

# Notes on Graph Laplacian, Fourier Transform, and Spectral Filtering

## Contents

<b>1 Graph Laplacian</b>	<b>1</b>
1.1 Adjacency, Degree and Laplacian . . . . .	1
1.2 Nodewise interpretation . . . . .	2
1.3 Global smoothness measure . . . . .	2
1.4 Spectral decomposition of $L$ . . . . .	3
<b>2 Graph Fourier Transform</b>	<b>3</b>
2.1 Relation to classical Fourier transform . . . . .	3
2.2 Graph Fourier transform (GFT) . . . . .	3
2.3 Inverse GFT . . . . .	3
2.4 Interpretation . . . . .	4
<b>3 Graph Spectral Filtering</b>	<b>4</b>
3.1 Filtering in the spectral domain . . . . .	4
3.2 Spectral filtering process . . . . .	5
3.3 Filtering as a function of Laplacian . . . . .	5
3.4 Polynomial filters . . . . .	5
<b>4 Convolution on Graphs</b>	<b>6</b>
4.1 $L^k$ and $k$ -hop neighborhoods . . . . .	6
<b>5 Simplified Parametric Filter and GCN</b>	<b>6</b>
5.1 General first-order spectral filter . . . . .	6
5.2 Using normalized Laplacian . . . . .	6
5.3 Kipf-Welling simplification . . . . .	7
5.4 Resulting propagation rule . . . . .	7
5.5 GCN form . . . . .	7

## 1 Graph Laplacian

Let a graph be  $G = (V, E)$  with  $|V| = N$ . A graph signal is a function

$$f : V \rightarrow \mathbb{R}, \quad f = (f(1), f(2), \dots, f(N))^\top.$$

### 1.1 Adjacency, Degree and Laplacian

Let  $W$  be the adjacency matrix:

$$W_{ij} = \begin{cases} w_{ij} > 0, & (i, j) \in E, \\ 0, & \text{otherwise.} \end{cases}$$

Degree matrix:

$$D = \text{diag}(d_1, \dots, d_N), \quad d_i = \sum_j W_{ij}.$$

The **combinatorial graph Laplacian**:

$$L = D - W.$$

## 1.2 Nodewise interpretation

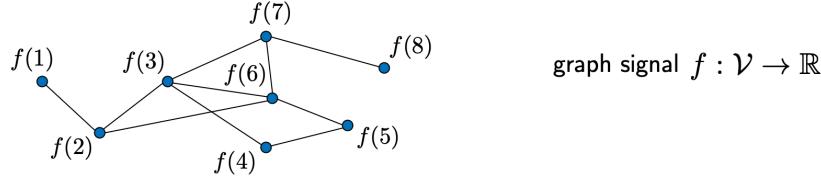
The  $i$ -th entry of  $Lf$  is

$$(Lf)(i) = \sum_{j=1}^N W_{ij} (f(i) - f(j)).$$

Hence:

- If  $f(i)$  is very different from its neighbors  $\{f(j)\}$ , then  $(Lf)(i)$  becomes large.
- If  $f$  is locally smooth around node  $i$ , then  $(Lf)(i) \approx 0$ .

## Graph Laplacian



$$\begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \\ f(5) \\ f(6) \\ f(7) \\ f(8) \end{pmatrix}$$

$$\begin{pmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \\ f(5) \\ f(6) \\ f(7) \\ f(8) \end{pmatrix}^T \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & 4 & -1 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & -1 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} f(1) \\ f(2) \\ f(3) \\ f(4) \\ f(5) \\ f(6) \\ f(7) \\ f(8) \end{pmatrix}$$

$$Lf(i) = \sum_{j=1}^N W_{ij} (f(i) - f(j))$$

$$f^T L f = \frac{1}{2} \sum_{i,j=1}^N W_{ij} (f(i) - f(j))^2$$

a measure of “smoothness”

## 1.3 Global smoothness measure

A common global smoothness functional is

$$f^T L f = \frac{1}{2} \sum_{i,j=1}^N W_{ij} (f(i) - f(j))^2.$$

Thus:

- $f^T L f$  is small  $\iff f$  varies slowly over edges.
- $f^T L f$  large  $\iff$  signal has many “jumps” across edges.

## 1.4 Spectral decomposition of $L$

Since  $L$  is symmetric and positive semidefinite, it has an orthonormal set of eigenvectors

$$L = X\Lambda X^\top,$$

where:

- $X = [\chi_0 \ \chi_1 \ \dots \ \chi_{N-1}]$  with  $X^\top X = I$ , -  $\Lambda = \text{diag}(\lambda_0, \dots, \lambda_{N-1})$ , with

$$0 = \lambda_0 \leq \lambda_1 \leq \dots \leq \lambda_{N-1}.$$

Eigenvectors play the role of “graph Fourier modes.”

