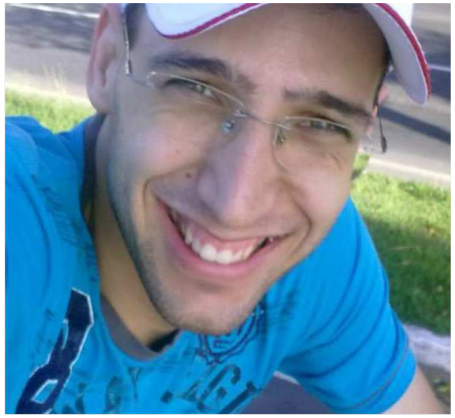
Workshop on Data Science - Emap/FGV

Spatio-Temporal Data Analytics via Graph Signal Processing

Prof. Luis Gustavo Nonato

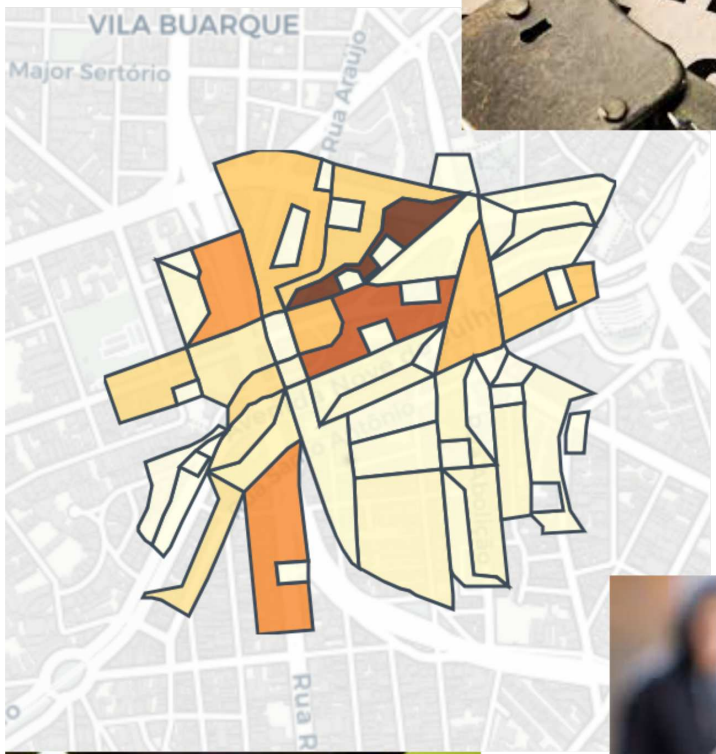
University of São Paulo - Brazil

Joint Work With

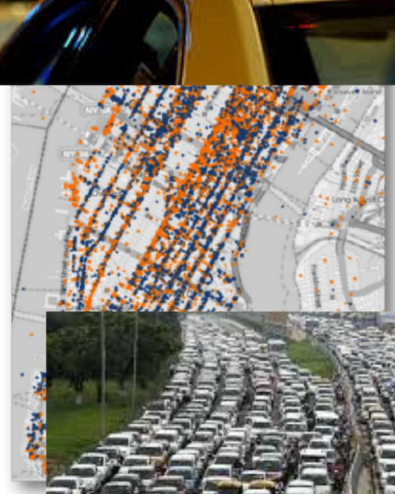


What is spatio-temporal data and why to understand it?

