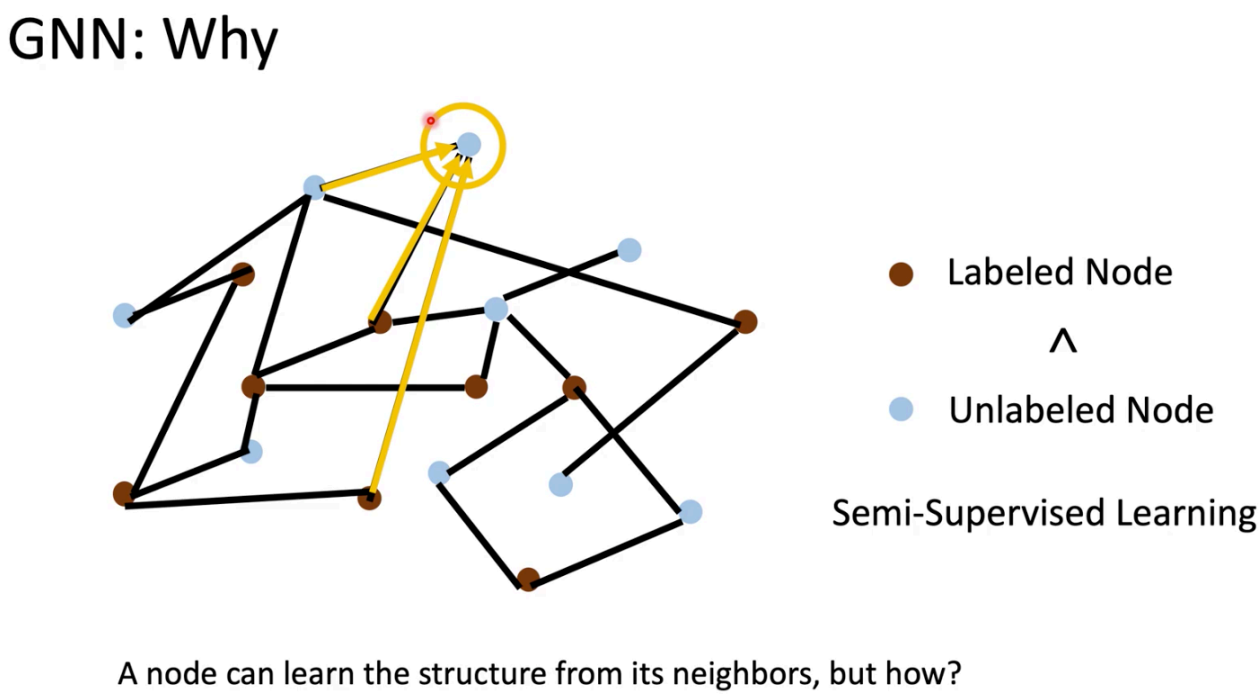
图神经网络GNN

- entity
- relationship

WHY



半监督学习semi-supervised learning

HOW 两类方法

- Spatial-based
- Spectral-based

ROADMAP

Convolution

Spatial-based

Aggregation	Method
Sum	NN4G
Mean	DCNN, DGC, GraphSAGE
Weighted sum	MoNET, GAT, GIN
LSTM	GraphSAGE
Max Pooling	GraphSAGE

Spectral-based

ChebNet → GCN → HyperGCN

Tasks

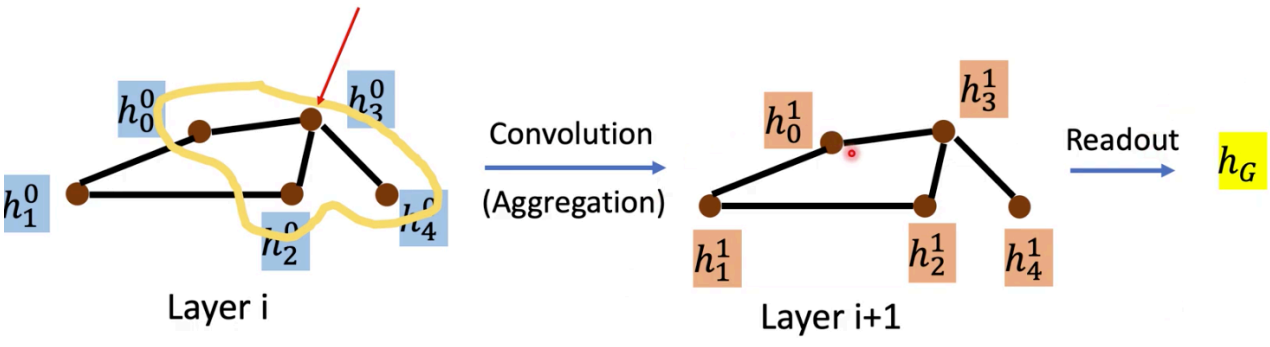
- Supervised classification
- Semi-Supervised Learning
- Representation learning: Graph InfoMax
- Generation: GraphVAE, MolGAN, etc.

