatively $O_c|E|$, with O_c the order of the Chebysheff polynomial [20]. The final computational cost of the method is thus $\mathcal{O}\left(O_c(K + K_2)M|\mathcal{E}|\right)$.

Error analysis The difference between the approximation and the exact PSD is caused by three different factors.

1. The inherent bias of the estimator, which is now directly controlled by the parameter σ.

2. We estimate the expected value using K realization of a the signal (often $K = 1$). For large graphs $N >> K$ and a few filters $M << N$, this error is usually low because the variance of $\|g_m(L)\mathbf{x}\|_2^2$ is inversely proportional to bias. The estimation error improves as $\frac{1}{K}$.

3. We use a fast-filtering method based on a polynomial approximation of the filter. For a rough approximation, $\sigma >> \frac{\lambda_{\max}}{N}$, this error is usually negligible. However, in the other cases, this error may become large.

5 Graph Wiener filters and optimization framework

Using stationary signals, we can naturally extend the framework of Wiener filters [4] largely used in signal processing for Mean Square Error (MSE) optimal linear prediction. Since the construction of Wiener filters is very similar for non-graph and graph signals, we present only the latter here. The main difference is that the traditional frequencies are replaced by the graph Laplacian eigenvalues[3] λ_ℓ. The full Wiener recovery process is summarized in Figure 4.

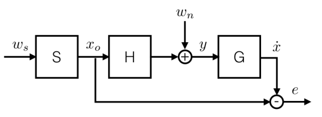


Figure 4: Wiener process model. w_s is a centered random variable with covariance I. w_n is the noise with PSD $n(\lambda)$
, h is the convolution filter and g the Wiener filter. x_o is the original signal, y the measurements, $\dot{x}$ the recovery and e the final error.

The Wiener filter can be used to produce a mean-square error optimal estimate of a stationary signal under a linear but noisy observation model. Let us consider a sample x_o of GWSS process $\mathbf{x}$ with PSD of $s^2(\lambda_\ell)$. The measurement y is given by:

$$y = h(L)x_o + w_n, \tag{10}$$

where h is a graph filter and w_n additive noise of PSD $n(\lambda_\ell)$.

To recover x_o, we extend Wiener filters to the graph case:

$$g(\lambda_\ell) = \frac{h(\lambda_\ell)s^2(\lambda_\ell)}{h^2(\lambda_\ell)s^2(\lambda_\ell) + n(\lambda_\ell)}. \tag{11}$$

The expression above can be derived identically as in the classical case and minimizes the expected quadratic error, which can be written as:

$$e(\lambda_\ell) = \mathbb{E}\left(\hat{x}_o(\lambda_\ell) - \hat{\dot{x}}(\lambda_\ell)\right)^2 = \mathbb{E}\left(\hat{x}(\lambda_\ell) - g(\lambda_\ell)\hat{y}(\lambda_\ell)\right)^2,$$

where $\dot{x} = g(L)y$ is the recovered signal. Theorem 4 proves the optimality of this filter for the graph case.

[3]The graph eigenvalues are equivalent to classical squared frequencies.

Wiener optimization In this contribution, we would like to address a more general problem. Let us suppose that our measurements are generated as:

$$y = Ax_o + w_n, \tag{12}$$

where the original signal x_o has a PSD denoted $s^2(\lambda_\ell)$ and the noise w_n a PSD of $n(\lambda_\ell)$. A is a general linear operator not assumed to be diagonalizable with L. As a result, we cannot build a Wiener filter that constructs a direct estimation of the signal x. If the signal x varies smoothly on the graph, i.e is low frequency based, a classic optimization scheme would be the following:

$$\arg\min_x \|Ax - y\|_2^2 + \gamma x^t Lx. \tag{13}$$

This optimization scheme presents two main disadvantages. Firstly, the parameter γ must be tuned in order to remove the best amount of noise. Secondly, it does not take into account the data structure characterized by the PSD $s^2(\lambda_\ell)$.

Our solution to overcome these issues is to solve the following optimization problem (that we suggestively call Wiener optimization):

$$\dot{x} = \arg\min_x \|Ax - y\|_2^2 + \|w(L)x\|_2^2, \tag{14}$$

where $w(\lambda_\ell)$ is the Fourier penalization weights. These weights are defined as

$$w(\lambda_\ell) = \left| \frac{\sqrt{n(\lambda_\ell)}}{s(\lambda_\ell)} \right| = \frac{1}{\sqrt{SNR(\lambda_\ell)}}.$$

In the noise-less case, one can alternatively solve the following problem

$$\dot{x} = \arg\min_x \|s^{-1}(L)x\|_2^2, \qquad \text{s. t. } Ax = y. \tag{15}$$

Problem (14) generalizes Problem (13) which assumes implicitly a PSD of $\frac{1}{\lambda_\ell}$ and a constant noise level of γ across all frequencies. Note that this framework generalizes two main assumptions done on the data in practice:

1. The signal is smooth on the graph, i.e: the edge derivative has a small ℓ_2-norm. As seen before this is done by setting the PSD as $\frac{1}{\lambda_\ell}$. This case is studied in [15].
2. The signal is band-limited, i.e: is it a linear combination of the k lowest graph Laplacian eigenvectors. This class of signal simply have a null PSD for $\lambda_\ell > \lambda_k$.

3 motivations for the optimization framework The first motivation is intuitive. The weight $w(\lambda_\ell)$ heavily penalizes frequencies associated to low SNR and vice versa.

The second and main motivation is theoretical. If we have a Gaussian Random process with i.i.d Gaussian noise, then Problem (14) is a MAP estimator.

Theorem 3. *If x is a sample of a Gaussian random process $x \sim \mathcal{N}\left(0, s^2(L)\right)$ and the noise is Gaussian i.i.d $w_n \sim \mathcal{N}\left(0, \sigma^2\right)$, then Problem* (14) *is a MAP estimator for $x|y$.*

The proof is given in Appendix B.

