Graph Fourier Transform

Prof. Luis Gustavo Nonato

SME/ICMC - USP

August 29, 2017

Graph Fourier Transform

From previous classes we learned that the eigenvectors of a graph Laplacian behave similarly to a Fourier basis, motivating the development of graph-based Fourier analysis theory.

Graph Fourier Transform

From previous classes we learned that the eigenvectors of a graph Laplacian behave similarly to a Fourier basis, motivating the development of graph-based Fourier analysis theory.

A major obstacle to the development of a graph signal processing theory is the irregular and coordinate-free nature of a graph domain.

Graph Fourier Transform

From previous classes we learned that the eigenvectors of a graph Laplacian behave similarly to a Fourier basis, motivating the development of graph-based Fourier analysis theory.

A major obstacle to the development of a graph signal processing theory is the irregular and coordinate-free nature of a graph domain.

For instance, signal translation is basic operation for signal processing. However, that operation is not naturally implemented in graph domains.

Let $G = (V, E)$ be a weighted graph, $\mathbf{L}$ be its corresponding graph Laplacian, and $f : V \to \mathbb{R}$ a function defined on the vertices of G.

Let $G = (V, E)$ be a weighted graph, $\mathbf{L}$ be its corresponding graph Laplacian, and $f : V \to \mathbb{R}$ a function defined on the vertices of G.

Graph Fourier Transform

The Graph Fourier Transform of f is defined as

$$\mathcal{GF}[f](\lambda_l) = \hat{f}(\lambda_l) =< f, u_l >= \sum_{i=1}^{n} f(i) u_l(i)$$

Graph Fourier Transform

Let $G = (V, E)$ be a weighted graph, $\mathbf{L}$ be its corresponding graph Laplacian, and $f : V \to \mathbb{R}$ a function defined on the vertices of G.

Graph Fourier Transform

The Graph Fourier Transform of f is defined as

$$\mathcal{GF}[f](\lambda_l) = \hat{f}(\lambda_l) = < f, u_l > = \sum_{i=1}^{n} f(i) u_l(i)$$

Inverse Graph Fourier Transform

The Inverse Graph Fourier Transform is given by

$$\mathcal{IGF}[\hat{f}](i) = f(i) = \sum_{l=0}^{n-1} \hat{f}(\lambda_l) u_l(i)$$

Graph Fourier Transform

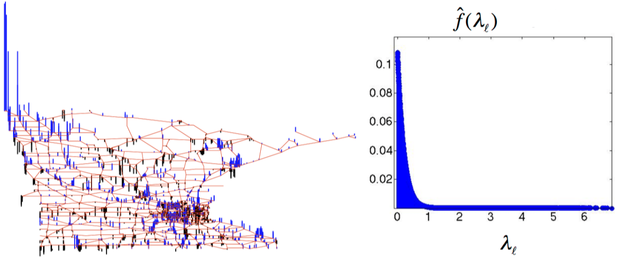
$$\hat{f}(\lambda_l) = \sum_{i=1}^{n} f(i) u_l(i)$$

The eigenvalues of $\mathbf{L}$ play the role of frequencies and the eigenvectors the Fourier basis.

$$\hat{f}(\lambda_l) = \sum_{i=1}^{n} f(i) u_l(i)$$

The eigenvalues of **L** play the role of frequencies and the eigenvectors the Fourier basis.

The graph Fourier transform (and its inverse) gives a way to represent a signal in two different domains: the vertex domain and the graph spectral domain.



Graph Fourier Transform: Synthesizing Signals

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

Graph Fourier Transform: Synthesizing Signals

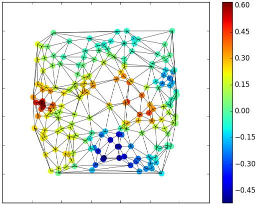
$$f(i) = \sum_{l=0}^{n-1} \boxed{\hat{f}(\lambda_l)} u_l(i) \Rightarrow f(i) = \sum_{l=0}^{n-1} \hat{g}(\lambda_l) u_l(i)$$

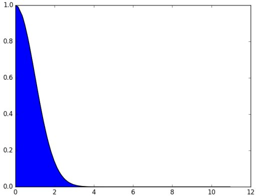
We can generate signals in the graph domain by using a kernel in the spectral domain.

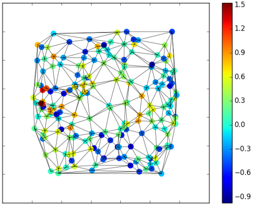
Graph Fourier Transform: Synthesizing Signals

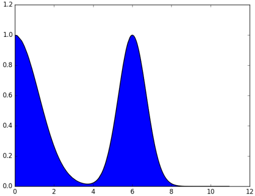
$$f(i) = \sum_{l=0}^{n-1} \boxed{\hat{f}(\lambda_l)} u_l(i) \Rightarrow f(i) = \sum_{l=0}^{n-1} \hat{g}(\lambda_l) u_l(i)$$

We can generate signals in the graph domain by using a kernel in the spectral domain.









