Stationary signal processing on graphs

Nathanaël Perraudin

École Polytechnique Fédérale de Lausanne (EPFL)
School of Engineering & School of Computer and Communication Sciences

Joint work with Pierre Vandergheynst

July 27, 2016

State of the art

B. Girault, P. Gonçalves, and É. Fleury, **August 2015**
Stationary graph signals using an isometric graph translation.
(EUSIPCO), 2015 23rd European (pp. 1516-1520). IEEE.

N. Perraudin and P. Vandergheynst, January 2016
Stationary signal processing on graphs.
preprint arXiv:1601.02522.

A. G. Marques, S. Segarra, G. Leus and A. Ribeiro, **March 2016**
Stationary Graph Processes and Spectral Estimation.
preprint arXiv:1603.04667.

Recent research topic

Contributions

1. **Definition**: Laplacian & Covariance: same eigenvectors

Contributions

1. **Definition**: Laplacian & Covariance: same eigenvectors
2. Power Spectrum Density (PSD): robust, scalable **estimator**

Contributions

1. **Definition**: Laplacian & Covariance: same eigenvectors
2. Power Spectrum Density (PSD): robust, scalable **estimator**
3. New **optimization framework**: provably optimal!

Contributions

1. **Definition**: Laplacian & Covariance: same eigenvectors
2. Power Spectrum Density (PSD): robust, scalable **estimator**
3. New **optimization framework**: provably optimal!
4. Applied to **real data**: it works!

Motivation

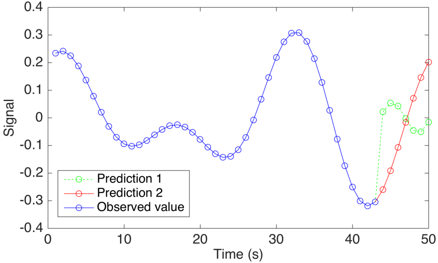


Figure: Signal prediction. The red curve is more likely to occur than the green curve because it respects the frequency statistics of the blue curve.

Motivation

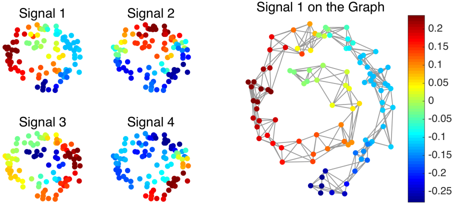


Figure: Example of a stationary stochastic signal on a graph.

Motivation

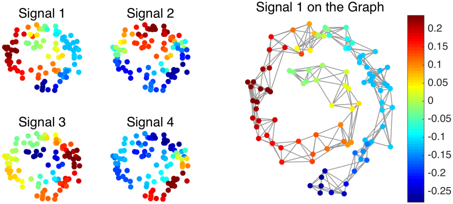


Figure: Example of a stationary stochastic signal on a graph.

Why stationarity?

- Modeling graph processes
- **Data adapted** optimization priors
- Robust covariance estimation from **few samples**

A reminder about graphs

- Weighted undirected graph: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, W\}$

A reminder about graphs

- Weighted undirected graph: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, W\}$
- Laplacian $L = D - W$

A reminder about graphs

- Weighted undirected graph: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, W\}$
- Laplacian $L = D - W$
- **Extendable**
    1. Fourier based on A
    2. normalized Laplacian
    3. directed graphs
- U is the Fourier basis: $L = U \Lambda U^*$

A reminder about graphs

- Weighted undirected graph: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, W\}$
- Laplacian $L = D - W$
- **Extendable**
  1. Fourier based on A
  2. normalized Laplacian
  3. directed graphs
- U is the Fourier basis: $L = U\Lambda U^*$
- We use: $g(L) = Ug(\Lambda)U^*$

Stationarity for temporal signals

Definition (Time Wide-Sense Stationarity)

A signal is Time Wide-Sense Stationary (WSS) if

Stationarity for temporal signals

Definition (Time Wide-Sense Stationarity)

A signal is Time Wide-Sense Stationary (WSS) if

1. $m_{\mathsf{x}}(t) = \mathbb{E}\{\mathsf{x}(t)\} = c \in \mathbb{R}$,

Stationarity for temporal signals

Definition (Time Wide-Sense Stationarity)

A signal is Time Wide-Sense Stationary (WSS) if

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

where γ_{x} is the autocorrelation.