Additionally, when A is diagonalizable, Problem (14) can be solved by a single filtering operation.

Theorem 4. *If the operator A is diagonalizable with L, (i.e: $A = a(L) = Ua(\Lambda)U^*$), then problem (14) is optimal with respect of the weighting w in the sense that its solution minimizes the mean square error:*

$$\mathbb{E}\left(\|e\|_2^2\right) = \mathbb{E}\left(\|\dot{x} - x_o\|_2^2\right) = \mathbb{E}\left(\sum_{i=1}^{N} (\dot{x}[i] - x_o[i])^2\right).$$

Additionally, the solution can be computed by the application of the corresponding Wiener filter.

The proof is given in Appendix C.

The last motivation is algorithmic and requires the knowledge of proximal splitting methods [23, 24]. Problem (14) can be solved by a splitting scheme that minimizes iteratively each of the terms. The minimization of the regularizer, i.e the proximal operator of $\|w(L)x\|_2^2$, becomes a Wiener de-noising operation:

$$\begin{aligned} \operatorname{prox}_{\frac{1}{2}\|w(L))x\|_2^2}(y) &= \arg\min_x \|w\mathcal{F}x\|_2^2 + \|x - y\|_2^2 \\ &= g(L)y \end{aligned}$$

with

$$g(\lambda_\ell) = \frac{1}{1 + w^2(\lambda_\ell)} = \frac{s^2(\lambda_\ell)}{s^2(\lambda_\ell) + n(\lambda_\ell)}.$$

Advantage of the Wiener optimization framework over a Gaussian MAP estimator Theorem 3 shows that the optimization framework is equivalent to a Gaussian MAP estimator. In practice, when the data is only close to stationary, the true MAP estimator will perform better than Wiener optimization. So one could ask why we bother defining stationarity on graphs. Firstly, assuming stationarity allows us for a much robust estimate of the covariance matrix. This is shown is in Figure 3, where only one signal is used to estimate the PSD (and thus the covariance matrix). Another example is the USPS experiment presented in the next section. We estimate the PSD by using only 20 digits. The final result is much better than a Gaussian MAP based on the empirical covariance. Secondly, we have a scalable solution for Problem (14) (See Algorithm 1 below). On the contrary the classical Gaussian MAP estimator requires the explicit computation of a large part of the covariance matrix and it's inverse, which are both not scalable operations.

Solving Problem (14) Note that Problem (14) can be solved with a simple gradient descent. However, for a large number of nodes N, the matrix $w(L)$ requires $O(N^3)$ operations to be computed and $O(N^2)$ bits to be stored. This difficulty can be overcome by applying its corresponding filter operator at each iteration. As already mentioned, the cost of the approximation is $O(|E|)$ [20].

When $s(\lambda_\ell) \approx 0$ for some λ_ℓ the operator $w(L)$ becomes badly conditioned. To overcome this issue, Problem (14) can be solved efficiently using a forward-backward splitting scheme [25, 23, 24]. The proximal operator of $\|w(L)x\|_2^2$

has been given above and we use the term $\|Ax - y\|_2^2$ as the differentiable function. Algorithm 1 uses an accelerated forward backward scheme [26] to solve Problem (14) where γ is the step size (we use $\gamma = \frac{1}{2\lambda_{\max}(A)^2}$), ϵ the stopping tolerance, J the maximum number of iterations and δ is a very small number to avoid a possible division by 0.

Algorithm 1 Fast Wiener optimization to solve (14)

```
INPUT: $z_1 = x$, $u_0 = x$, $t_1 = 1$, $\epsilon > 0$, $\gamma \leq \frac{1}{2\lambda_{\max}(A)^2}$
SET: $g(\lambda) = \frac{s^2(\lambda)}{s^2(\lambda)+\gamma n(\lambda)}$                ▷ Wiener filter
for  $j = 1, \dots J$  do
    $v = z_j - \gamma A^*(Az_j - y)$                ▷ Gradient step
    $u_{j+1} = g(L)v$                ▷ Proximal step
    $t_{j+1} = \frac{1+\sqrt{1+4t_j^2}}{2}$                ▷ FISTA scheme
    $z_{j+1} = z_j + \frac{t_j - 1}{t_{j+1}}(u_j - u_{j-1})$                ▷ Update step
    if $\frac{\|z_{j+1}-z_j\|_F^2}{\|z_j\|_F^2+\delta} < \epsilon$ then                ▷ Stopping criterion
        BREAK
    end if
end for
SOLUTION: $z_J$
```

6 Stationarity in real datasets: illustration with USPS

Stationarity may not be an obvious hypothesis for a general dataset, since our intuition does not allow us to easily capture the kind of shift invariance that is really implied. In this section we give additional insights on stationarity from a more experimental point of view. To do so, we will show that the well-known USPS dataset is close to stationary on a nearest neighbor graph. We show similar results with a dataset of faces.

Images can be considered as signals on the 2-dimensional euclidean plane and, naturally, when the signal is sampled, a grid graph is used as a discretization of this manifold. The corresponding eigenbasis is the 2 dimensional DCT[4]. Following Julesz's initial assumption [28] many papers have exploited the fact that natural texture images are stationary 2-dimensional processes, i.e stationary signals on the grid graph [29]. In [30], the authors go one step further and ask the following question: suppose that pixels of images have been permuted, can we recover their relative two-dimensional location? Amazingly, they answer positively adding that only a few thousand images are enough to approximatively recover the relative location of the pixels. The grid graph seems naturally encoded within images.