Graph Fourier Transform: Filtering

In classical signal processing, frequency filtering is given by amplifying or attenuating the contributions of some Fourier Basis.

$$\mathcal{L}[f] = f * h \rightarrow \mathcal{F}[f * h] = \hat{f}\hat{h}$$

Graph Fourier Transform: Filtering

In classical signal processing, frequency filtering is given by amplifying or attenuating the contributions of some Fourier Basis.

$$\mathcal{L}[f] = f * h \rightarrow \mathcal{F}[f * h] = \hat{f}\hat{h}$$

Taking the IGFT

$$\mathcal{L}[f](i) = \sum_{l=0}^{n-1} \hat{f}(\lambda_l)\hat{h}(\lambda_l)u_l(i) \quad (1)$$

Suppose $\hat{h}(\lambda_l) = \sum_{k=0}^{K} a_k \lambda_l^k$ (polynomial on the spectral domain)

Graph Fourier Transform: Filtering

In classical signal processing, frequency filtering is given by amplifying or attenuating the contributions of some Fourier Basis.

$$\mathcal{L}[f] = f * h \rightarrow \mathcal{F}[f * h] = \hat{f}\hat{h}$$

Taking the IGFT

$$\mathcal{L}[f](i) = \sum_{l=0}^{n-1} \hat{f}(\lambda_l)\hat{h}(\lambda_l)u_l(i) \qquad (1)$$

Suppose $\hat{h}(\lambda_l) = \sum_{k=0}^{K} a_k \lambda_l^k$ (polynomial on the spectral domain)

Replacing $\hat{h}(\lambda_l)$ and the definition of $\hat{f}(\lambda_l)$ in Eq. (1) we get

$$\mathcal{L}[f](i) = \sum_{j=1}^{n} f(j) \sum_{k=0}^{K} a_k \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

Graph Fourier Transform: Filtering

$$\mathcal{L}[f](i) = \sum_{j=1}^{n} f(j) \sum_{k=0}^{K} a_k \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

Graph Fourier Transform: Filtering

$$\mathcal{L}[f](i) = \sum_{j=1}^{n} f(j) \sum_{k=0}^{K} a_k \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

$$(\mathbf{L}^k)_{ij} = \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

Graph Fourier Transform: Filtering

$$\mathcal{L}[f](i) = \sum_{j=1}^{n} f(j) \sum_{k=0}^{K} a_k \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

$$(\mathbf{L}^k)_{ij} = \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

It can be shown that $(\mathbf{L}^k)_{ij} = 0$ when the shortest distance between nodes i and j is greater than k.

Graph Fourier Transform: Filtering

$$\mathcal{L}[f](i) = \sum_{j=1}^{n} f(j) \sum_{k=0}^{K} a_k \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

$$(\mathbf{L}^k)_{ij} = \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

It can be shown that $(\mathbf{L}^k)_{ij} = 0$ when the shortest distance between nodes i and j is greater than k.
Defining

$$b_{ij} = \sum_{k=0}^{K} a_k \, \mathbf{L}_{ij}^k$$

Graph Fourier Transform: Filtering

$$\mathcal{L}[f](i) = \sum_{j=1}^{n} f(j) \sum_{k=0}^{K} a_k \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

$$(\mathbf{L}^k)_{ij} = \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

It can be shown that $(\mathbf{L}^k)_{ij} = 0$ when the shortest distance between nodes i and j is greater than k.
Defining

$$b_{ij} = \sum_{k=0}^{K} a_k \, \mathbf{L}_{ij}^k$$

$$\mathcal{L}[f](i) = \sum_{j \in N_k(i)}^{n} b_{ij} f(j)$$

Graph Fourier Transform: Filtering

$$\mathcal{L}[f](i) = \sum_{j=1}^{n} f(j) \sum_{k=0}^{K} a_k \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

$$(\mathbf{L}^k)_{ij} = \sum_{l=1}^{n} \lambda_l^k u_l(j) u_l(i)$$

It can be shown that $(\mathbf{L}^k)_{ij} = 0$ when the shortest distance between nodes i and j is greater than k.
Defining

$$b_{ij} = \sum_{k=0}^{K} a_k \, \mathbf{L}_{ij}^k$$

$$\mathcal{L}[f](i) = \sum_{j \in N_k(i)}^{n} b_{ij} f(j)$$

Therefore, when the filter is a polynomial in the spectral domain, the filtered signal in each node i is a linear combination of the original signal in the neighborhood of i.