Stationarity for temporal signals

Definition (Time Wide-Sense Stationarity)

A signal is Time Wide-Sense Stationary (WSS) if

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

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

The Power Spectral Density (PSD) is the Fourier transform of the auto-correlation $\gamma_{\mathbf{x}}$:

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

The localization operator

For a continuous **kernel** $\mathbb{R}^+ \to \mathbb{R}$, the localization operator is defined as:

$$\mathcal{T}_i g[n] = \sum_{\ell=0}^{N-1} g(\lambda_\ell) u_\ell^*[i] u_\ell[n] = (g(L))_{in} . \tag{2}$$

The localization operator

For a continuous **kernel** $\mathbb{R}^+ \to \mathbb{R}$, the localization operator is defined as:

$$\mathcal{T}_i g[n] = \sum_{\ell=0}^{N-1} g(\lambda_\ell) u_\ell^*[i] u_\ell[n] = (g(L))_{in}. \tag{2}$$

- Replace the translation operator for graph

The localization operator

For a continuous **kernel** $\mathbb{R}^+ \to \mathbb{R}$, the localization operator is defined as:

$$\mathcal{T}_i g[n] = \sum_{\ell=0}^{N-1} g(\lambda_\ell) u_\ell^*[i] u_\ell[n] = (g(L))_{in} \,. \quad (2)$$

- Replace the translation operator for graph
- When g is smooth, then $\mathcal{T}_i g$ is localized arround i

The localization operator

For a continuous **kernel** $\mathbb{R}^+ \rightarrow \mathbb{R}$, the localization operator is defined as:

$$\mathcal{T}_i g[n] = \sum_{\ell=0}^{N-1} g(\lambda_\ell) u_\ell^*[i] u_\ell[n] = (g(L))_{in} \,. \tag{2}$$

- Replace the translation operator for graph
- When g is smooth, then $\mathcal{T}_i g$ is localized arround i
- Does not preserve the norm

The localization operator

For a continuous **kernel** $\mathbb{R}^+ \to \mathbb{R}$, the localization operator is defined as:

$$\mathcal{T}_i g[n] = \sum_{\ell=0}^{N-1} g(\lambda_\ell) u_\ell^*[i] u_\ell[n] = (g(L))_{in} . \tag{2}$$

- Replace the translation operator for graph
- When g is smooth, then $\mathcal{T}_i g$ is localized arround i
- Does not preserve the norm
- For a ring graph, we recover the translation (of the inverse Fourier transform of the kernel)

The localization operator - example

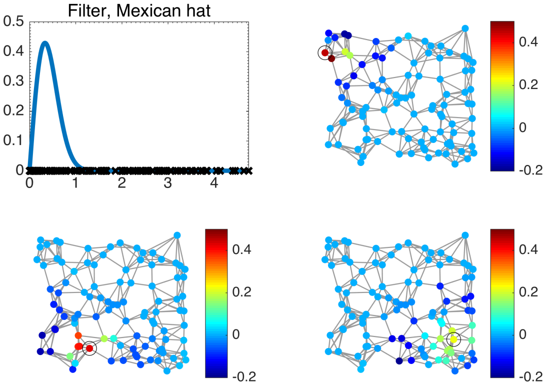


Figure: Top left: Mexican hat filter in the spectral domain $g(x) = \frac{5x}{\lambda_{\max}} \exp\left(-\frac{25x^2}{\lambda_{\max}^2}\right)$. The filter is localized around three different vertices (highlighted by a black circle).