The observation of [30] motivates the following experiment involving stationarity on graphs. Let us select the USPS data set which contains 9298 digit images of 16×16 pixels. We create 5 classes of data: (a) the circularly shifted

[4]This is a natural extension of [27]

Data \Graph	2-dimensional grid	20 nearest neighbors graph
Shifted all digits	0.86	1
All digits	0.73	0.84
Digit 3	0.53	0.97
Digit 7	0.44	0.97
Digit 9	0.49	0.97

Table 2: $s_r(\Gamma) = \frac{\|\text{diag}(\Gamma)\|_2}{\|\Gamma\|_F}$: stationarity measures for different graphs and different datasets. The nearest neighbors graph adapts to the data. The individual digits are stationary with the nearest neighbor graph.

digits[5], (b) the original digits and (c), (d) and (e) the classes of digit 3, 7 and 9. For those 5 cases, we compute the covariance matrix Σ and its graph PSD,

$$\Gamma = U^* \Sigma U, \tag{16}$$

for 2 different graphs: (a) the grid and (b) the 20 nearest neighbors graph. In this latter case, each node is a pixel and is associated to a feature vector containing the corresponding pixel value of all images. We use the squared euclidean distance between feature vectors to define edge weights. We then compute the stationarity level of each class of data with both graphs using the following measure:

$$s_r(\Gamma) = \left(\frac{\sum_\ell \Gamma_{\ell,\ell}^2}{\sum_{\ell_1} \sum_{\ell_2} \Gamma_{\ell_1,\ell_2}^2} \right)^{\frac{1}{2}} = \frac{\|\text{diag}(\Gamma)\|_2}{\|\Gamma\|_F}. \tag{17}$$

The closer $s_r(\Gamma)$ is to 1, the more diagonal the matrix Γ is and the more stationary the process. Table 2 shows the obtained stationarity measures. The less universal the data, the less stationary it is on the grid. Clearly, specificity inside the data requires a finer structure than a grid. This is confirmed by the behavior of the nearest neighbors graph. When only one digit class is selected the nearest neighbors graph still yields very stationary signals.

Let us focus on the digit 3. For this experiment, we build a 20 nearest neighbors graph with only 50 samples. Figure 5 shows the eigenvectors of the Laplacian and of the covariance matrix. Because of stationarity, they are very similar. Moreover, they have a 3-like shape. Since the data is almost stationary, we can use graph and the PSD to generate samples by filtering Gaussian random noise with the following PSD based kernel: $g(\lambda_\ell) = \sqrt{\Gamma_{\ell,\ell}}$. The resulting digits have a 3-like shape confirming the that the class is stationary on the nearest neighbors graph.

To further illustrate this phenomenon on a different dataset, we use the CMUPIE set of cropped faces. With a nearest neighbor graph we obtained a stationarity level of $s_r = 0.88$. This has already been observed in [31] where the concept of Laplacianfaces is introduced. Finally in [32] the authors succesfully use the graph between features to improve the quality of a low-rank recovery problem. The reason seems to be that the principal components of the data are the lowest eigenvectors of the graph, which is again a stationarity assumption.

[5]We performed all possible shifts. Because of this, the covariance matrix becomes Toeplitz

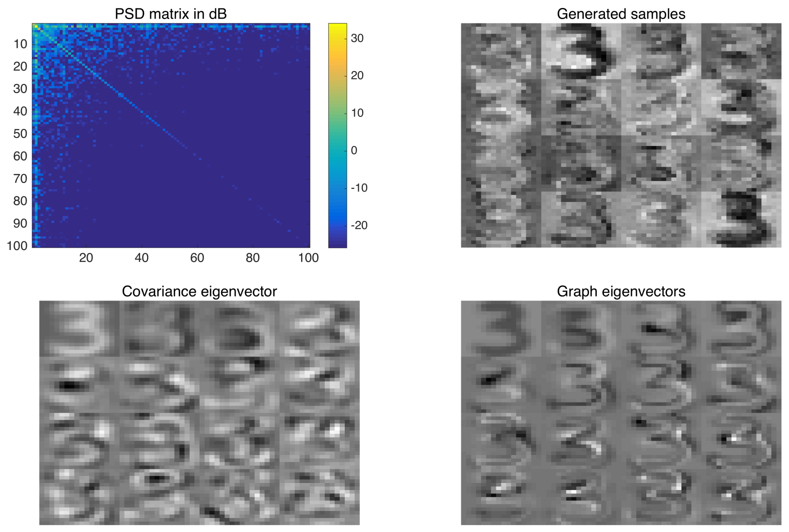


Figure 5: Studying the number 3 of USPS using a 20-neighbors graph. Top left: PSD of the data (Note the diagonal shape of the matrix). We only display the upper left part for better visibility. Top right: generated samples by filtering Gaussian random noise on the graph. Bottom left: Covariance eigenvectors associated with the 16 highest eigenvalues. Bottom right: Laplacian eigenvectors associated to the 16 smallest non-zero eigenvalues. Because of stationarity, Laplacian eigenvectors are similar to the covariance eigenvectors.

To intuitively motivate the effectiveness of nearest neighbors at producing stationary signals, let us define the centering operator $J = I - \mathbf{1}\mathbf{1}^\top/N$. A straightforward calculation shows that the matrix of average squared distances between the centered features is directly proportional to the covariance matrix :

$$\frac{1}{d}\Sigma_{\mathbf{x}} = -\frac{1}{2} J D_{\mathbf{x}} J,$$

where $D_{ij} = \sum_k (x_k[i] - x_k[j])^2$ and $\mathbb{E}D = D_{\mathbf{x}}$. The nearest-neighbors graph can be seen as an approximation of the original distance matrix, which pleads for using it as a good proxy destined at leveraging the spectral content of the covariance. Put differently, when using realizations of the process as features and computing the k-NN graph we are connecting strongly correlated variables via strong edge weights.

7 Experiments

All experiments were performed with the GSPBox [33] and the UNLocBoX [34] two open-source softwares. The code to reproduce all figures of the paper can be downloaded at: `https://lts2.epfl.ch/rrp/stationarity/`. As the stationary signals are random, the reader may obtain slightly different results. However, conclusions shall remain identical. The models used in our comparisons are detailed in the Appendix A for completeness, where we also detail how the tuning of the parameters.

7.1 Synthetic dataset

In order to obtain a first insight into applications using stationarity, we begin with some classical problems solved on a synthetic dataset. Compared to real data, this framework allows us to be sure that the signal is stationary on the graph.

