What is a Graph Neural Network ?

From Fourier Transform to Graph Neural Networks (GNNs)

Jun Wang

Department of Computer Science and Information Engineering
National Taiwan University

April 4, 2025

Overview

1. Fourier Transform
2. Graph Fourier Transform
3. Graph Filter

Fourier Transform

- Orthogonality of Trigonometric Functions
- Fourier Series Expansion of a Function with Period 2π
- Complex Form of the Fourier Series
- Fourier Transform

Orthogonality of Trigonometric Functions

First, we introduce what the system of trigonometric functions.

trigonometric functions

$\{1, \sin(x), \cos(x), \sin(2x), \cos(2x), \ldots, \sin(nx), \cos(nx), \ldots\}$

In the trigonometric function system mentioned above, for any two different functions $\sin(nx)$ and $\cos(nx)$, their product over the interval $-\pi$ to π equals zero. However, the product of the same functions over the interval $-\pi$ to π is not equal to zero.

$$\int_{-\pi}^{\pi} \sin(nx)\cos(nx)\,dx = 0 \tag{1}$$

$$\int_{-\pi}^{\pi} \sin^2(nx)\,dx \neq 0, \quad \int_{-\pi}^{\pi} \cos^2(nx)\,dx \neq 0 \tag{2}$$

Fourier Series Expansion of a Function with Period 2π

With the concept of the trigonometric function system in place, we now introduce the Fourier series expansion of a function $f(x)$ that is periodic with period 2π, i.e., $f(x) = f(x + 2\pi)$.

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos(nx) + b_n \sin(nx)) \tag{3}$$

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(nx)\, dx, \quad b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin(nx)\, dx \tag{4}$$

$$a_0 = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x)\, dx \tag{5}$$

Fourier Series Expansion of a Function with Period 2π

Now, we extend our discussion to the Fourier series expansion of a function with period $2L$, i.e., $f(t) = f(t + 2L)$. To establish a connection with the previous case (where the period is 2π), we make the substitution:

$$x = \frac{\pi}{L} t \quad \text{so that} \quad t = \frac{L}{\pi} x$$

This change of variable maps the interval $[-L, L]$ in t-space to $[-\pi, \pi]$ in x-space, allowing us to apply the same trigonometric basis as before.

L

L is half of the period, and it is used to facilitate the derivation of Fourier coefficients, especially over symmetric intervals like $[-L, L]$ or $[-\pi, \pi]$.

Fourier Series Expansion of a Function with Period 2π

With the change of variables $x = \frac{\pi}{L}t$, we can now express the Fourier series of the function $f(t)$, which is periodic with period $2L$, as:

$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos\left(\frac{n\pi t}{L}\right) + b_n \sin\left(\frac{n\pi t}{L}\right) \right) \tag{6}$$

Where the coefficients a_n, b_n, and a_0 are given by:

$$a_n = \frac{1}{L}\int_{-L}^{L} f(t)\cos\left(\frac{n\pi t}{L}\right) dt, \quad b_n = \frac{1}{L}\int_{-L}^{L} f(t)\sin\left(\frac{n\pi t}{L}\right) dt \tag{7}$$

$$a_0 = \frac{1}{L}\int_{-L}^{L} f(t)\, dt \tag{8}$$

Fourier Series Expansion of a Function with Period 2π

However, in practical engineering applications, the variable t typically starts from time zero. Therefore, we introduce the notation $T = 2L$, and define the angular frequency $\omega = \frac{\pi}{L} = \frac{2\pi}{T}$.

$$T = 2L, \quad \omega = \frac{\pi}{L} = \frac{2\pi}{T} \tag{9}$$

This allows us to rewrite the Fourier series in a more standard engineering form, where the frequency components are expressed in terms of ω and the period T.

Fourier Series Expansion of a Function with Period 2π

Using the definition $\omega = \frac{2\pi}{T}$, the Fourier series of a function $f(t)$ with period T can be written as:

$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos(n\omega t) + b_n \sin(n\omega t)\right) \tag{10}$$

Where the coefficients are given by:

$$a_n = \frac{2}{T} \int_0^T f(t) \cos(n\omega t)\, dt, \quad b_n = \frac{2}{T} \int_0^T f(t) \sin(n\omega t)\, dt \tag{11}$$

$$a_0 = \frac{2}{T} \int_0^T f(t)\, dt \tag{12}$$

This is the standard form of the Fourier series in engineering, where time starts at 0 and the periodic signal is decomposed into harmonics of the fundamental angular frequency $\omega = \frac{2\pi}{T}$.