Overview

Stationarity for graphs

Definition

A stochastic process $\mathbf{x}$ is Graph Wide-Sense **Stationary** $\iff$:

Stationarity for graphs

Definition

A stochastic process $\mathbf{x}$ is Graph Wide-Sense **Stationary** $\iff$:

1. $\mathbb{E}\{\mathbf{x}[i]\} = c \in \mathbb{R}$

Stationarity for graphs

Definition

A stochastic process $\mathbf{x}$ is Graph Wide-Sense **Stationary** $\iff$:

1. $\mathbb{E}\left\{\mathbf{x}[i]\right\} = c \in \mathbb{R}$
2. its covariance matrix $\Sigma_{\mathbf{x}}[i,j] = \mathbb{E}\left\{\tilde{\mathbf{x}}[i]\tilde{\mathbf{x}}[j])\right\}$ is jointly diagonalizable with the Laplacian ($\tilde{\mathbf{x}} = \mathbf{x} - \mathbb{E}\{\mathbf{x}\}$)

Stationarity for graphs

Definition

A stochastic process $\mathbf{x}$ is Graph Wide-Sense **Stationary** $\iff$:

1. $\mathbb{E}\left\{\mathbf{x}[i]\right\} = c \in \mathbb{R}$
2. its covariance matrix $\Sigma_{\mathbf{x}}[i,j] = \mathbb{E}\left\{\tilde{\mathbf{x}}[i]\tilde{\mathbf{x}}[j])\right\}$ is jointly diagonalizable with the Laplacian ($\tilde{\mathbf{x}} = \mathbf{x} - \mathbb{E}\{\mathbf{x}\}$)

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

where $\gamma_{\mathbf{x}}$ is the PSD

Examples

- White gaussian noise: graph stationary with a PSD σ^2

Examples

- White gaussian noise: graph stationary with a PSD σ^2
- To generate a stationary process with PSD s^2: just filter white Guassian noise with s

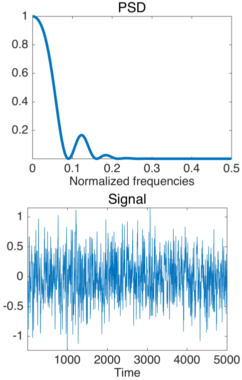
	Classical	Graph
Stationary with respect to	Translation	The localization operator
First Moment	$\mathbb{E}(\mathrm{x}[i]) = m_{\mathrm{x}} = c \in \mathbb{R}$	$\mathbb{E}(\mathrm{x}[i]) = m_{\mathrm{x}} = c \in \mathbb{R}$
Second Moment $\tilde{\mathrm{x}} = \mathrm{x} - m_{\mathrm{x}}$	$\Sigma_{\mathrm{x}}[i,n]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

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

Overview

PSD estimator from a single realization

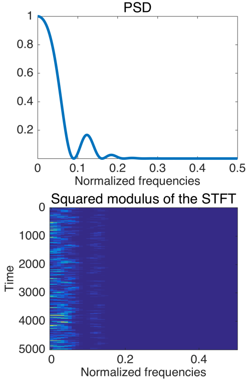
Bartlett method:



PSD estimator from a single realization

Bartlett method:

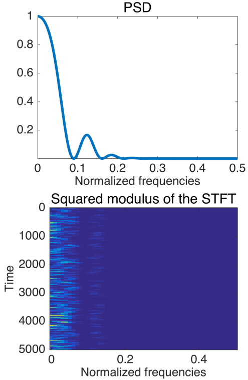
- **Compute the Short Time Fourier Transform**



PSD estimator from a single realization

Bartlett method:

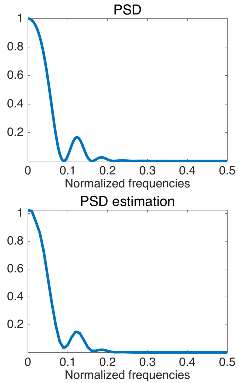
- Compute the Short Time Fourier Transform
- **Take the amplitude squared**



PSD estimator from a single realization

Bartlett method:

- Compute the Short Time Fourier Transform
- Take the amplitude squared
- **Average over time** (with normalization and interpolation if necessary)



Scalable robust PSD estimator

PSD Estimation for graph stationnary process

- Same method using **only graph filtering** operations

Scalable robust PSD estimator

PSD Estimation for graph stationnary process

- Same method using **only graph filtering** operations
- Special normalization: irregular eigenvalues positions

Scalable robust PSD estimator

PSD Estimation for graph stationnary process

- Same method using **only graph filtering** operations
- Special normalization: irregular eigenvalues positions
- **Scale** with the number of edges!