TASKS DATASET BENCHMARK（基准）

Spatial-based GNN 第一类方法

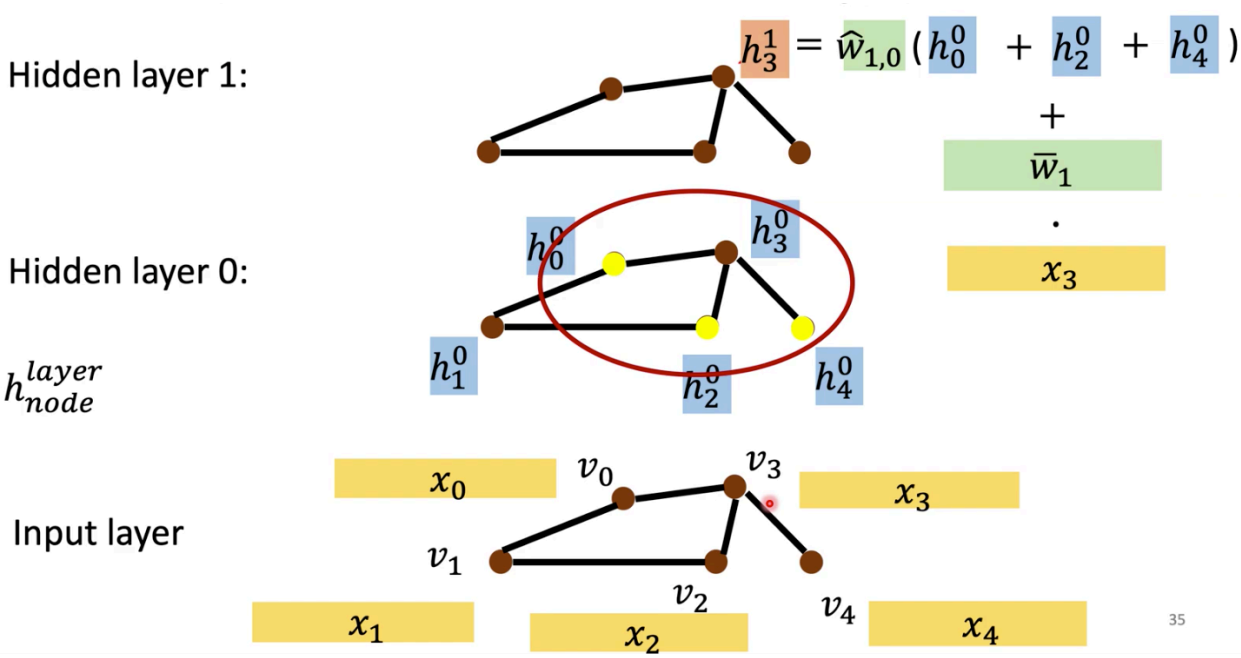
Terminology：

- Aggregate：用neighbor feature update下一层的hidden state
- Readout：把所有nodes的feature集合起来代表整个graph