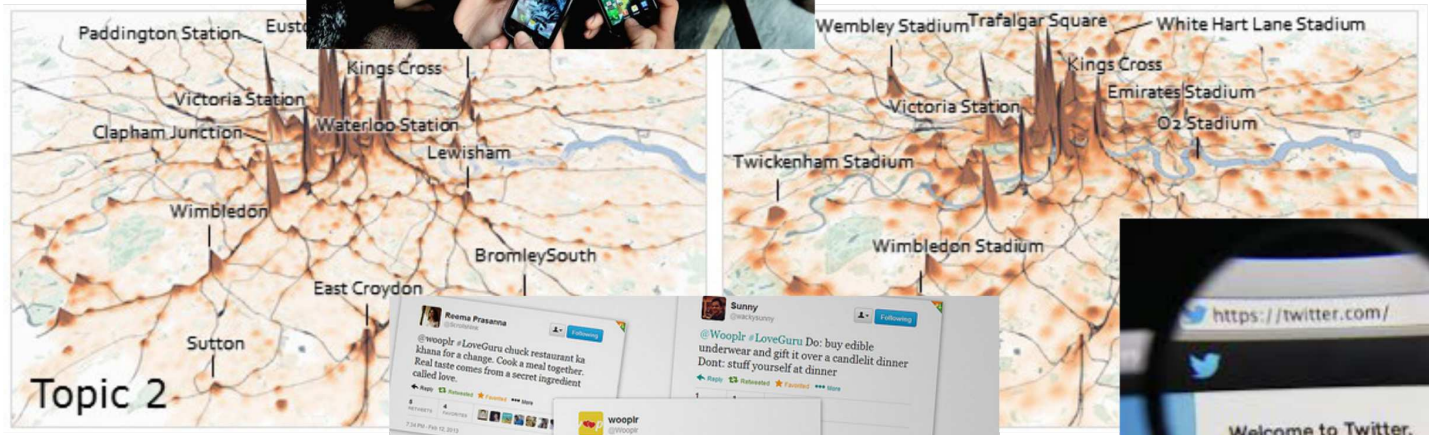
Spatio-Temporal Data Analysis



Spatio-Temporal Data Analysis

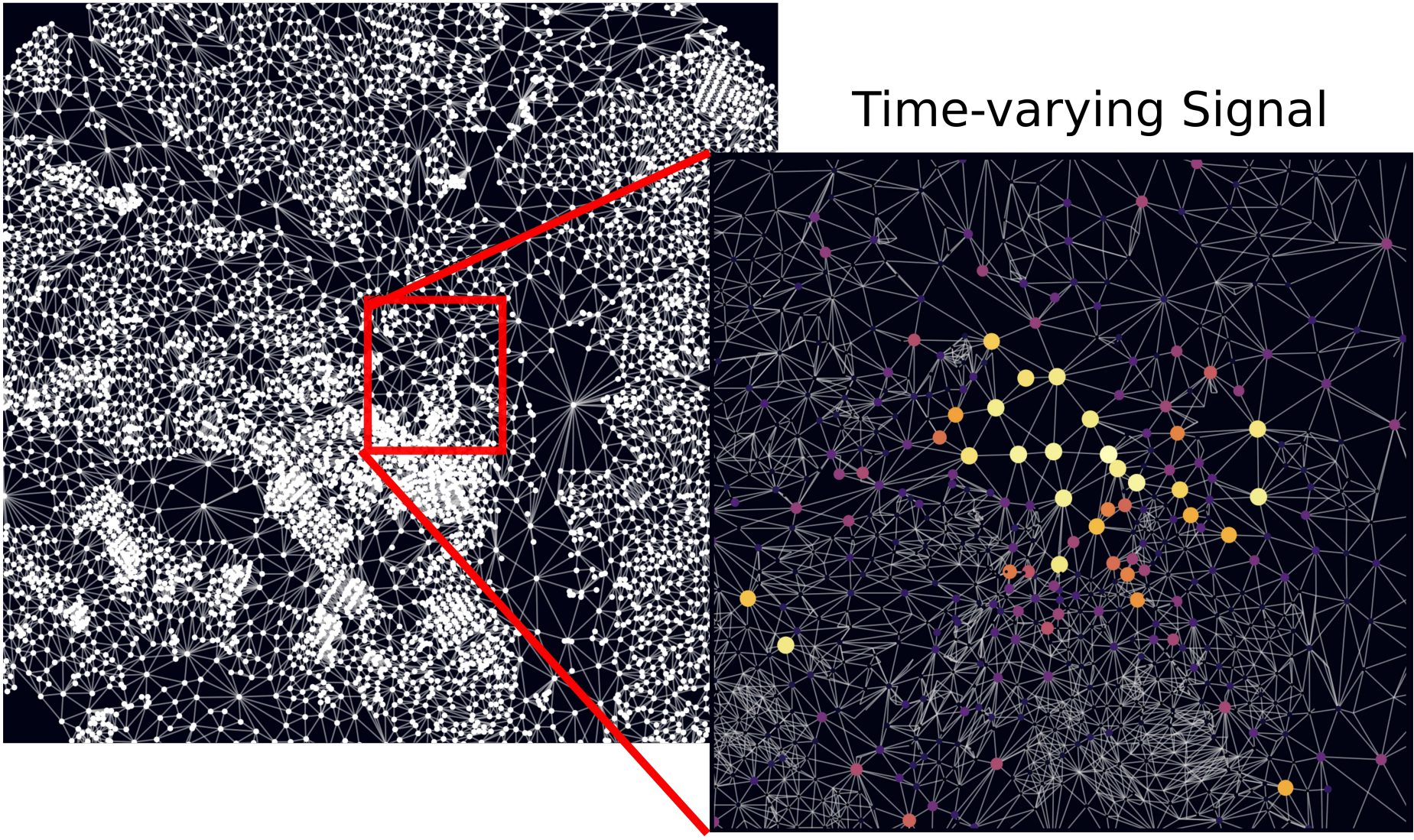


Spatio-Temporal Data Analysis



Graph

Time-varying Signal



Graph Signals Processing

How to analyze signals defined on graphs so as to uncover phenomena and identify patterns hidden in the data?

Graph Signals Processing

How to analyze signals defined on graphs so as to uncover phenomena and identify patterns hidden in the data?

Graph Signal Processing (GSP) has emerged as a main alternative in this context.

Graph Signals Processing

How to analyze signals defined on graphs so as to uncover phenomena and identify patterns hidden in the data?

Graph Signal Processing (GSP) has emerged as a main alternative in this context.

The main idea of GSP is to adapt well stablished signal processing tools such as Linear Filtering, Fourier, and Wavelet transforms to the context of graphs.

Basic Concepts and Notation

The Graph Laplacian

The Graph Laplacian

A undirected weighted graph $G = (V, E, W)$ is defined by a set of nodes $V = \{\tau_i\}$, edges $E = \{e_{ij}\}$ linking pairs of nodes τ_i and τ_j, and a symmetric weight matrix W whose entries satisfies