Graph Fourier Transform: Convolution

Graph Fourier Transform: Convolution

The definition of convolution between two functions f and g is

$$(f * g)(x) = \int_{-\infty}^{\infty} f(x-t)g(t)dt$$

Graph Fourier Transform: Convolution

The definition of convolution between two functions f and g is

$$(f * g)(x) = \int_{-\infty}^{\infty} f(x-t)g(t)dt$$

Such definition can not be directly applied because the shift $f(x-t)$ (or singal translation) is not defined in the context of graphs.

Graph Fourier Transform: Convolution

The definition of convolution between two functions f and g is

$$(f * g)(x) = \int_{-\infty}^{\infty} f(x-t)g(t)dt$$

Such definition can not be directly applied because the shift $f(x-t)$ (or singal translation) is not defined in the context of graphs.

However, convolution can be defined as:

$$(f * g) = \mathcal{IGF}[\mathcal{GF}[(f * g)]]$$

Graph Fourier Transform: Convolution

The definition of convolution between two functions f and g is

$$(f * g)(x) = \int_{-\infty}^{\infty} f(x-t)g(t)dt$$

Such definition can not be directly applied because the shift $f(x-t)$ (or singal translation) is not defined in the context of graphs.

However, convolution can be defined as:

$$(f * g) = \mathcal{IGF}[\mathcal{GF}[(f * g)]]$$

$$(f * g)(i) = \sum_{l=1}^{n} \hat{f}(\lambda_l)\hat{g}(\lambda_l)u_l(i)$$

Graph Fourier Transform: Convolution

Some properties of the convolution:

Graph Fourier Transform: Convolution

Some properties of the convolution:

- $\widehat{f * g} = \hat{f}\hat{g}$

Graph Fourier Transform: Convolution

Some properties of the convolution:

- $\widehat{f * g} = \hat{f}\hat{g}$
- $f * g = g * f$

Graph Fourier Transform: Convolution

Some properties of the convolution:

- $\widehat{f * g} = \hat{f}\hat{g}$
- $f * g = g * f$
- $f * (g + h) = f * g + f * h$

Graph Fourier Transform: Convolution

Some properties of the convolution:

- $\widehat{f * g} = \hat{f}\hat{g}$
- $f * g = g * f$
- $f * (g + h) = f * g + f * h$
- $\sum_{i=1}^{n}(f * g)(i) = \sqrt{n}\hat{f}(0)\hat{g}(0)$

Graph Fourier Transform: Translation

Graph Fourier Transform: Translation

The classical translation

$$(T_t f)(x) = f(x - t)$$

can not be directly generalized to the graph setting.

Graph Fourier Transform: Translation

The classical translation

$$(T_t f)(x) = f(x - t)$$

can not be directly generalized to the graph setting.

However, translation can be seen as a convolution with the delta function

$$(T_t f)(x) = (f * \delta_t)(x)$$

Graph Fourier Transform: Translation

The classical translation

$$(T_t f)(x) = f(x - t)$$

can not be directly generalized to the graph setting.

However, translation can be seen as a convolution with the delta function

$$(T_t f)(x) = (f * \delta_t)(x)$$

Recalling that $\mathcal{F}[\delta(t-a)](\lambda) = e^{-\mathrm{i}2\pi\lambda a}$, we define translation to a vertex j as

$$(T_j f)(i) = \sqrt{n}(f * \delta_j)(i) = \sqrt{n} \sum_{l=0}^{n-1} \hat{f}(\lambda_l) u_l(j) u_l(i)$$

Graph Fourier Transform: Translation

The classical translation

$$(T_t f)(x) = f(x - t)$$

can not be directly generalized to the graph setting.

However, translation can be seen as a convolution with the delta function

$$(T_t f)(x) = (f * \delta_t)(x)$$

Recalling that $\mathcal{F}[\delta(t - a)](\lambda) = e^{-\mathrm{i}2\pi\lambda a}$, we define translation to a vertex j as

$$(T_j f)(i) = \sqrt{n}(f * \delta_j)(i) = \sqrt{n} \sum_{l=0}^{n-1} \hat{f}(\lambda_l) u_l(j) u_l(i)$$

The normalizing constant $\sqrt{n}$ ensures that the translation operator preserves the mean of a signal, i.e., $\sum_{i=1}^{n}(T_j f)(i) = \sum_{i=1}^{n} f(i)$