NN4G （Neural Networks for Graph）

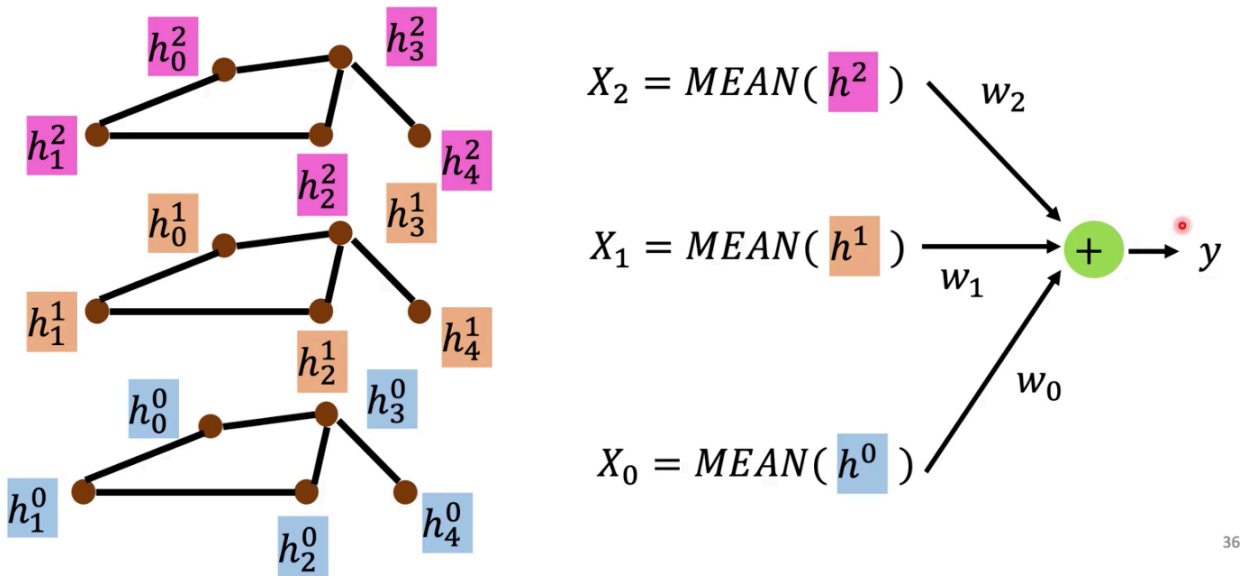
- aggregate



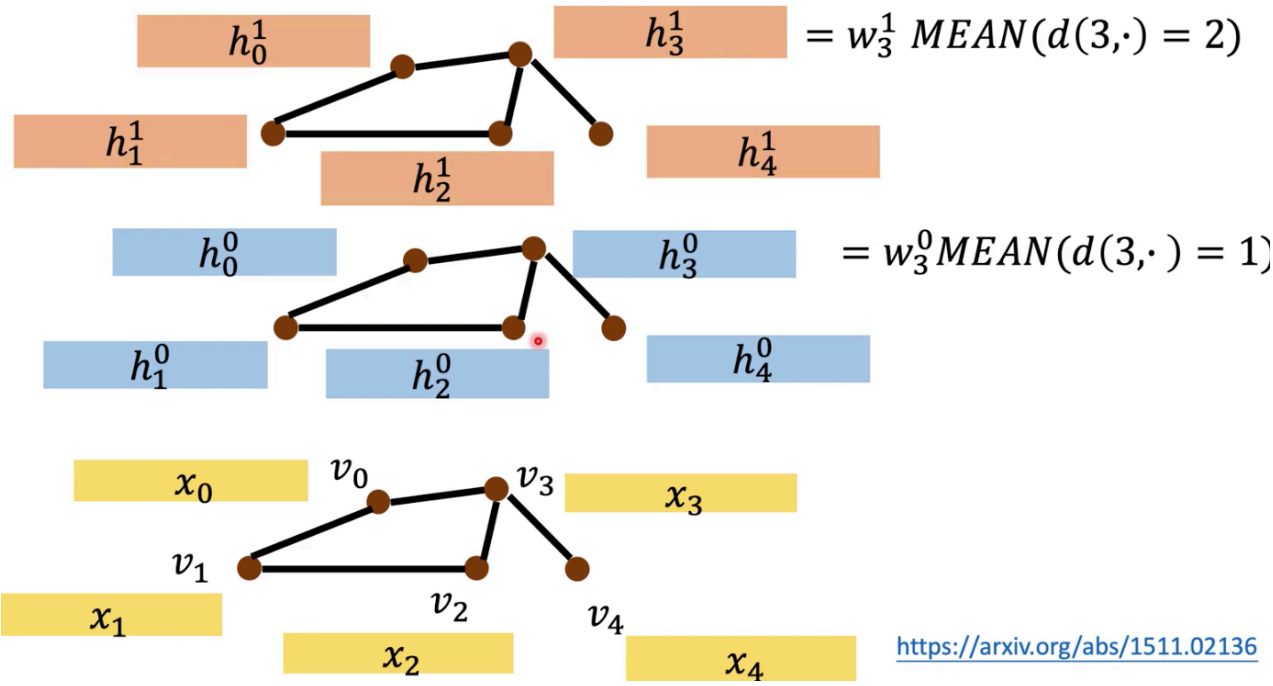
每个node等于邻居节点和自身节点相加

- readout

✓ Readout:



DCNN（Diffusion-Convolution Neural Network）

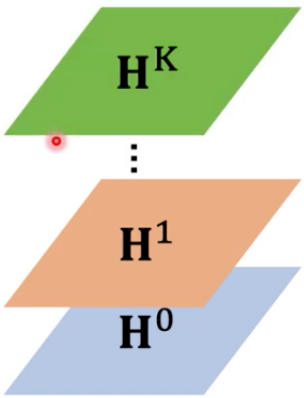


https://arxiv.org/abs/1511.02136

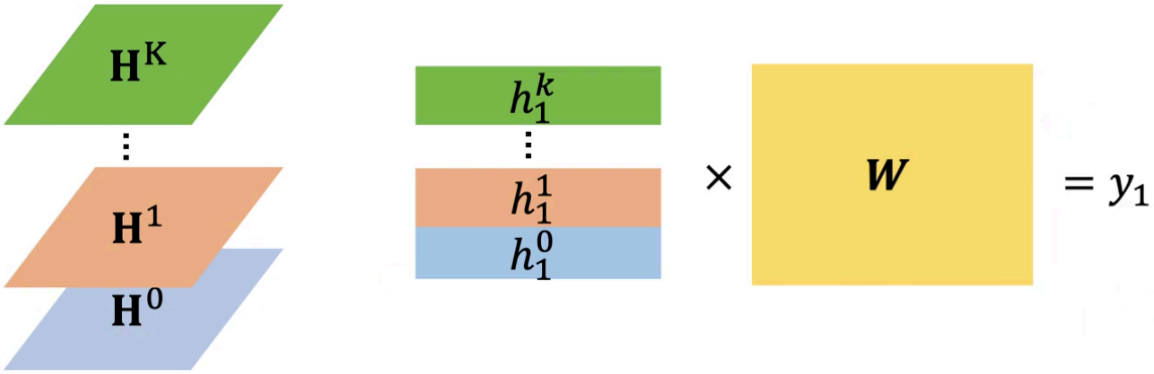
第一层将距离3号node距离为1的node求MEAN

第二层将距离3号node距离为2的node求MEAN

- 注意每次都是在原图的基础上进行求MEAN

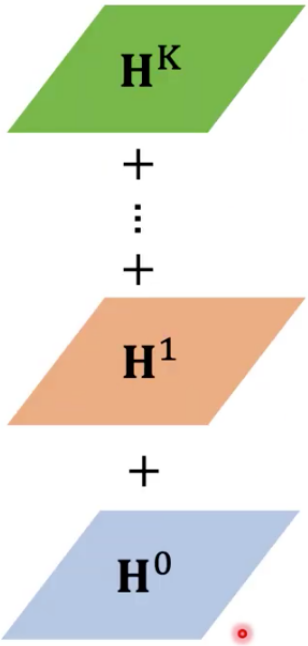


✓ Node features



提取出每个node的 最终乘一个权重矩阵 得到该node的输出

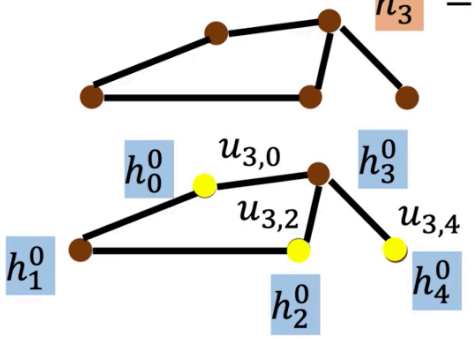
DGC（Diffusion Graph Convolution）



直接相加得到结果

MoNET（Mixture Model Networks）

- ✓ Define a measure on node ‘distances’
- ✓ Use weighted sum (mean) instead of simply summing up (averaging) neighbor features.



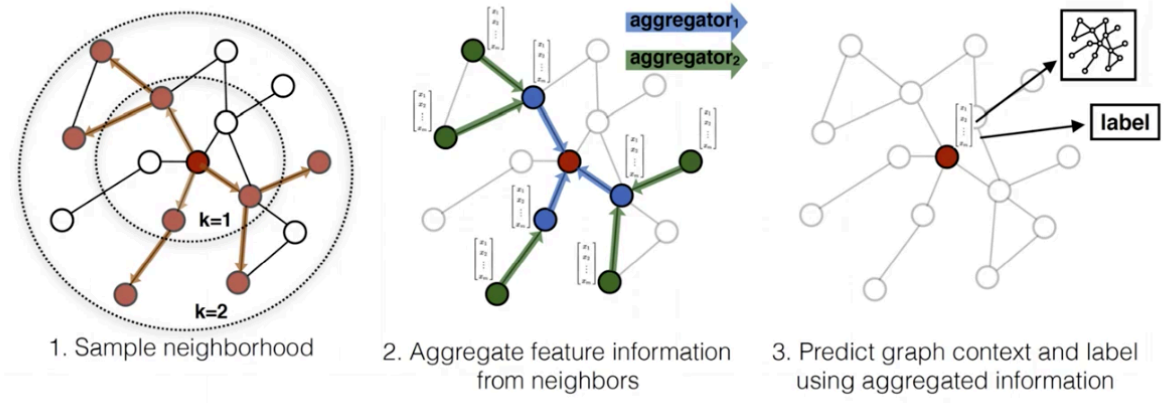
$$h_3^1 = w(\hat{u}_{3,0}) \times h_0^0 + w(\hat{u}_{3,2}) \times h_2^0 + w(\hat{u}_{3,2}) \times h_4^0$$