$$\begin{cases} w_{ij} > 0 & \text{if } \exists\ e_{ij} \in E \\ w_{ij} = 0 & \text{otherwise} \end{cases}$$

w_{ij} indicates how strongly the nodes τ_i and τ_j are linked.

A undirected weighted graph $G = (V, E, W)$ is defined by a set of nodes $V = \{\tau_i\}$, edges $E = \{e_{ij}\}$ linking pairs of nodes τ_i and τ_j, and a symmetric weight matrix W whose entries satisfies

$$\begin{cases} w_{ij} > 0 & \text{if } \exists\ e_{ij} \in E \\ w_{ij} = 0 & \text{otherwise} \end{cases}$$

w_{ij} indicates how strongly the nodes τ_i and τ_j are linked.

Graph Laplacian

$$L = D - W$$

where D is a diagonal matrix with entries $d_{ii} = \sum_j w_{ij}$.

The Graph Laplacian

A undirected weighted graph $G = (V, E, W)$ is defined by a set of nodes $V = \{\tau_i\}$, edges $E = \{e_{ij}\}$ linking pairs of nodes τ_i and τ_j, and a symmetric weight matrix W whose entries satisfies

$$\begin{cases} w_{ij} > 0 & \text{if } \exists\ e_{ij} \in E \\ w_{ij} = 0 & \text{otherwise} \end{cases}$$

w_{ij} indicates how strongly the nodes τ_i and τ_j are linked.