Complex Form of the Fourier Series

Based on Euler's formula:

$$e^{inx} = \cos(nx) + i\sin(nx), \quad e^{-inx} = \cos(nx) - i\sin(nx) \tag{13}$$

By substituting this into Equation 10, we can obtain the complex form of the Fourier series:

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e^{in\omega t} \tag{14}$$

Where the complex Fourier coefficients c_n are given by:

$$c_n = \frac{1}{T}\int_0^T f(t)e^{-in\omega t}\,dt$$

Fourier Transform

Given that $f(t)$ is periodic with period T, i.e.,

$$f_T(t) = f(t + T) \tag{15}$$

Its complex form of the Fourier series expansion is:

$$f_T(t) = \sum_{n=-\infty}^{\infty} c_n e^{in\omega_0 t}, \quad \text{where } \omega_0 = \frac{2\pi}{T} \tag{16}$$

And the complex Fourier coefficients are calculated as:

$$c_n = \frac{1}{T}\int_0^T f_T(t)e^{-in\omega_0 t}\,dt = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} f_T(t)e^{-in\omega_0 t}\,dt \tag{17}$$

For a non-periodic function, its general form can be understood as the limit of a periodic function as the period $T \to \infty$. That is:

$$\lim_{T \to \infty} f_T(t) = f(t) \tag{18}$$

where $f_T(t)$ is a periodic extension of $f(t)$ with period T. As T becomes infinitely large, $f_T(t)$ converges to the original non-periodic function $f(t)$.

We can observe that the part that truly distinguishes one function from another is the set of coefficients c_n.

These coefficients c_n capture the amplitude and phase information of each frequency component, while the exponential basis functions $e^{in\omega_0 t}$ are the same for all signals.

we can observe that the part that truly distinguishes one function from another is the set of coefficients c_n. These coefficients c_n capture the amplitude and phase information of each frequency component, while the exponential basis functions $e^{in\omega_0 t}$ are the same for all signals. We know that c_n is a complex number. If we represent c_n in a 3D plot, it would have a real axis, an imaginary axis, and a rotation angle corresponding to $n\omega_0$.

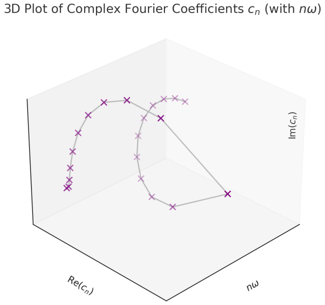

Figure: C_n

Fourier Transform

For a non-periodic function, as $T \to \infty$, the discrete frequency components become continuous. In this case:
The summation in the original Fourier series expression becomes an integral from $-\infty$ to $+\infty$, The frequency step $\Delta\omega = \omega_0 = \frac{2\pi}{T}$ becomes infinitesimally small, turning into $d\omega$, The discrete frequency $n\omega_0$ becomes a continuous variable ω, and the coefficient term c_n transitions from a discrete spectrum to a continuous function $F(\omega)$. Also, since:

$$\Delta\omega = \frac{2\pi}{T} \quad \Rightarrow \quad \frac{1}{T} = \frac{\Delta\omega}{2\pi}$$

Fourier Transform

Then, when taking the limit, the coefficient c_n becomes part of the continuous function $F(\omega)$, and the factor $\frac{1}{T}$ is replaced by $\frac{1}{2\pi}$.

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}\,dt \tag{19}$$

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}\,d\omega \tag{20}$$

This is the Fourier Transform, derived as the limiting case of the Fourier series as the period tends to infinity.

Fourier Transform

Project

The core idea of the Fourier Transform is: To project a time-domain function $f(t)$ onto a set of basis functions in the form of complex exponentials (or sines and cosines). In other words, it is a basis transformation.

What the Fourier transform essentially does is: It expresses the signal $f(t)$ as a linear combination of complex exponential basis functions $e^{-i2\pi\xi t}$:

$$f(t) = \int_{-\infty}^{\infty} F(\xi) e^{i2\pi\xi t} \, d\xi$$

And the coefficient for each frequency component—i.e., the projection onto each frequency basis—is computed by:

$$F(\xi) = \int_{-\infty}^{\infty} f(t) e^{-i2\pi\xi t} \, dt$$

Fourier Transform

To project a time-domain function $f(t)$ onto a set of basis functions in the form of complex exponentials (or sines and cosines). In other words, it is a basis transformation. Just like a 2D vector $\vec{v}$ can be written as a linear combination of basis vectors $\hat{i}$ and $\hat{j}$:

$$\vec{v} = v_x\hat{i} + v_y\hat{j} \tag{21}$$

The Fourier transform projects $f(t)$ onto the "directions" defined by frequencies ξ, where each "direction" is a periodic function (in the form of a complex exponential). These periodic functions form an orthogonal basis in the function space.