$w(\cdot)$ is a NN, $\hat{u}$ is a transform from u

$$\mathbf{u}(x,y) = \left(\frac{1}{\sqrt{\deg(x)}}, \frac{1}{\sqrt{\deg(y)}}\right)^\top$$

GraphSAGE

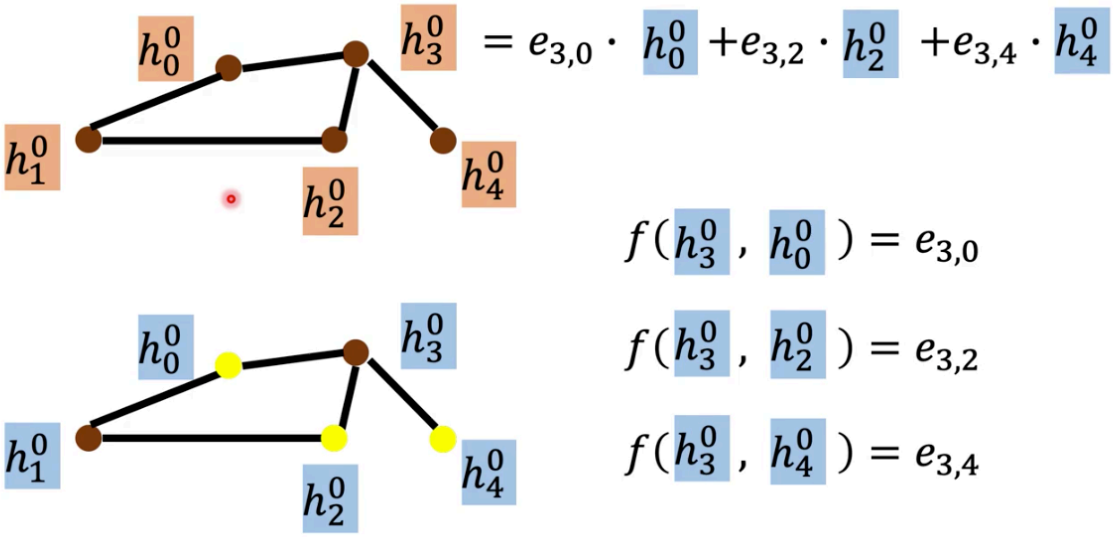
aggregation：mean、max-pooling、LSTM



LSTM用来处理序列数据 在图的处理中每次8随机形成邻接顶点序列，或许最终可以得到一个忽略序列的结果

GAT（Graph Attention Networks） 常用的模型

对邻居做Attention，求e



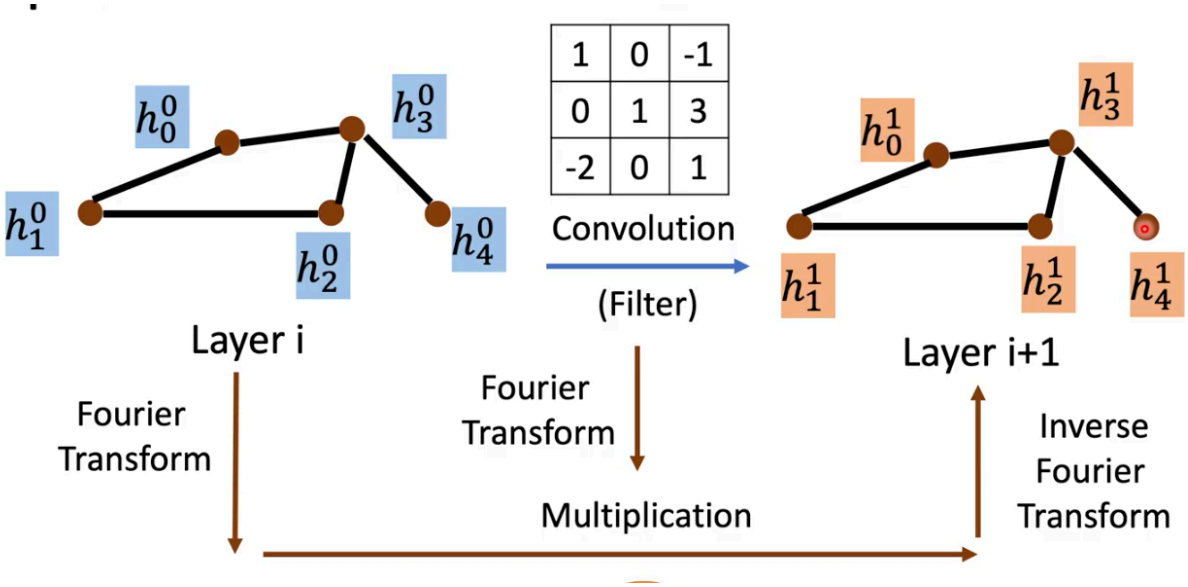
GIN（Graph Isomorphism Network）

理论证明

- sum instead of mean or max

$$h_v^{(k)} = \mathrm{MLP}^{(k)}\left(\left(1+\epsilon^{(k)}\right)\cdot h_v^{(k-1)} + \sum_{u\in\mathcal{N}(v)} h_u^{(k-1)}\right)$$