Graph Laplacian

$$L = D - W$$

where D is a diagonal matrix with entries $d_{ii} = \sum_j w_{ij}$.

L is a symmetric positive semidefinite matrix.

Spectral Decomposition

Eigenvectors of L

Eigenvalues of L

$$\underbrace{\begin{bmatrix} \vdots & \vdots & & \vdots \\ u_1 & u_2 & \cdots & u_n \\ \vdots & \vdots & & \vdots \end{bmatrix}}_{U} \qquad \underbrace{\begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}}_{\Lambda}$$

$$L = U \Lambda U^\top$$

Spectral Decomposition

Eigenvectors of L Eigenvalues of L

$$\underbrace{\begin{bmatrix} \vdots & \vdots & & \vdots \\ u_1 & u_2 & \cdots & u_n \\ \vdots & \vdots & & \vdots \end{bmatrix}}_{U} \quad \underbrace{\begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}}_{\Lambda}$$

$$L = U\Lambda U^\top$$

$$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$$

Spectral Decomposition

Eigenvectors of L Eigenvalues of L

$$\underbrace{\begin{bmatrix} \vdots & \vdots & & \vdots \\ u_1 & u_2 & \cdots & u_n \\ \vdots & \vdots & & \vdots \end{bmatrix}}_{U} \quad \underbrace{\begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}}_{\Lambda}$$

$$L = U\Lambda U^\top$$

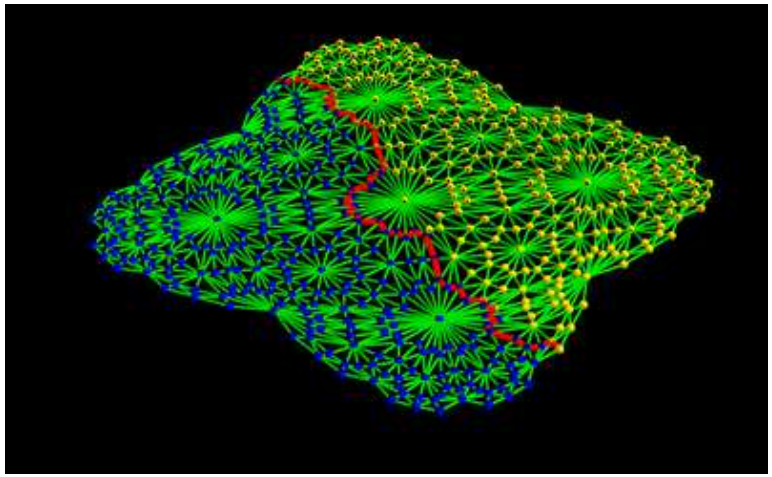
$$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$$