Graph Fourier Transform

- Graph Convolution
- Graph Filter
- Spectral-based GNNs

Graph Convolution

The eigendecomposition of Laplacian matrix

$$\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T = \boldsymbol{U}\begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{pmatrix}\boldsymbol{U}^T \tag{22}$$

where $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n]$, $\boldsymbol{\Lambda} = \mathrm{diag}\left([\lambda_1, \ldots, \lambda_n]\right)$, $\boldsymbol{u}_i$ and λ_i for $i \in \{1, 2, \ldots, n\}$ denote the eigenvectors and eigenvalues, respectively, and $\lambda_i \in [0, 2]$. Orthonormal basis: $\boldsymbol{U} \cdot \boldsymbol{U}^T = \boldsymbol{I}$,

Graph Convolution

Graph Fourier Transform of a signal: $\hat{\boldsymbol{x}} = \boldsymbol{U}^T \boldsymbol{x}$
Inverse Graph Fourier Transform of a signal: $\boldsymbol{x} = \boldsymbol{U}\hat{\boldsymbol{x}}$

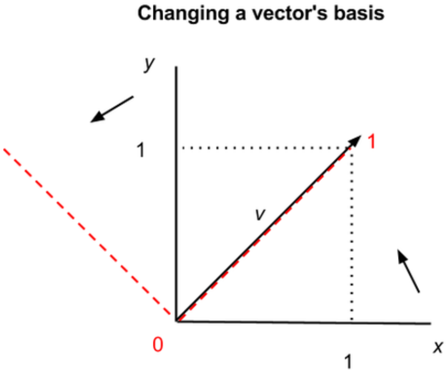

Figure: Project

$$\begin{bmatrix} \hat{\boldsymbol{x}}(1) \\ \hat{\boldsymbol{x}}(2) \\ \vdots \\ \hat{\boldsymbol{x}}(n) \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}_1(1) & \boldsymbol{u}_1(2) & \dots & \boldsymbol{u}_1(n) \\ \boldsymbol{u}_2(1) & \boldsymbol{u}_2(2) & \dots & \boldsymbol{u}_2(n) \\ \vdots & \vdots & \dots & \dots \\ \boldsymbol{u}_n(1) & \boldsymbol{u}_n(2) & \dots & \boldsymbol{u}_n(n) \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{x}(1) \\ \boldsymbol{x}(2) \\ \vdots \\ \boldsymbol{x}(n) \end{bmatrix}$$

Figure: Caption

Graph Convolution

Theorem (Convolution theorem)

(Convolution theorem): the Fourier transform of a convolution of two signals is the pointwise product of their Fourier transforms.

$$\boldsymbol{x} *_G \boldsymbol{g} = \boldsymbol{U}\left(\left(\boldsymbol{U}^T\boldsymbol{x}\right) \odot \left(\boldsymbol{U}^T\boldsymbol{g}\right)\right)$$

where $\odot$ denotes Hadamard products, $\boldsymbol{U}^T\boldsymbol{g}$ is the convolution filter. Reparametrize $\boldsymbol{U}^T\boldsymbol{g}$ as $\mathbf{diag}\left[\theta_1, \ldots, \theta_n\right]$:

Graph Convolution

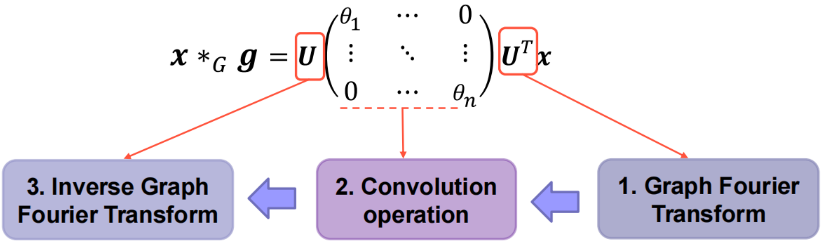

Figure: Convoluation

Graph Convolution

Further reparametrize $\theta_i = h(\lambda_i)$

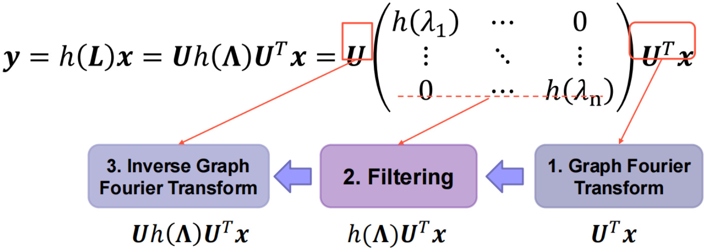

Figure: Caption

We call $h(\Lambda)/h(\lambda)$ (graph) filter

Graph Filter

Let us now consider a graph with self-loops as an example to understand how the Graph Laplacian can be decomposed and how it affects the design of graph filters (i.e., spectral responses to eigenvalues).