Spectral-Based Convolution



- **Spectral Graph Theory**

✓ Graph: $G = (V, E), N = |V|$

✓ $A \in \mathbb{R}^{N \times N}$, adjacency matrix (weight matrix).

$$A_{i,j} = 0 \; if \; e_{i,j} \notin E, else \; A_{i,j} = w(i,j)$$

✓ We only consider undirected graph

✓ $D \in \mathbb{R}^{N \times N}$, degree matrix

$$D_{i,j} = \begin{cases} d(i) & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad \text{(Sum of row } i \text{ in } A\text{)}$$

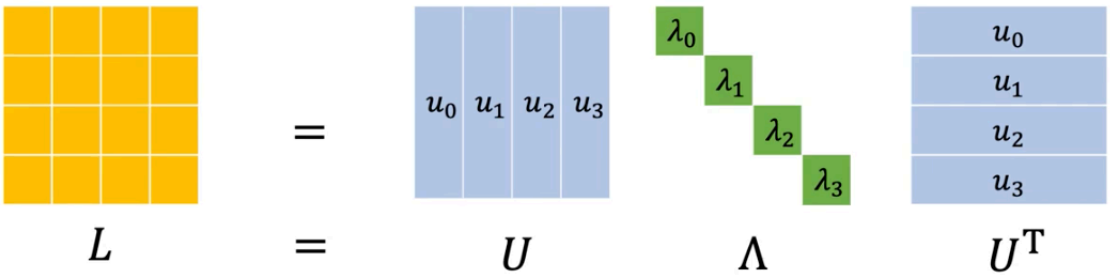
✓ $f: V \to \mathbb{R}^N$, signal on graph (vertices). $f(i)$ denotes the signal on vertex i

f表示图的信号大小（例如城市的气温或人口增长数 好像是node特征？？）

diagonal matrix 对角矩阵（只有对角线上有值）

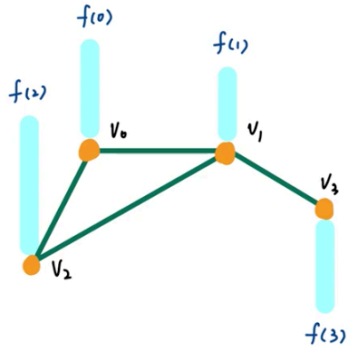
Spectral decomposition 频谱分解

- ✓ Graph Laplacian $L = D - A,\ L \succcurlyeq 0$ (Positive semidefinite)
- ✓ L is symmetric (for undirected graph)
- ✓ $L = U\Lambda U^{\mathrm{T}}$ (spectral decomposition)
- ✓ $\Lambda = \mathrm{diag}(\lambda_0, \dots, \lambda_{N-1}) \in \mathbb{R}^{N\times N}$
- ✓ $U = [u_0, \dots, u_{N-1}] \in \mathbb{R}^{N\times N}$, orthonormal
- ✓ λ_l is the frequency, u_l is the basis corresponding to λ_l



- **example**

✓ Vertex domain signal



$$D = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$L = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \quad \Lambda = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

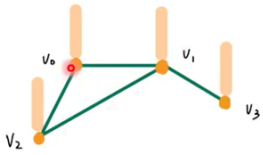
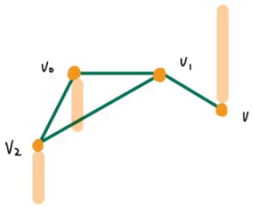
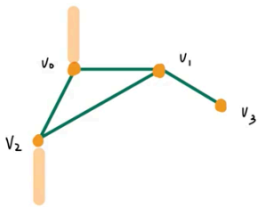
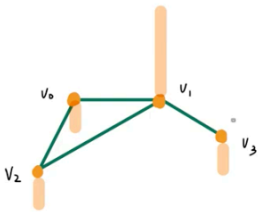
$$f = \begin{bmatrix} 4 \\ 2 \\ 4 \\ -3 \end{bmatrix} \quad U = \begin{bmatrix} 0.5 & -0.41 & 0.71 & -0.29 \\ 0.5 & 0 & 0 & 0.87 \\ 0.5 & -0.41 & -0.71 & -0.29 \\ 0.5 & 0.82 & 0 & -0.29 \end{bmatrix}$$

将U矩阵结果形象化表示：（信号）

U中的每一列都是一个eigen vector特征向量

eigen value 特征值 eigen vector 特征向量

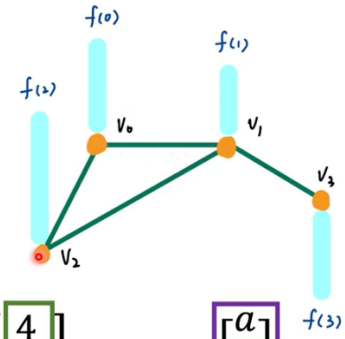
Low frequency → High frequency

λ	0	1	3	4
u	$[0.5 \quad 0.5 \quad 0.5 \quad 0.5]^T$	$[-0.41 \quad 0 \quad -0.41 \quad 0.82]^T$	$[0.71 \quad 0 \quad -0.71 \quad 0]^T$	$[-0.29 \quad 0.87 \quad -0.29 \quad -0.29]^T$
				

(DC component)

- 频率越大，相邻两点之间的信号变化量就越大

Spectral Graph Theory



✓ Interpreting vertex frequency

➢ L as an operator on graph

➢ Given a graph signal f, what does Lf mean?