### Graph Laplacian



- $L$  has a complete set of orthonormal eigenvectors:  $L = \chi\Lambda\chi^T$

$$L = \begin{bmatrix} & & & \\ \chi_0 & \cdots & \chi_{N-1} & \\ & & & \end{bmatrix} \begin{bmatrix} \lambda_0 & & 0 & \\ & \ddots & & \\ 0 & & \lambda_{N-1} & \\ & & & \end{bmatrix} \begin{bmatrix} \chi_0^T \\ \vdots \\ \chi_{N-1}^T \end{bmatrix}$$

$$\chi \quad \Lambda \quad \chi^T$$

- Eigenvalues are usually sorted increasingly:  $0 = \lambda_0 < \lambda_1 \leq \dots \leq \lambda_{N-1}$

## 2 Graph Fourier Transform

### 2.1 Relation to classical Fourier transform

Classical Fourier transform uses eigenfunctions of the 1D Laplace operator  $-\nabla^2$ :

$$(-\nabla^2) e^{j\omega x} = \omega^2 e^{j\omega x}.$$

Thus sinusoids are eigenfunctions of the operator.

On graphs, replace the Laplace operator with graph Laplacian:

$$L\chi_\ell = \lambda_\ell\chi_\ell.$$

### 2.2 Graph Fourier transform (GFT)

Given graph signal  $f \in \mathbb{R}^N$ , its GFT coefficients are:

$$\hat{f}(\ell) = \langle \chi_\ell, f \rangle = \chi_\ell^\top f.$$

This is a projection of  $f$  onto eigenvector  $\chi_\ell$ .

### 2.3 Inverse GFT

Because eigenvectors form an orthonormal basis:

$$f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_\ell(i), \quad f = X\hat{f}.$$

## 2.4 Interpretation

-  $\lambda_0 = 0$  corresponds to the constant eigenvector  $\rightarrow$  *lowest frequency*. - Larger  $\lambda_\ell$  correspond to eigenvectors with more oscillations  $\rightarrow$  *higher frequencies*. Thus GFT decomposes  $f$  into low/high-frequency variations on the graph.

### Graph Fourier transform



- The Laplacian  $L$  admits the following eigendecomposition:  $L\chi_\ell = \lambda_\ell\chi_\ell$

$$\begin{aligned}
 & \text{one-dimensional Laplace operator: } -\nabla^2 \\
 & \downarrow \\
 & \text{eigenfunctions: } e^{j\omega x} \\
 & \downarrow \\
 & \text{Classical FT: } \hat{f}(\omega) = \int [e^{j\omega x}]^* f(x) dx \\
 & \quad f(x) = \frac{1}{2\pi} \int \hat{f}(\omega) e^{j\omega x} d\omega
 \end{aligned}$$

$$\begin{aligned}
 & \text{graph Laplacian: } L \\
 & \downarrow \\
 & \text{eigenvectors: } \chi_\ell \\
 & \downarrow \\
 & f : V \rightarrow \mathbb{R}^N \\
 & \text{Graph FT: } \hat{f}(\ell) = \langle \chi_\ell, f \rangle = \sum_{i=1}^N \chi_\ell^*(i) f(i) \\
 & \quad f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_\ell(i)
 \end{aligned}$$

## 3 Graph Spectral Filtering

### 3.1 Filtering in the spectral domain

#### Graph spectral filtering



$$\begin{aligned}
 & \text{GFT: } \hat{f}(\ell) = \langle \chi_\ell, f \rangle = \sum_{i=1}^N \chi_\ell^*(i) f(i) \quad f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_\ell(i) \\
 & \xrightarrow{\text{GFT}} \hat{f}(\ell) \xrightarrow{\text{filtering}} \hat{g}(\lambda_\ell) \hat{f}(\ell) \xrightarrow{\text{IGFT}} f(i) = \sum_{\ell=0}^{N-1} \hat{g}(\lambda_\ell) \hat{f}(\ell) \chi_\ell(i) \\
 & \begin{array}{c} \text{Diagram showing } f \rightarrow \hat{f}(\ell) \text{ (GFT), } \hat{f}(\ell) \rightarrow \hat{g}(\lambda_\ell) \hat{f}(\ell) \text{ (filtering), and } \hat{g}(\lambda_\ell) \hat{f}(\ell) \rightarrow f(i) \text{ (IGFT). Below is a plot of } \hat{g}(\lambda_\ell) \text{ versus } \lambda_\ell. \\ \text{The plot shows a blue curve } \hat{g}(\lambda_\ell) \text{ decreasing from approximately 0.8 at } \lambda_0 \text{ to 0.1 at } \lambda_{N-1}. A red dot marks the value } \hat{g}(\lambda_\ell) \text{ at a specific eigenvalue } \lambda_\ell. \end{array}
 \end{aligned}$$