Graph Wiener deconvolution We start with a de-convolution example on a random geometric graph. This can model an array of sensors distributed in space or simply a mesh. The signal is chosen with a low frequency band-limited PSD. To produce the measurements, the signal is convolved with the heat kernel $e^{-\tau x}$. Additionally, we add some Gaussian noise. The heat kernel is chosen because it simulates a diffusion process. Using de-convolution we aim at recovering the original signal before diffusion. For this experiment, we put ourselves in an ideal case and suppose that both the PSD of the input signal and the noise level are known.

Figure 6 presents the results. We observe that Wiener filtering is able to de-convolve the measurements. The second plot shows the reconstruction errors for three different methods: Tikonov presented in problem (19), TV in (21) and Wiener filtering in (11). Wiener filtering performs clearly much better than the other methods because it has a much better prior assumption.

Graph Wiener in-painting In our second example, we use Wiener optimization to solve an in-painting problem. This time, we suppose that the PSD of the input signal is unknown and we estimate it using 50 signals. Figure 7 presents quantitative results for the in-painting. Again, we compare three different optimization methods: Tikonov (18), TV (21) and Wiener (14). Additionally we compute the classical MAP estimator based on the empirical covariance matrix (see [35] 2.23). Wiener optimization performs clearly much better than the other methods because it has a much better prior assumption. Even with 50 measurements, the MAP estimator performs poorly compared to graphs method. The reason is that the graph contains a lot of the covariance information. Note that the PSD estimated with only one measurement is sufficient to outperform Tikonov and TV.

7.2 USPS dataset

We perform the same kind of in-painting/de-noising experiment with the USPS dataset. For our experiments, we consider every digit as an independent realization of a GWSS process. As sole pre-processing, we remove the mean (over pixels and digits)[6]. We create the graph[7] and estimate the PSD using only the

[6]It is also possible to remove the mean of each pixel separately. It might increase the stationarity level of the data. In this contribution, we choose not to perform this pre-processing as we consider the raw data stationary.

[7]The graph is created using patches of pixels of size 5×5. The pixels' patches help because we have only a few digits available. When the size of the data increases, a nearest neighbor graph performs even better.

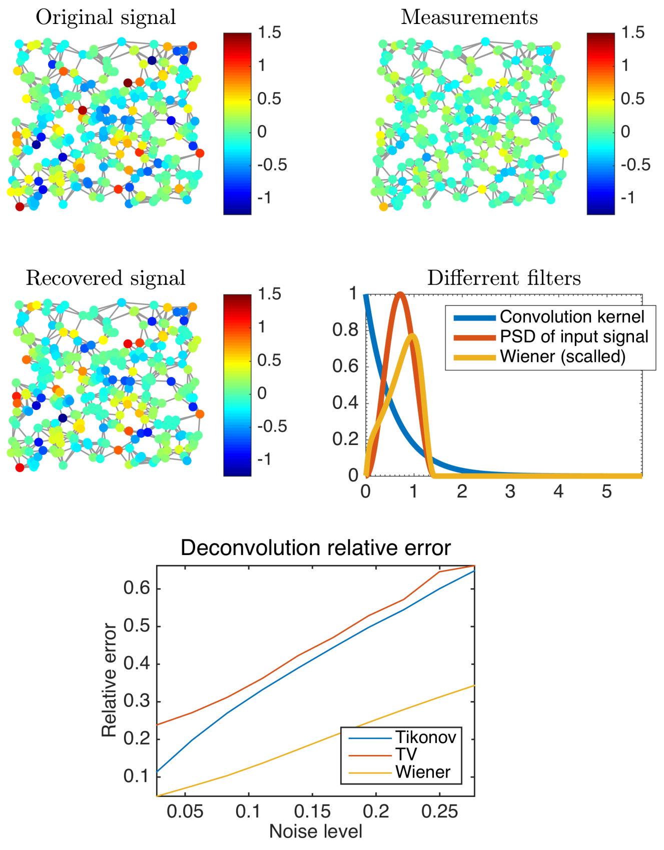


Figure 6: Graph de-convolution on a geometric random graph. The convolution kernel is $e^{-\frac{10x}{\lambda_{\max}}}$. Top: Signal and filters for a noise level of 0.16. Bottom: evolution of the error with respect of the noise.

first 20 digits and we use 500 of the remaining ones to test our algorithm. We use a mask covering 50 per cent of the pixel and various amount of noise. We then average the result over 500 experiments (corresponding to the 500 digits) to obtain the curves displayed in Figure 8[8]. For this experiment, we also compare to traditional TV de-noising [36] and Tikonov de-noising. The optimization problems used are similar to (18). Additionally we compute the classical MAP estimator based on the empirical covariance matrix for the solution see ([35] 2.23). The results presented in Figure 8 show that graph optimization is outperforming classical techniques meaning that the grid is not the optimal graph for the USPS dataset. Wiener once again outperforms the other graph-based models. Moreover, this experiment shows that our PSD estimation is robust when the number of signals is small. In other words, using the graph allows

[8]All parameters have been tuned optimally in a probabilistic way. This is possible since the noise is added artificially. The models presented in Appendix A have only one parameter to be tuned: ϵ which is set to $\epsilon = \sigma\sqrt{\#y}$, where σ is the variance of the noise and $\#y$ the number of elements of the vector y. In order to be fair with the MAP estimator, we construct the graph with the only 20 digits used in the PSD estimation.