➢ $Lf = (D - A)f = Df - Af$

$$D = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad f = \begin{bmatrix} 4 \\ 2 \\ 4 \\ -3 \end{bmatrix} \quad Lf = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

➢ Lets focus on the first row of Lf

➢ $a = [2\,0\,0\,0] \cdot [4\,2\,4\,-3] - [0\,1\,1\,0] \cdot [4\,2\,4\,-3]$

# of neighbors of v_0 — Signal on v_0's neighbors

$= 2 \times 4 - 2 - 4 = (4-2) + (4-4) = 2$

Signal on v_0 — Sum of difference between v_0 and its neighbors

69

L是拉普拉斯矩阵 f是每个node的信号值

Lf相当于node与相邻节点的信号差之和

✓ $(Lf)(v_i) = \sum_{v_j \in V} w_{i,j}(f(v_i) - f(v_j))$, where $w_{i,j}$ is the $(i,j)^{th}$ entry of A

$$f^T Lf = \sum_{v_i \in V} f(v_i) \sum_{v_j \in V} w_{i,j}(f(v_i) - f(v_j))$$

$$= \sum_{v_i \in V} \sum_{v_j \in V} w_{i,j}(f^2(v_i) - f(v_i)f(v_j))$$

$$= \frac{1}{2} \sum_{v_i \in V} \sum_{v_j \in V} w_{i,j}(f^2(v_i) - f(v_i)f(v_j) + f^2(j) - f(v_j)f(v_i))$$

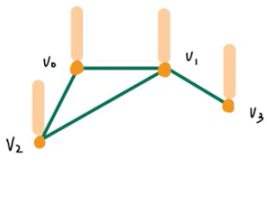
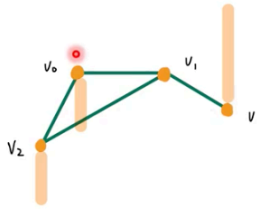
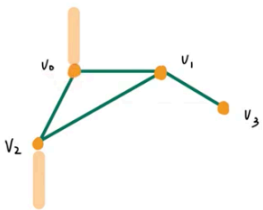
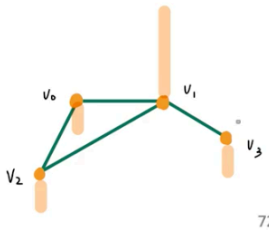
$$= \frac{1}{2} \sum_{v_i \in V} \sum_{v_j \in V} w_{i,j} \left(f(v_i) - f(v_j)\right)^2$$

70

将得到的能量取平方——两个节点之间的信号“能量”差

represent ‘power’ of signal variantion between nodes

Low frequency → High frequency

λ	0	1	3	4
u	$[0.5 \quad 0.5 \quad 0.5 \quad 0.5]^T$	$[-0.41 \quad 0 \quad -0.41 \quad 0.82]^T$	$[0.71 \quad 0 \quad -0.71 \quad 0]^T$	$[-0.29 \quad 0.87 \quad -0.29 \quad -0.29]^T$
				

(DC component)