A graph filter  $g$  is defined by its action on GFT coefficients:

$$\hat{f}(\ell) \mapsto \hat{g}(\lambda_\ell) \hat{f}(\ell).$$

### 3.2 Spectral filtering process

$$f \xrightarrow{\text{GFT}} \hat{f} = X^\top f,$$

$$\hat{f}(\ell) \xrightarrow{\text{spectral multiplication}} \hat{g}(\lambda_\ell) \hat{f}(\ell),$$

$$\text{IGFT} \quad f_{\text{out}} = X \hat{g}(\Lambda) X^\top f.$$

### 3.3 Filtering as a function of Laplacian

Since

$$L = X \Lambda X^\top,$$

we define the operator

$$g(L) := X \hat{g}(\Lambda) X^\top.$$

Thus spectral filtering becomes the linear operator:

$$f_{\text{out}} = g(L) f.$$

### 3.4 Polynomial filters

Often  $\hat{g}(\lambda)$  is chosen as a polynomial:

$$\hat{g}_\theta(\lambda) = \sum_{j=0}^K \theta_j \lambda^j.$$

Then

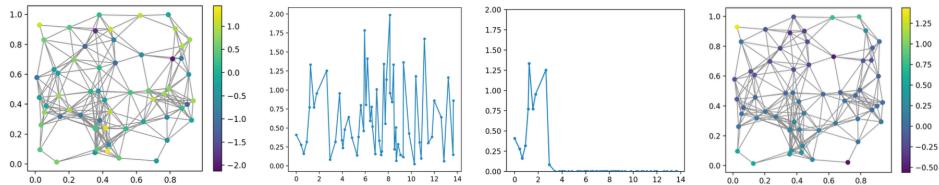
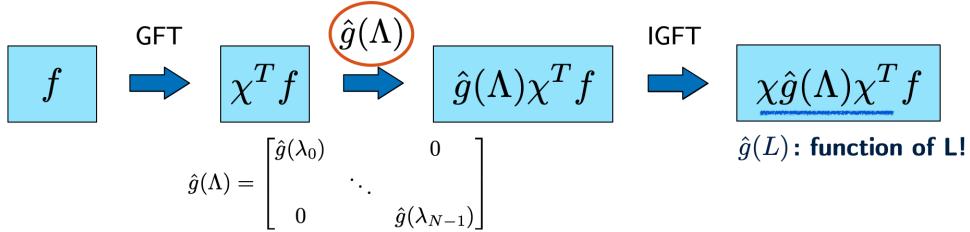
$$g_\theta(L) = \sum_{j=0}^K \theta_j L^j.$$

Advantage: no need to compute eigenvectors. Computation depends only on multiplication by  $L$ .

#### Graph spectral filtering



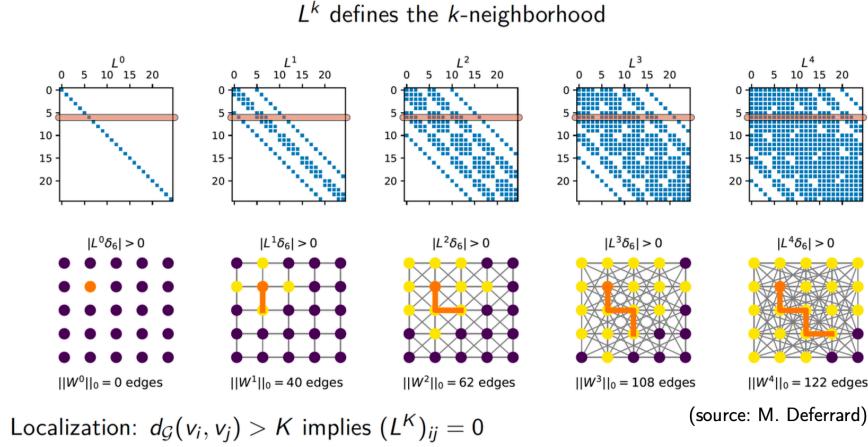
$$\text{GFT: } \hat{f}(\ell) = \langle \chi_\ell, f \rangle = \sum_{i=1}^N \chi_\ell^*(i) f(i) \quad f(i) = \sum_{\ell=0}^{N-1} \hat{f}(\ell) \chi_\ell(i)$$



## 4 Convolution on Graphs

### 4.1 $L^k$ and $k$ -hop neighborhoods

#### Powers of graph Laplacian



Observe:

- $L = D - W$  contains information of 1-hop connections.
- Multiplying  $Lf$  mixes information from 1-hop neighbors.
- $L^2 f$  mixes information from 2-hop neighbors, etc.