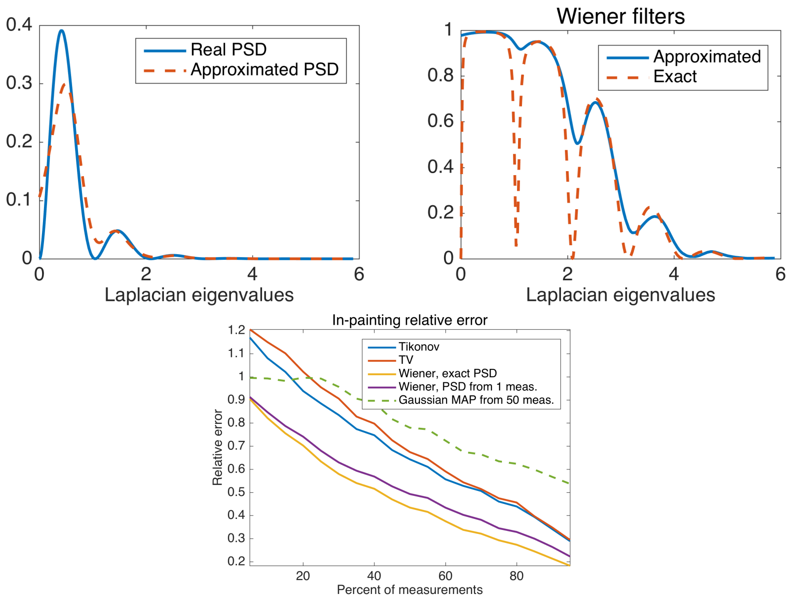


Figure 7: Wiener in-painting on a geometric graph of 400 nodes. Top: true VS approximated PSD and resulting Wiener filters. Bottom: in-painting relative error with respect to number of measurements.

us for a much better covariance estimation than a simple empirical average. When the number of measurements increases, the MAP estimator improves in performance and eventually outperforms Wiener because the data is close to stationary on the graph.

7.3 ORL dataset

For this last experiment, we use the ORL faces dataset. We have a good indication that this dataset is close to stationary since CMUPIE (a smaller faces dataset) is also close to stationary. Each image has $112 \times 92 = 10304$ pixels making it complicated to estimate the covariance matrix and to use a Gaussian MAP estimator. Wiener optimization on the other hand does not necessitate an explicit computation of the covariance matrix. Instead, we estimate the PSD using the algorithm presented in Section 4. A detailed experiment is performed in Figure 9. After adding Gaussian noise to the image, we remove randomly 25% of the pixels. We consider the obtained image as the measurement and we reconstruct the original image using TV, Tikonov and Wiener priors. In Figure 10, we display the reconstruction results for various noise levels. We create the graph with 300 faces and estimate the PSD with 100 faces. We test the different algorithms on the 100 remaining faces.

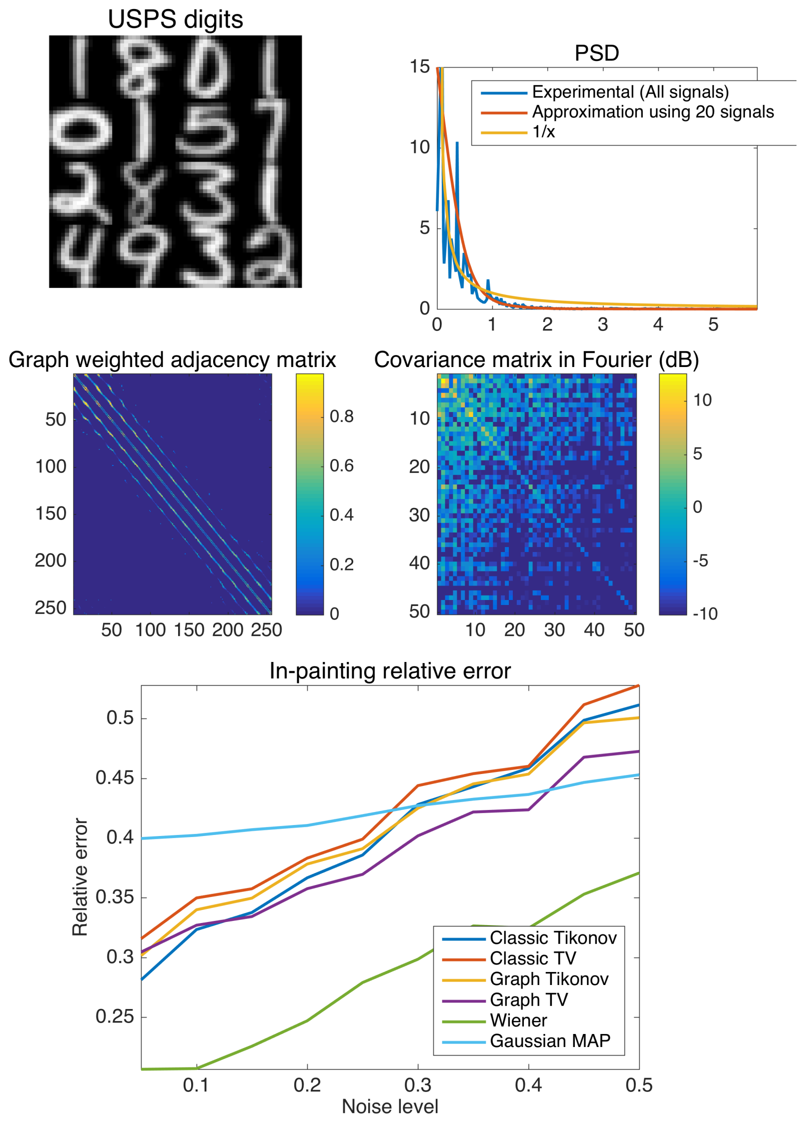


Figure 8: Top right: Some digits of the USPS dataset. Top left: Different PSDs. Compared to $\frac{1}{x}$, the approximation is a smoothed version of the experimental PSD. Middle left: Weights matrix of the 10 nearest neighbors (patch) graph (The diagonal shape indicates the grid base topology of the graph). Middle right: spectral covariance matrix for the first 50 graph frequencies. Since we only use 20 digits for the graph construction, the stationarity level is low. Nevertheless, Wiener optimization outperforms other methods. Bottom: Recovery errors for different noise levels. Methods using the graph perform better. Even if the data is not stationary on the graph, the stationarity assumption helps a lot in the recovery.