72

顶点域->光谱域：

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

分析 $u_0 \cdot x$
分析 $u_1 \cdot x$
分析 $u_2 \cdot x$
分析 $u_3 \cdot x$

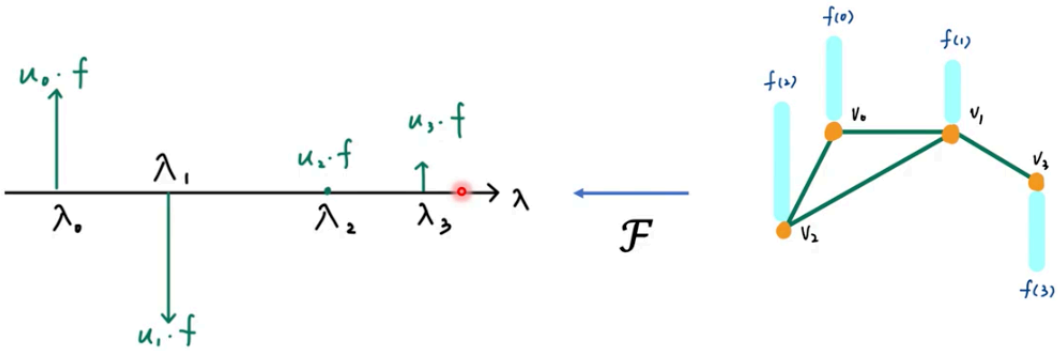
$=$

u_0
u_1
u_2
u_3

x

$\hat{x} = U^{\mathrm{T}} x$ 合成

Spectral domain Vertex domain



✓ Inverse Graph Fourier Transform of signal $\hat{x}$: $x = U\hat{x}$

x

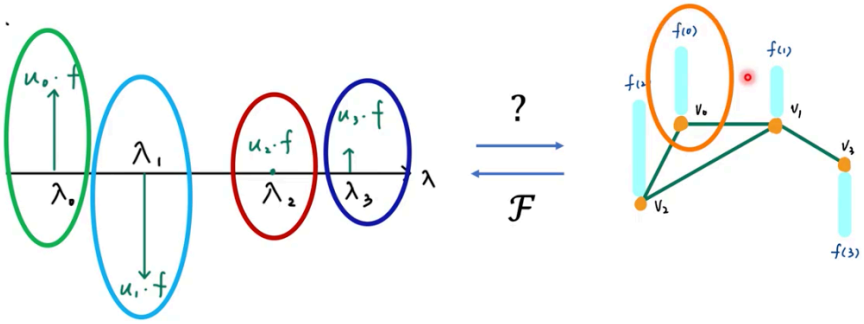
$=$

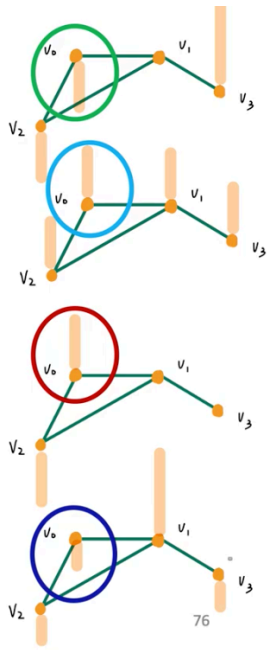
u_0 u_1 u_2 u_3

$u_0 \cdot x$
$u_1 \cdot x$
$u_2 \cdot x$
$u_3 \cdot x$

$x = U \hat{x}$ 合成

Vertex domain Spectral domain





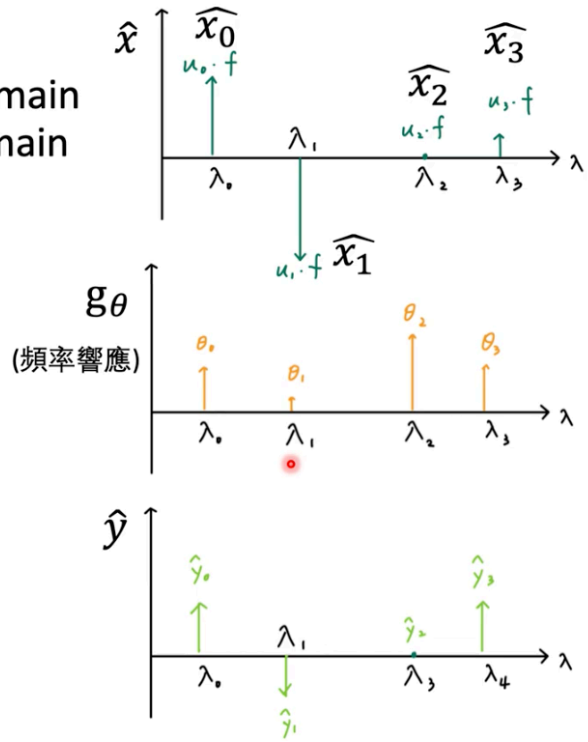
Spectral Graph Theory

- ✓ Filtering: Convolution in time domain is multiplication in frequency domain
- ✓ $\hat{y} = g_\theta(\Lambda)\,\hat{x}$

$$\begin{bmatrix}\bar{y}_0\\ \bar{y}_1\\ \bar{y}_2\\ \bar{y}_3\end{bmatrix} = \begin{bmatrix}\theta_0 & & & \\ & \theta_1 & & \\ & & \theta_2 & \\ & & & \theta_3\end{bmatrix} \begin{bmatrix}\bar{x}_0\\ \bar{x}_1\\ \bar{x}_2\\ \bar{x}_3\end{bmatrix}$$

$$\hat{y} = g_\theta(\Lambda)\,\hat{x}$$

$$g_\theta(\lambda_i) = \theta_i$$



右边上面两个相乘得到下面的

$$\begin{bmatrix}\\ \\ \\ \\ \\ \end{bmatrix} = \begin{bmatrix}u_0 & u_1 & u_2 & u_3\end{bmatrix} \begin{bmatrix}\bar{y}_0\\ \bar{y}_1\\ \bar{y}_2\\ \bar{y}_3\end{bmatrix} = \begin{bmatrix}u_0 & u_1 & u_2 & u_3\end{bmatrix} \begin{bmatrix}\theta_0 & & & \\ & \theta_1 & & \\ & & \theta_2 & \\ & & & \theta_3\end{bmatrix} \begin{bmatrix}u_0\\ u_1\\ u_2\\ u_3\end{bmatrix} \begin{bmatrix}\\ \\ \\ \\ \\ \end{bmatrix}$$

$$y \quad U \quad \hat{y} = U \quad g_\theta(\Lambda) \quad U^{\mathrm{T}} \quad x$$

$$= g_\theta(U\Lambda U^{\mathrm{T}})x$$

$$= g_\theta(L)x$$

Parameters to be learnt

$$y = g_\theta(L)x$$

$g_\theta(\cdot)$ can be any function. For example,

$$g_\theta(L) = \log(\mathrm{I} + L) = L - \frac{L^2}{2} + \frac{L^3}{3} \ldots, \lambda_{max} < 1$$

Problem 1: Learning complexity is $O(N)$!!!

遇到两个问题：

- Problem1：学习的复杂度是$O(n)$
- Problem2：not localize

ChebNet

ChebNet

✓ Solution to Problem 1 and 2:

➢Use polynomial to parametrize $g_\theta(L)$

$$g_\theta(L) = \sum_{k=0}^{K} \theta_k L^k$$

Now it is K-localized

Parameters to be learnt: $O(K)$

$$g_\theta(\Lambda) = \sum_{k=0}^{K} \theta_k \Lambda^k$$

Problem 3:

Time complexity: $O(N^2)$

$$y = U g_\theta(\Lambda) U^T x = U\left(\sum_{k=0}^{K} \theta_k \Lambda^k\right) U^T x$$

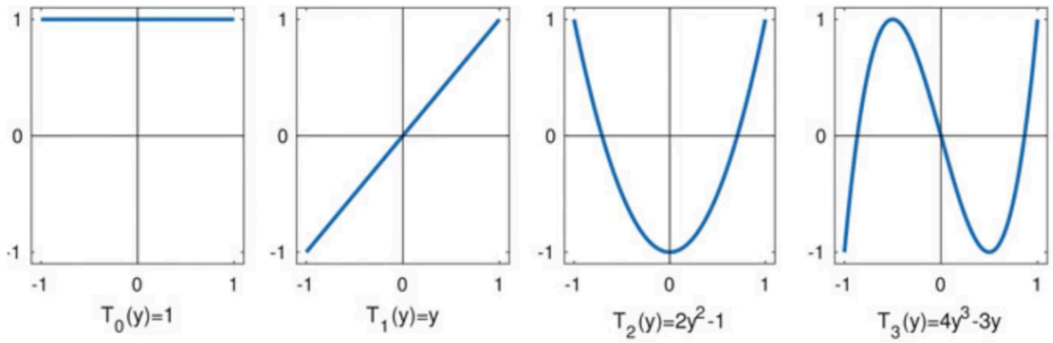
该模型需要解决复杂度问题

✓ Solution to Problem 3:

➢Use a polynomial function that can be computed recursively from L

✓ Chebyshev polynomial

➢$T_0(x) = 1, T_1(x) = x, T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x), x \in [-1, 1]$



切比雪夫多项式

$$g_\theta(\Lambda) = \sum_{k=0}^{K} \theta_k \Lambda^k \longrightarrow g_{\theta'}(\tilde{\Lambda}) = \sum_{k=0}^{K} \theta'_k T_k(\tilde{\Lambda})$$