Graph Fourier Transform: Translation

Some properties of translation

- $T_j(f * g) = (T_j f) * g = f * (T_j g)$

Graph Fourier Transform: Translation

Some properties of translation

- $T_j(f * g) = (T_j f) * g = f * (T_j g)$
- $T_j T_k f = T_k T_j f$

Graph Fourier Transform: Translation

Some properties of translation

- $T_j(f * g) = (T_j f) * g = f * (T_j g)$
- $T_j T_k f = T_k T_j f$
- $\sum_{i=1}^{n} (T_j f)(i) = \sqrt{n} \hat{f}(0) = \sum_{i=1}^{n} f(i)$

Graph Fourier Transform: Translation

Some properties of translation

- $T_j(f * g) = (T_j f) * g = f * (T_j g)$
- $T_j T_k f = T_k T_j f$
- $\sum_{i=1}^{n} (T_j f)(i) = \sqrt{n}\hat{f}(0) = \sum_{i=1}^{n} f(i)$
- $\|T_j f\| \neq \|f\|$ (the energy of the signal is not preserved)

Graph Fourier Transform: Dilation

$$(D_s f)(x) = \frac{1}{s} f(\frac{1}{s}) \Longrightarrow (\widehat{D_s f})(\lambda) = \hat{f}(s\lambda)$$

Graph Fourier Transform: Dilation

$$(D_s f)(x) = \frac{1}{s} f(\frac{1}{s}) \Longrightarrow (\widehat{D_s f})(\lambda) = \hat{f}(s\lambda)$$

in the graph setting, $\frac{t}{s}$ is not well defined, impairing the direct generalization of dilation in the graph domain.

Graph Fourier Transform: Dilation

$$(D_s f)(x) = \frac{1}{s} f(\frac{1}{s}) \Longrightarrow (\widehat{D_s f})(\lambda) = \hat{f}(s\lambda)$$

in the graph setting, $\frac{t}{s}$ is not well defined, impairing the direct generalization of dilation in the graph domain.

An alternative is to define dilation via GFT:

$$(D_s f)(i) = \sum_{l=0}^{n-1} \hat{f}(s \cdot \lambda_l) u_l(i)$$

Graph Fourier Transform: Dilation

$$(D_s f)(x) = \frac{1}{s} f(\frac{1}{s}) \implies (\widehat{D_s f})(\lambda) = \hat{f}(s\lambda)$$

in the graph setting, $\frac{t}{s}$ is not well defined, impairing the direct generalization of dilation in the graph domain.

An alternative is to define dilation via GFT:

$$(D_s f)(i) = \sum_{l=0}^{n-1} \hat{f}(s \cdot \lambda_l) u_l(i)$$

Notice that $\hat{f}(s \cdot \lambda_l)$ might not be in the interval $[0, \lambda_n]$.

Graph Fourier Transform: Dilation

$$(D_s f)(x) = \frac{1}{s} f(\frac{1}{s}) \Longrightarrow (\widehat{D_s f})(\lambda) = \hat{f}(s\lambda)$$

in the graph setting, $\frac{t}{s}$ is not well defined, impairing the direct generalization of dilation in the graph domain.

An alternative is to define dilation via GFT:

$$(D_s f)(i) = \sum_{l=0}^{n-1} \hat{f}(s \cdot \lambda_l) u_l(i)$$

Notice that $\hat{f}(s \cdot \lambda_l)$ might not be in the interval $[0, \lambda_n]$. Therefore, dilation can only be used when f is generated from a kernel defined in the whole spectral domain.

Graph Fourier Transform: Modulation

Continous case:

$$M_\lambda f(t) = e^{-(2\pi \mathrm{i} \lambda t)} f(t)$$

Graph Fourier Transform: Modulation

Continous case:

$$M_\lambda f(t) = e^{-(2\pi \mathrm{i} \lambda t)} f(t)$$

In the graph setting we can define

$$M_{\lambda_l} f(i) = \sqrt{n}\, u_l(i) f(i)$$

Graph Fourier Transform: Modulation

Continous case:

$$M_\lambda f(t) = e^{-(2\pi \mathrm{i} \lambda t)} f(t)$$

In the graph setting we can define

$$M_{\lambda_l} f(i) = \sqrt{n}\, u_l(i) f(i)$$

In contrast to the continous case, modulation in the graph setting does not correspond to a translation in the spectral domain.

Graph Fourier Transform: Modulation

Continous case:

$$M_\lambda f(t) = e^{-(2\pi \mathrm{i} \lambda t)} f(t)$$

In the graph setting we can define

$$M_{\lambda_l} f(i) = \sqrt{n}\, u_l(i) f(i)$$

In contrast to the continous case, modulation in the graph setting does not correspond to a translation in the spectral domain.
However, if f is generated from a kernel localized in 0, than $\widehat{M_{\lambda_l} f}(i)$ is concentrated in λ_l.