Formally, for adjacency matrix  $A = W$ :

$$(A^k)_{ij} > 0 \iff \text{there exists a length-}k \text{ walk between } i \text{ and } j.$$

Thus:

- $L^0 f = f$  (0-hop) -  $Lf$  aggregates 1-hop neighbor differences
- $L^2 f$  aggregates 2-hop information
- $L^k f$  spreads information over  $k$ -hop neighborhoods

This yields a notion of graph convolution: polynomial in  $L$ .

## 5 Simplified Parametric Filter and GCN

### 5.1 General first-order spectral filter

Take a first-order polynomial:

$$\hat{g}_\theta(\lambda) = \theta_0 + \theta_1 \lambda.$$

Then

$$g_\theta(L) = \theta_0 I + \theta_1 L.$$

### 5.2 Using normalized Laplacian

Normalized Laplacian:

$$L_{\text{norm}} = I - D^{-1/2} W D^{-1/2}.$$

Substitute into filter:

$$g_\theta(L_{\text{norm}}) = \theta_0 I + \theta_1 (I - D^{-1/2} W D^{-1/2}) = (\theta_0 + \theta_1) I - \theta_1 D^{-1/2} W D^{-1/2}.$$

## A simplified parametric filter

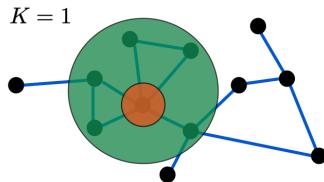


$$f * g = \chi \hat{g}(\Lambda) \chi^T f = \hat{g}(L) f$$



simplified parametric filter

$$\hat{g}_\theta(L) = \sum_{j=0}^K \theta_j L^j$$



**normalised Laplacian**

$$\begin{aligned} L_{\text{norm}} &= D^{-\frac{1}{2}} L D^{-\frac{1}{2}} \\ &= D^{-\frac{1}{2}} (D - W) D^{-\frac{1}{2}} \\ &= I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}} = I - W_{\text{norm}} \end{aligned}$$

**K = 1**

$$\xrightarrow{\text{normalised Laplacian}} = \theta_0 I - \theta_1 (D^{-\frac{1}{2}} W D^{-\frac{1}{2}})$$

(localisation within **1-hop** neighbourhood)

$$\xrightarrow{\alpha = \theta_0 = -\theta_1} = \alpha (I + D^{-\frac{1}{2}} W D^{-\frac{1}{2}})$$

renormalisation

$$\xrightarrow{\alpha (\tilde{D}^{-\frac{1}{2}} \tilde{W} \tilde{D}^{-\frac{1}{2}})}$$

**renormalisation**

$$\tilde{W} = W + I \quad \tilde{D} = D + I$$

### 5.3 Kipf–Welling simplification

Impose constraint:

$$\theta_0 = -\theta_1 = \alpha.$$

Then

$$g_\alpha(L_{\text{norm}}) = \alpha(I + D^{-1/2} W D^{-1/2}).$$

This is the key simplification enabling:

- locality (1-hop only),
- numerical stability,
- avoiding eigen-decomposition.

### 5.4 Resulting propagation rule

For node  $i$ :

$$y_i = \alpha f_i + \alpha \sum_{j \sim i} \frac{1}{\sqrt{d_i d_j}} f_j.$$

For unweighted graphs with constant degree  $d_i = d = 4$  (grid example):

$$y_i = \alpha f_i + \frac{\alpha}{4} \sum_{j \sim i} f_j.$$

### 5.5 GCN form

Add self-loops and re-normalize:

$$\tilde{A} = A + I, \quad \tilde{D}_{ii} = \sum_j \tilde{A}_{ij}.$$

GCN layer becomes:

$$H^{(l+1)} = \sigma \left( \tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} H^{(l)} W^{(l)} \right).$$

This is exactly the spatial message-passing form but derived from spectral filtering.

## A simplified parametric filter



$$f * g = \chi \hat{g}(\Lambda) \chi^T f = \hat{g}(L) f$$



simplified parametric filter

$$\hat{g}_\alpha(L) = \alpha(I + D^{-\frac{1}{2}} W D^{-\frac{1}{2}})$$



$$y_i = \alpha f_i + \alpha \frac{1}{\sqrt{d_i}} \sum_{j:(i,j) \in \mathcal{E}} w_{ij} \frac{1}{\sqrt{d_j}} f_j$$



unitary edge weights

$$y_i = \alpha f_i + \frac{1}{4} \alpha \sum_{j:(i,j) \in \mathcal{E}} f_j$$

3 <sub>0</sub>	3 <sub>1</sub>	2 <sub>2</sub>	1	0
0 <sub>2</sub>	0 <sub>2</sub>	1 <sub>0</sub>	3	1
3 <sub>0</sub>	1 <sub>1</sub>	2 <sub>2</sub>	2	3
2	0	0	2	2
2	0	0	0	1

12.0	12.0	17.0
10.0	17.0	19.0
9.0	6.0	14.0