这样的式子转变可以让计算变得更加简单

ChebNet

$$\bar{x}_0 = x, \bar{x}_1 = \tilde{L}x, \bar{x}_k = 2\tilde{L}\bar{x}_{k-1} - \bar{x}_{k-2}$$

$$y = g_{\theta'}(L)x = \sum_{k=0}^{K} \theta'_k T_k(\tilde{L})x$$

$$= \theta'_0 T_0(\tilde{L})x + \theta'_1 T_1(\tilde{L})x + \theta'_2 T_2(\tilde{L})x + \cdots + \theta'_K T_K(\tilde{L})x$$

$$= \theta'_0 \bar{x}_0 + \theta'_1 \bar{x}_1 + \theta'_2 \bar{x}_2 + \cdots + \theta'_K \bar{x}_K$$

$$= [\bar{x}_0 \ \ \bar{x}_1 \ \ \ldots \ \ \bar{x}_K][\theta'_0 \ \ \theta'_1 \ \ \ldots \theta'_K]$$

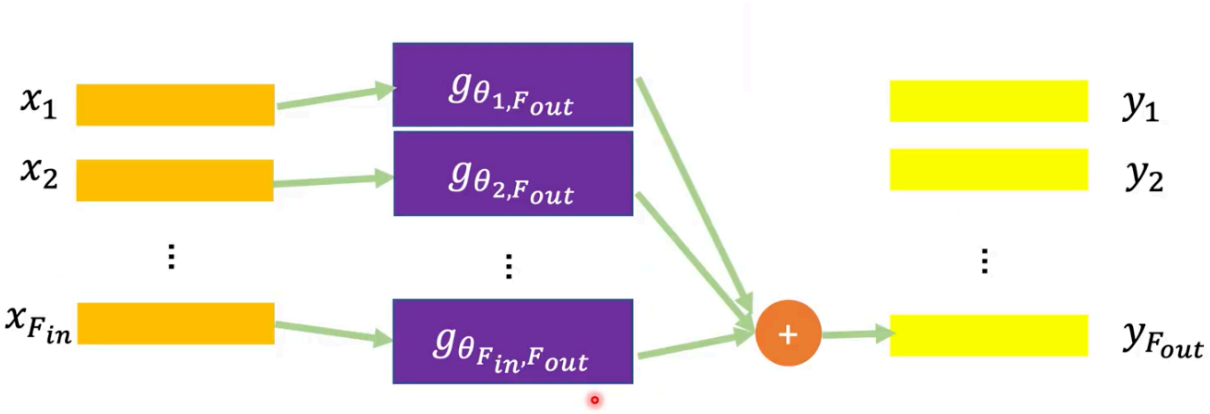
Calculating $\bar{x}_k = 2\tilde{L}\bar{x}_{k-1} - \bar{x}_{k-2}$ cost $O(E)$

Total complexity: $O(KE)$

降低了计算复杂度

先计算x，然后训练得到θ

可以有很多的channel：



GCN 图卷积神经网络 常用的模型

GCN

https://openreview.net/pdf?id=SJU4

✓ $y = g_{\theta'}(L)x = \sum_{k=0}^{K} \theta'_k T_k(\tilde{L})x \ , K = 1$

$$y = g_{\theta'}(L)x = \theta'_0 x + \theta'_1 \tilde{L} x \qquad \because \tilde{L} = \frac{2L}{\lambda_{max}} - I$$

$$= \theta'_0 x + \theta'_1 (\frac{2L}{\lambda_{max}} - I)x \qquad \because \lambda_{max} \approx 2$$

$$= \theta'_0 x + \theta'_1 (L - I)x \qquad \because L = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$$

$$= \theta'_0 x - \theta'_1 (D^{-\frac{1}{2}} A D^{-\frac{1}{2}})x \qquad \because \theta = \theta'_0 = -\theta'_1$$

$$= \theta(I + D^{-\frac{1}{2}} A D^{-\frac{1}{2}})x$$

$$\textit{renormalization trick: } I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$$

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

$$h_v = f\left(\frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} W x_u + b\right), \quad \forall v \in \mathcal{V}.$$

相加所有的neighbor 取平均 加bias 经过一个激活函数 得到h（hidden layer）

DropEdge

- ✓ Randomly drop the edge of input graph with probability p
  - ➢每一層都用同一個 drop 過的 adjacency matrix
  - ➢每一層都用不同的 drop 過的 adjacency matrix
- ✓ 可以避免 over-smoothing 的問題

Summary and Take Home Notes

- ✓ GAT and GCN are the most popular GNNs
- ✓ Although GCN is mathematically driven, we tend to ignore its math
- ✓ GNN (or GCN) suffers from information lose while getting deeper
- ✓ Many deep learning models can be slightly modified and designed to fit graph data, such as Deep Graph InfoMax, Graph Transformer, GraphBert.
- ✓ Theoretical analysis must be dealt with in the future
- ✓ GNN can be applied to a variety of tasks

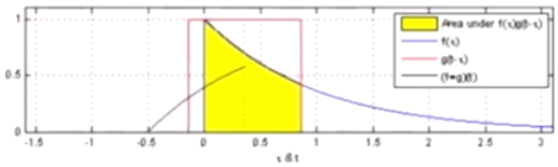
沈华伟 图卷积神经网络

- 卷积神经网络 卷积核的平移不变性可以学习到与位置无关的特征信息

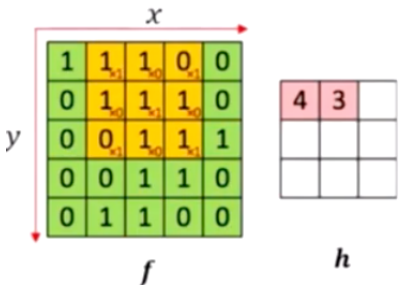
也就是将卷积神经网络从欧式空间的数据迁移到非欧式空间的数据（例如图）中

主要任务是定义图上的卷积和定义图上的pooling

- 连续数据的卷积



- 离散数据的卷积



共有两种方法：空间方法（Spatial Methods）、谱方法（Spectral Methods）