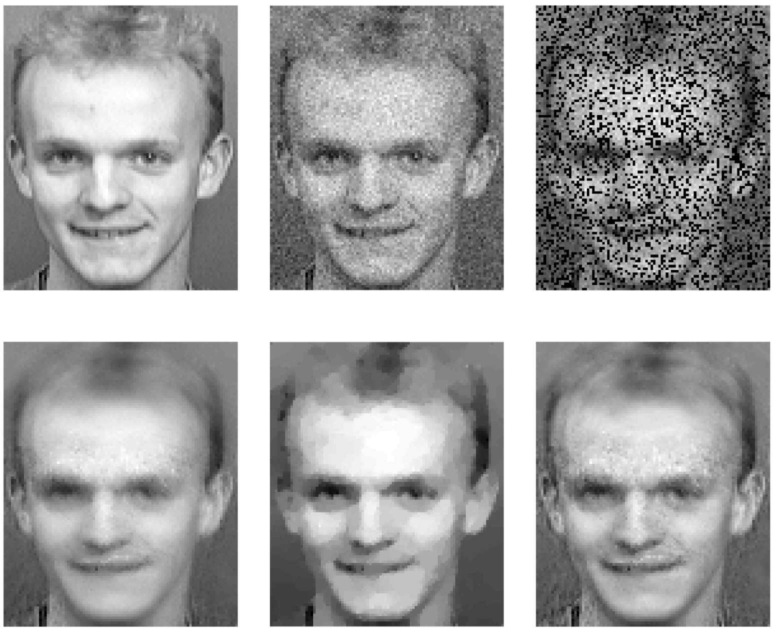


Figure 9: ORL dataset, single in-painting experiment. Top left: Original image. Top center: Noisy image. Top right: Measurements 75% of the noisy image. Bottom left: Reconstruction using Tikonov prior (relative error 21.3%). Bottom center: Reconstruction using classic TV prior (relative error 19.7%). Bottom right: Reconstruction using Wiener optimization (relative error 18.2%).

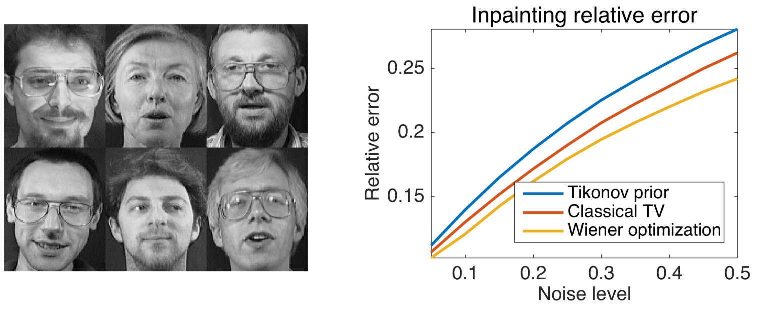


Figure 10: Inpainting experiment on ORL dataset. Left: some images of the dataset. Right: reconstruction error.

8 Conclusion

In this contribution, we have extended the common concept of stationarity to graph signals. Using this statistical model, we proposed a new regularization framework that leverages the stationarity hypothesis by using the Power Spectral Density (PSD) of the signal. Since the PSD can be efficiently estimated, even for large graphs, the proposed Wiener regularization framework offers a compelling way to solve traditional problems such as de-noising, regression or semi-supervised learning. We believe that stationarity is a natural hypothesis for many signals on graphs and showed experimentally that it is deeply connected with the popular nearest neighbor graph construction. As future work, it would be very interesting to clarify this connection and explore if stationarity could be used to infer the graph structure from training signals, in the spirit of [37].

Acknowledgment

This work has been supported by the Swiss National Science Foundation research project *Towards Signal Processing on Graphs*, grant number: 2000_21/154350/1.

A Convex models

Convex optimization has recently become a standard tool for problems such as de-noising, de-convolution or in-painting. Graph priors have been used in this field for more than a decade [8, 9, 10]. The general assumption is that the signal varies smoothly along the edges, which is equivalent to saying that the signal is low-frequency-based. Using this assumption, one way to express mathematically an in-painting problem is the following:

$$\dot{x} = \arg\min_x x^T L x \qquad \text{s.t.} \qquad \|Mx - y\|_2 \leq \epsilon \tag{18}$$

where M is a masking operator and ϵ a constant computed thanks to the noise level. We could also rewrite the objective function as $x^T Lx + \gamma\|Mx - y\|_2^2$, but this implies a greedy search of the regularization parameter γ even when the level of noise is known. For our simulations, we use Gaussian i.i.d. noise of standard deviation n. It allows us to optimally set the regularization parameter $\epsilon = n\sqrt{\#y}$, where $\#y$ is the number of elements of the measurement vector.

Graph de-convolution can also be addressed with the same prior assumption leading to

$$\dot{x} = \arg\min_x x^T L x \qquad \text{s.t.} \qquad \|h(L)x - y\|_2 \leq \epsilon \tag{19}$$

where h is the convolution kernel. To be as generic as possible, we combine problems (18) and (19) together leading to a model capable of performing de-convolution, in-painting and de-noising at the same time:

$$\dot{x} = \arg\min_x x^T L x \qquad \text{s.t.} \qquad \|Mh(L)x - y\|_2 \leq \epsilon. \tag{20}$$

When the signal is piecewise smooth on the graph, another regularization term can be used instead of $x^T Lx = \|\nabla_G x\|_2^2$, which is the ℓ_2-norm of the gradient on the graph[9]. Using the ℓ_1-norm of the gradient favours a small number of major changes in signal and thus is better for piecewise smooth signals. The resulting model is:

$$\dot{x} = \arg\min_x \|\nabla_G x\|_1 \qquad \text{s.t.} \qquad \|Mh(L)x - y\|_2 \leq \epsilon \tag{21}$$

[9]The gradient on the graph is defined as $\nabla_G x(i,j) = \frac{1}{2}\sqrt{W(i,j)}\,(x(i) - x(j))$

In order to solve these problems, we use a subset of convex optimization tools called proximal splitting methods. Since we are not going to summarise them here, we encourage a novice reader to consult [23, 24] and the references therein for an introduction to the field.