Scalable robust PSD estimator

PSD Estimation for graph stationnary process

- Same method using **only graph filtering** operations
- Special normalization: irregular eigenvalues positions
- **Scale** with the number of edges!

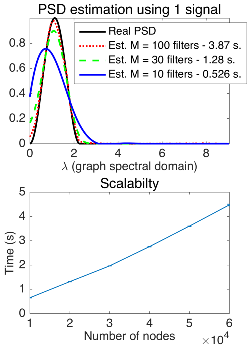
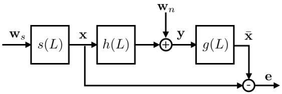


Figure: Left: PSD estimation on a graph of $20'000$ nodes with $K = 1$ measurements.

Overview

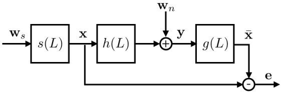
Wiener filters

- x_o: stationary with PSD $s^2(\lambda_\ell)$
- h: graph filter
- y: measurements
- w_n: noise of PSD $n(\lambda_\ell)$



Wiener filters

- x_o: stationary with PSD $s^2(\lambda_\ell)$
- h: graph filter
- y: measurements
- w_n: noise of PSD $n(\lambda_\ell)$



To recover x_o from noisy y, use the Wiener filter:

$$g(\lambda_\ell) = \frac{h(\lambda_\ell)s^2(\lambda_\ell)}{h^2(\lambda_\ell)s^2(\lambda_\ell) + n(\lambda_\ell)}. \tag{4}$$

Optimization framework

- Measurements: $y = Ax_o + w_n$
- x_o: stationary with PSD $s^2(\lambda_\ell)$
- A: linear operator
- w_n: noise of PSD $n(\lambda_\ell)$

Classical "Tikhonov" approach

Wiener optimization

Optimization framework

- Measurements: $y = Ax_o + w_n$
- x_o: stationary with PSD $s^2(\lambda_\ell)$
- A: linear operator
- w_n: noise of PSD $n(\lambda_\ell)$

Classical "Tikhonov" approach

$$\arg\min_x \|Ax - y\|_2^2 + \gamma x^t Lx. \tag{5}$$

Wiener optimization

Optimization framework

- Measurements: $y = Ax_o + w_n$
- x_o: stationary with PSD $s^2(\lambda_\ell)$
- A: linear operator
- w_n: noise of PSD $n(\lambda_\ell)$

Classical "Tikhonov" approach

$$\arg\min_x \|Ax - y\|_2^2 + \gamma x^t Lx. \tag{5}$$

Wiener optimization

$$\dot{x} = \arg\min_x \|Ax - y\|_2^2 + \|w(L)x\|_2^2, \tag{6}$$

where $w(\lambda_\ell)$ are the Fourier penalization weights.

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

Wiener optimization - theorems

Theorem

*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 (6) is a **MAP estimator** for $x|y$.*

Wiener optimization - theorems

Theorem

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 (6) is a* ***MAP estimator*** *for $x|y$.*

Theorem

If the operator A ***is diagonalizable with*** *L, (i.e: $A = a(L) = Ua(\Lambda)U^*$), then problem (6) is optimal with respect of the weighting w in the sense that its solution* ***minimizes the mean square error****:*

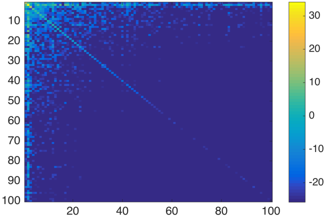
$$\mathbb{E}\left(\|e\|_2^2\right) = \mathbb{E}\left(\|\dot{x} - x_o\|_2^2\right) = \mathbb{E}\left(\sum_{i=1}^{N} \left(\dot{x}[i] - x_o[i]\right)^2\right).$$