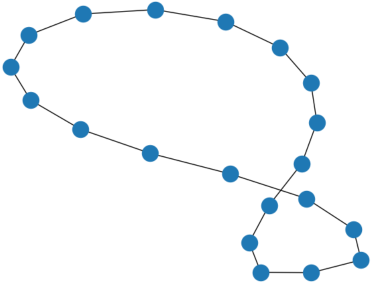

Figure: Ring Graph

Graph Filter

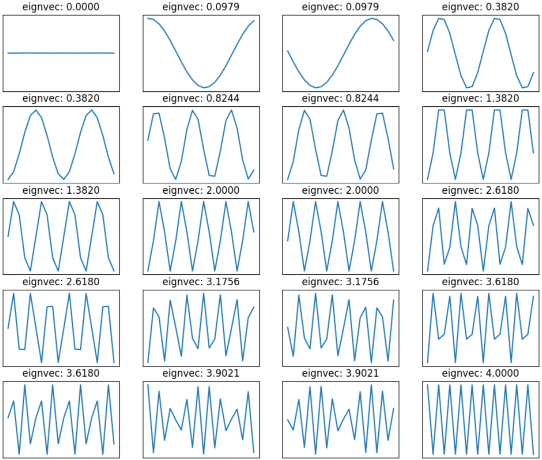

Figure: Filter

Graph Filter

From the graph, we observe that the smaller the eigenvalue, the smoother the variation of the corresponding eigenvector (i.e., the filter) across the graph.
This is because low eigenvalues correspond to slowly varying signals over the graph structure (i.e., neighboring nodes have similar values). Conversely, high eigenvalues correspond to rapidly oscillating signals.
Therefore, for homogeneous or regular graphs, we can design the following type of filter:

$$g(\lambda) = \begin{cases} 1, & \text{if } \lambda \text{ is small (low frequency)} \\ 0, & \text{if } \lambda \text{ is large (high frequency)} \end{cases} \tag{23}$$

This is a typical low-pass filter in the graph spectral domain. It preserves the smooth components of the graph signal (e.g., community structure, similar neighborhoods) while suppressing high-frequency noise.

Graph Filter

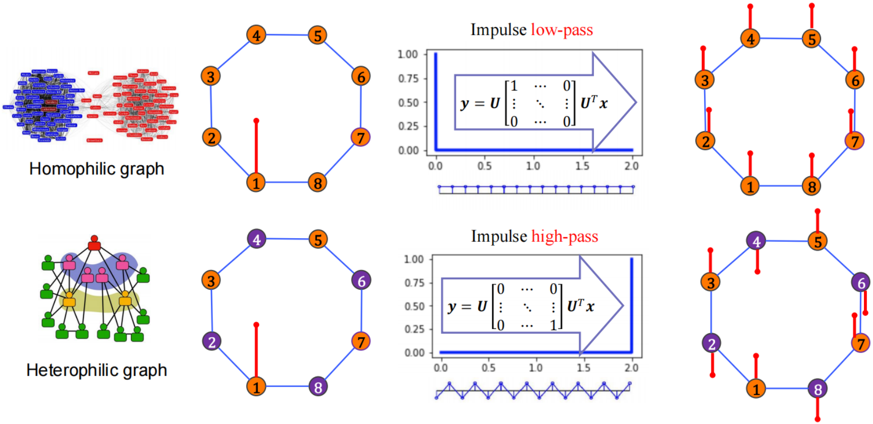

Figure: Graph Filter

Frame Title

- How to design arbitrary filters?

$$\boldsymbol{y} = \boldsymbol{U} \begin{pmatrix} h(\lambda_1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & h(\lambda_n) \end{pmatrix} \boldsymbol{U}^T \boldsymbol{x}$$

  - The $\boldsymbol{O}(\boldsymbol{n}^2)$ complexity of eigendecomposition is too high.

- Approximating filters by polynomials, complexity drops to $\boldsymbol{O}(\boldsymbol{m})$.

$$\boldsymbol{y} \approx \boldsymbol{U} \begin{pmatrix} \sum_{k=0}^{K} w_k {\lambda_1}^k & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sum_{k=0}^{K} w_k {\lambda_n}^k \end{pmatrix} \boldsymbol{U}^T \boldsymbol{x} = \sum_{k=0}^{K} w_k \boldsymbol{L}^k \boldsymbol{x}$$

Figure: Filter

Citation

An example of the `\cite` command to cite within the presentation:

This statement requires citation [Smith, 2012].

References

Smith, J. (2012).
Title of the publication.
Journal Name, 12(3):45–678.

The End

Bullet Points

- Lorem ipsum dolor sit amet, consectetur adipiscing elit
- Aliquam blandit faucibus nisi, sit amet dapibus enim tempus eu
- Nulla commodo, erat quis gravida posuere, elit lacus lobortis est, quis porttitor odio mauris at libero
- Nam cursus est eget velit posuere pellentesque
- Vestibulum faucibus velit a augue condimentum quis convallis nulla gravida