B Proof of Theorem 3

Proof. The proof is a classic developement used in bayesian machine learning. By assumption x is a sample of a Gaussian random process $x \sim \mathcal{N}\left(0, s^2(L)\right)$. The measurements are given by

$$y = Ax + w_n,$$

where $w_n \sim \mathcal{N}\left(0, \sigma^2\right)$ and thus have the following first and second moments: $y|x \sim \mathcal{N}\left(Ax, \sigma^2 I\right)\big)$. We can write the probabilities of x and $y|x$ as:

$$\mathbb{P}(x) = \frac{1}{Z_{Ax}} e^{-\|s^{-1}(L)x\|_2^2},$$
$$\mathbb{P}(y|x) = \frac{1}{Z_{sx}} e^{-\sigma^2 \|(Ax-y)\|_2^2}.$$

Using Bayes law we find

$$\mathbb{P}(x|y) = \frac{\mathbb{P}(y|x)\mathbb{P}(x)}{\mathbb{P}(y)}.$$

The MAP estimator is

$$\begin{aligned}
\dot{x} &= \arg\max_x \mathbb{P}(x|y) \\
&= \arg\max_x \log\left(\mathbb{P}(x|y)\right) \\
&= \arg\min_x - \log\left(\mathbb{P}(y|x)\right) - \log\left(\mathbb{P}(x)\right) + \log\left(\mathbb{P}(y)\right) \\
&= \arg\min_x \|s^{-1}(L)x\|_2^2 + \sigma^{-2}\|(Ax-y)\|_2^2 \\
&= \arg\min_x \|w(L)x\|_2^2 + \|(Ax-y)\|_2^2,
\end{aligned}$$

where $w(L) = \sigma s^{-1}(L)$. □

C Proof of Theorem 4

The following is a generalization of the classical proof.

Proof. Because, by hypothesis $A = a(L) = Ua(\Lambda)U^*$, we can rewrite the optimization problem (14) in the graph Fourier domain using the Parseval identity $\|x\|_2 = \|Ux\|_2 = \|\hat{x}\|_2$:

$$\dot{\hat{x}} = \arg\min_{\hat{x}} \|w(\Lambda)\hat{x}\|_2^2 + \|a(\Lambda)\hat{x} - \hat{y}\|_2^2.$$

Since the matrix Λ is diagonal, the solution of this problem satisfies for all graph eigenvalue λ_ℓ

$$w^2(\lambda_\ell)\hat{x}[\ell] + a^2(\lambda_\ell)\hat{x}[\ell] - a(\lambda_\ell)\hat{y}[\ell] = 0. \tag{22}$$

For simplicity, we drop the notation (λ_ℓ) and $[\ell]$. The previous equation is transformed in

$$\dot{\hat{x}} = \frac{a}{w^2 + a^2}\hat{y}.$$

As a next step, we use the fact that $\hat{y} = a\hat{x}_o + \hat{w}_n$ to find:

$$\dot{\hat{x}} = \frac{a^2\hat{x}_o + a\hat{w}_n}{w^2 + a^2}.$$

The error performed by the algorithm becomes

$$\hat{e} = \dot{\hat{x}} - \hat{x}_o = \frac{-w^2\hat{x}_o}{w^2 + a^2} + \frac{a\hat{n}}{w^2 + a^2}.$$

The expectation of the error can thus be computed:

$$\begin{aligned} \mathbb{E}\left(\hat{e}^2\right) &= \frac{w^4\mathbb{E}\left(\hat{x}_o^2\right)}{\left(w^2 + a^2\right)^2} + \frac{a^2\mathbb{E}\left(\hat{w}_n^2\right)}{\left(w^2 + a^2\right)^2} - \frac{aw^2\mathbb{E}\left(\hat{x}_o\hat{w}_n\right)}{\left(w^2 + a^2\right)^2} \\ &= \frac{w^4 s^2 + a^2 n}{\left(w^2 + a^2\right)^2}, \end{aligned}$$

with s^2 the PSD of x_o and n the PSD of the noise w_n. Note that $\mathbb{E}\left(\hat{x}_o\hat{w}_n\right) = 0$ because x and w_n are uncorrelated. Let us now substitute w^2 by z and minimize the expected error (for each λ_ℓ) with respect to z:

$$\begin{aligned} \frac{\partial}{\partial z}\mathbb{E}\left(\hat{e}^2\right) &= \frac{\partial}{\partial z}\frac{zs^2 + a^2 n}{\left(z + a^2\right)^2} \\ &= \frac{2zs^2\left(z + a^2\right) - 2\left(z^2 s^2 + a^2 n\right)}{\left(z + a^2\right)^3} = 0. \end{aligned}$$

From the numerator, we get:

$$2zs^2a^2 - 2a^2n = 0$$

The three possible solution for z are $z_1 = \frac{n}{s^2}$, $z_2 = \infty$ and $z_3 = -\infty$. z_3 is not possible because z is required to be positive. z_2 leads to $\dot{x} = 0$ which is optimal only if $s^2 = 0$. The optimal solution is therefore $z(\lambda_\ell) = \frac{n(\lambda_\ell)}{s^2(\lambda_\ell)}$, resulting in

$$w(\lambda_\ell) = \sqrt{\frac{n(\lambda_\ell)}{s^2(\lambda_\ell)}}.$$

This finishes the first part of the proof. To show that the solution to (14) is a Wiener filtering operation, we replace $w^2(\lambda_\ell)$ by $\frac{n(\lambda_\ell)}{s^2(\lambda_\ell)}$ in (22) and find

$$\hat{x}(\lambda_\ell) = \frac{s^2(\lambda_\ell)a(\lambda_\ell)\hat{y}(\lambda_\ell)}{a^2(\lambda_\ell)s^2(\lambda_\ell) + n(\lambda_\ell)},$$

which is the Wiener filter associated to the convolution $a(L) = A$. $\square$

References

[1] C. Williams, “Prediction with Gaussian processes: From linear regression to linear prediction and beyond,” *Learning in graphical models*, 1998.