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

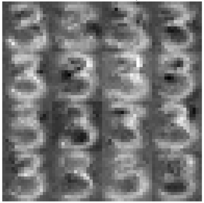
Overview

Evidence of stationarity: USPS

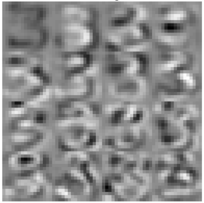
PSD matrix in dB



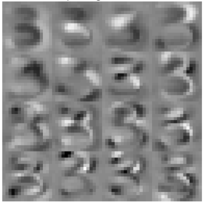
Generated samples



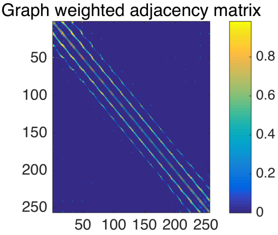
Covariance eigenvector

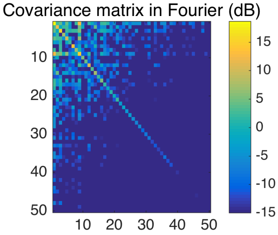


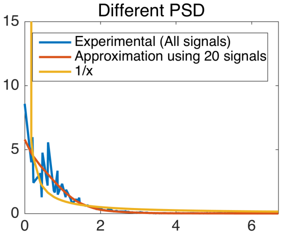
Graph eigenvectors

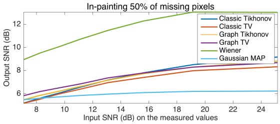


USPS digits inpainting









Conclusions

Why stationarity is great?

To go further:

Conclusions

Why stationarity is great?

- Scalable robust covariance estimator for **small number of samples**.

To go further:

Conclusions

Why stationarity is great?

- Scalable robust covariance estimator for **small number of samples**.
- Real data: **close to stationary**! faces, digits

To go further:

Conclusions

Why stationarity is great?

- Scalable robust covariance estimator for **small number of samples**.
- Real data: **close to stationary**! faces, digits
- Can be included as **prior** in optimization problems

To go further:

Conclusions

Why stationarity is great?

- Scalable robust covariance estimator for **small number of samples**.
- Real data: **close to stationary**! faces, digits
- Can be included as **prior** in optimization problems

To go further:

- Paper code available:
  `https://lts2.epfl.ch/rrp/stationarity/`

Conclusions

Why stationarity is great?

- Scalable robust covariance estimator for **small number of samples**.
- Real data: **close to stationary**! faces, digits
- Can be included as **prior** in optimization problems

To go further:

- Paper code available:
  `https://lts2.epfl.ch/rrp/stationarity/`
- Available in the GSPBox:

```
% 1) PSD estimation
psd = gsp_estimate_psd(G,X);
% 2) Prediction
S = gsp_wiener_inpainting(G, Y, Mask, psd, psd_noise);
```

Conclusions

Why stationarity is great?

- Scalable robust covariance estimator for **small number of samples**.
- Real data: **close to stationary**! faces, digits
- Can be included as **prior** in optimization problems

To go further:

- Paper code available: `https://lts2.epfl.ch/rrp/stationarity/`
- Available in the GSPBox:

```
% 1) PSD estimation
psd = gsp_estimate_psd(G,X);
% 2) Prediction
S = gsp_wiener_inpainting(G, Y, Mask, psd, psd_noise);
```

- Problem: learning the graph

Conclusions

Why stationarity is great?

- Scalable robust covariance estimator for **small number of samples**.
- Real data: **close to stationary**! faces, digits
- Can be included as **prior** in optimization problems

To go further:

- Paper code available: `https://lts2.epfl.ch/rrp/stationarity/`
- Available in the GSPBox:

```
% 1) PSD estimation
psd = gsp_estimate_psd(G,X);
% 2) Prediction
S = gsp_wiener_inpainting(G, Y, Mask, psd, psd_noise);
```

- Problem: learning the graph
- Already extended to time evolving processes

Thank you

Any question(s)?

N. Perraudin and P. Vandergheynst **(January 2016)**
Stationary signal processing on graphs.
preprint arXiv:1601.02522.