[2] D. I. Shuman, B. Ricaud, and P. Vandergheynst, “Vertex-frequency analysis on graphs,” *arXiv preprint arXiv:1307.5708*, 2013.

[3] N. Wiener, “Generalized harmonic analysis,” *Acta mathematica*, vol. 55, no. 1, pp. 117–258, 1930.

[4] ——, *Extrapolation, interpolation, and smoothing of stationary time series.* MIT press Cambridge, MA, 1949, vol. 2.

[5] P. Welch, “The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms,” *IEEE Transactions on audio and electroacoustics*, pp. 70–73, 1967.

[6] M. S. Bartlett, “Periodogram analysis and continuous spectra,” *Biometrika*, pp. 1–16, 1950.

[7] 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,” *Signal Processing Magazine, IEEE*, vol. 30, no. 3, pp. 83–98, 2013.

[8] A. J. Smola and R. Kondor, “Kernels and regularization on graphs,” in *Learning theory and kernel machines.* Springer, 2003, pp. 144–158.

[9] D. Zhou and B. Schölkopf, “A regularization framework for learning from graph data,” 2004.

[10] G. Peyré, S. Bougleux, and L. Cohen, “Non-local regularization of inverse problems,” in *Computer Vision–ECCV 2008.* Springer, 2008, pp. 57–68.

[11] A. J. Smola and R. Kondor, “Kernels and Regularization on Graphs,” in *Proc. Ann. Conf. Comp. Learn. Theory*, B. Schölkopf and M. Warmuth, Eds. Springer, 2003, pp. 144–158.

[12] D. K. Hammond, P. Vandergheynst, and R. Gribonval, “Wavelets on graphs via spectral graph theory,” *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, 2011.

[13] A. Sandryhaila and J. M. Moura, “Discrete signal processing on graphs,” *IEEE transactions on signal processing*, vol. 61, pp. 1644–1656, 2013.

[14] B. Girault, P. Gonçalves, and É. Fleury, “Translation and stationarity for graph signals,” Ph.D. dissertation, École Normale Supérieure de Lyon; Inria Rhône-Alpes, 2015.

[15] B. Girault, P. Goncalves, E. Fleury, and A. S. Mor, “Semi-supervised learning for graph to signal mapping: A graph signal wiener filter interpretation,” in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on.* IEEE, 2014, pp. 1115–1119.

[16] A. Gadde and A. Ortega, "A probabilistic interpretation of sampling theory of graph signals," *arXiv preprint arXiv:1503.06629*, 2015.

[17] C. Zhang, D. Florêncio, and P. A. Chou, "Graph signal processing–a probabilistic framework," 2015.

[18] F. R. Chung, *Spectral graph theory.* AMS Bookstore, 1997, vol. 92.

[19] F. Chung, "Laplacians and the cheeger inequality for directed graphs," *Annals of Combinatorics*, vol. 9, no. 1, pp. 1–19, 2005.

[20] A. Susnjara, N. Perraudin, D. Kressner, and P. Vandergheynst, "Accelerated filtering on graphs using lanczos method," *arXiv preprint arXiv:1509.04537*, 2015.

[21] N. Perraudin, B. Ricaud, D. Shuman, and P. Vandergheynst, "Global and local uncertainty principles for signals on graphs," *arXiv preprint arXiv:1603.03030*, 2016.

[22] A. Papoulis and S. U. Pillai, *Probability, random variables, and stochastic processes.* Tata McGraw-Hill Education, 2002.

[23] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-point algorithms for inverse problems in science and engineering.* Springer, 2011, pp. 185–212.

[24] N. Komodakis and J.-C. Pesquet, "Playing with duality: An overview of recent primal-dual approaches for solving large-scale optimization problems," *arXiv preprint arXiv:1406.5429*, 2014.

[25] P. L. Combettes and V. R. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling & Simulation*, vol. 4, no. 4, pp. 1168–1200, 2005.

[26] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[27] G. Strang, "The discrete cosine transform," *SIAM review*, vol. 41, no. 1, pp. 135–147, 1999.

[28] B. Julesz, "Visual Pattern Discrimination," *Information Theory, IRE Transactions on*, vol. 8, no. 2, pp. 84–92, 1962.

[29] R. C. Dubes and A. K. Jain, "Random field models in image analysis," *Journal of applied statistics*, 1989.

[30] N. L. Roux, Y. Bengio, P. Lamblin, M. Joliveau, and B. Kégl, "Learning the 2-d topology of images," in *Advances in Neural Information Processing Systems*, 2008, pp. 841–848.

[31] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang, "Face recognition using laplacianfaces," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 3, pp. 328–340, 2005.

[32] N. Shahid, N. Perraudin, V. Kalofolias, and P. Vandergheynst, "Fast robust pca on graphs," *arXiv preprint arXiv:1507.08173*, 2015.

[33] N. Perraudin, J. Paratte, D. Shuman, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, "GSPBOX: A toolbox for signal processing on graphs," *ArXiv e-prints*, Aug. 2014.

[34] N. Perraudin, D. Shuman, G. Puy, and P. Vandergheynst, "UNLocBoX A matlab convex optimization toolbox using proximal splitting methods," *ArXiv e-prints*, Feb. 2014.

[35] C. E. Rasmussen, "Gaussian processes for machine learning," 2006.

[36] A. Chambolle, "An algorithm for total variation minimization and applications," *Journal of Mathematical imaging and vision*, vol. 20, no. 1-2, pp. 89–97, 2004.

[37] V. Kalofolias, "How to learn a graph from smooth signals," *arXiv preprint arXiv:1601.02513*, 2016.