Eigenvectors are function defined on the vertices of G

$$u_i : V \to \mathbb{R}$$

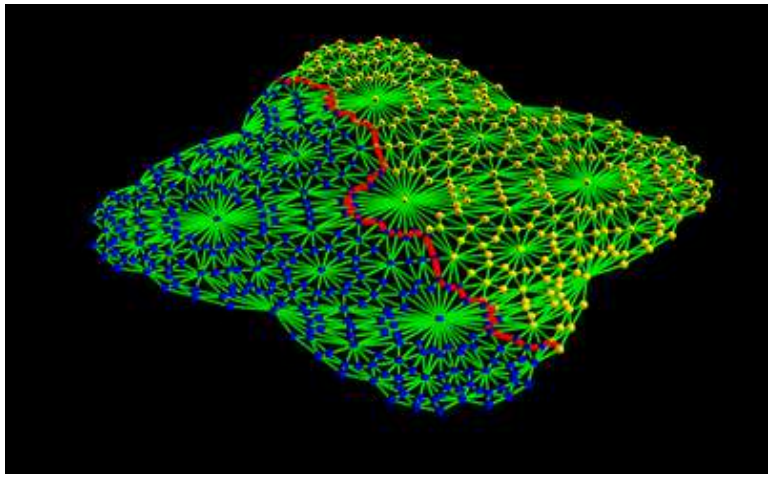
The *Discrete Courant's Nodal Theorem*

The *Discrete Courant's Nodal Theorem*

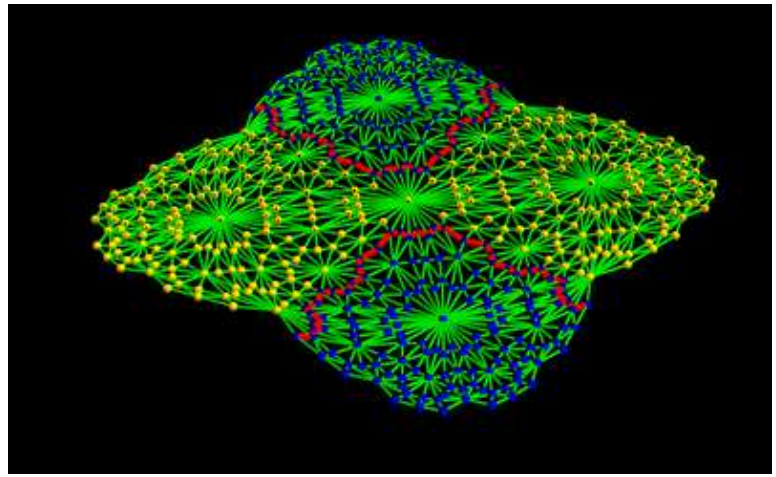


u_2

The *Discrete Courant's Nodal Theorem*

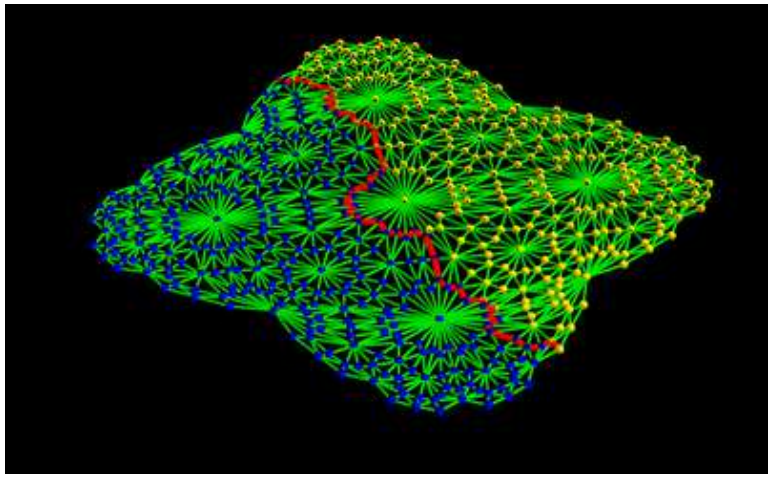


u_2

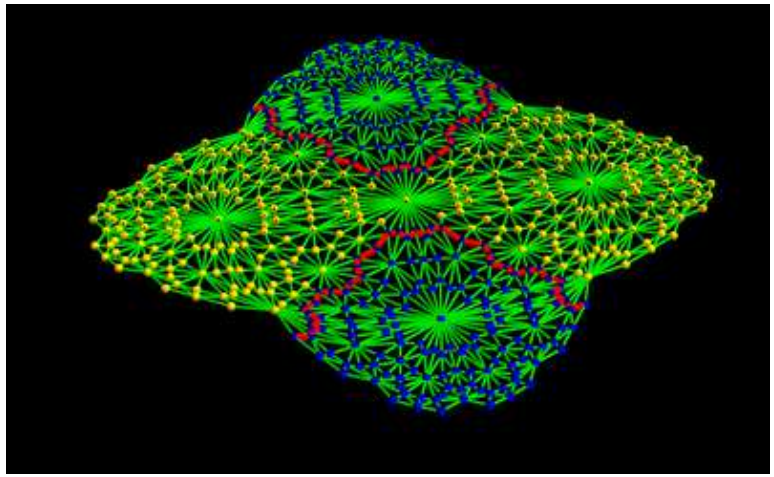


u_4

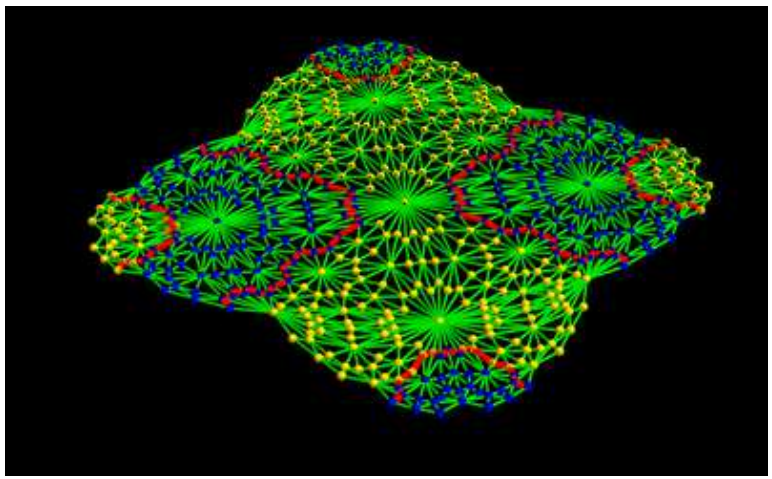
The *Discrete Courant's Nodal Theorem*



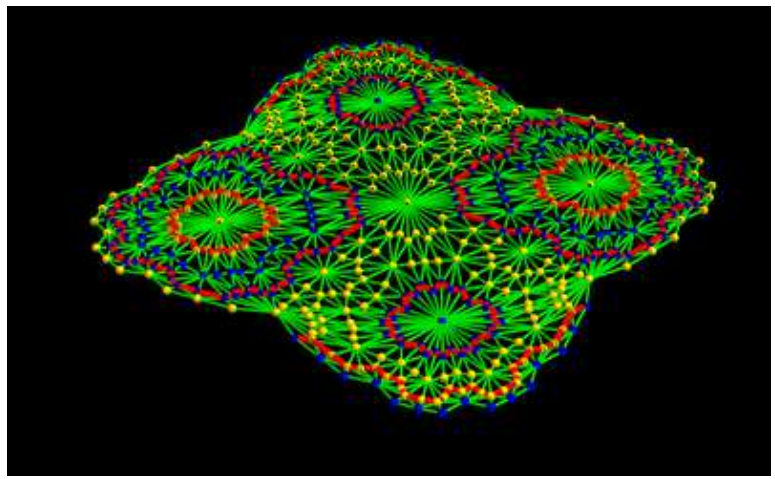
u_2



u_4



u_9



u_{11}

Graph Laplacian

Eigenvectors of the graph Laplacian behaves similarly to Fourier basis, tending to increase oscillation as the corresponding eigenvalue increases.

Eigenvectors of the graph Laplacian behaves similarly to Fourier basis, tending to increase oscillation as the corresponding eigenvalue increases.

$$\lambda_l \quad \longrightarrow \quad \text{frequencies}$$

Graph Laplacian

Eigenvectors of the graph Laplacian behaves similarly to Fourier basis, tending to increase oscillation as the corresponding eigenvalue increases.

$$\begin{aligned} \lambda_l &\longrightarrow \text{frequencies} \\ u_l &\longrightarrow \text{basis function} \end{aligned}$$

Graph Fourier Transform

Graph Fourier Transform

Graph Fourier Transform

Given a function $f : V \to \mathbb{R}$ defined on the vertices of G:

Graph Fourier Transform

Given a function $f : V \to \mathbb{R}$ defined on the vertices of G:

Graph Fourier Transform (GFT)

The Graph Fourier Transform of f is defined as

$$GFT[f](\lambda_l) = \widehat{f}(\lambda_l) = < f, u_l > = \sum_{i=1}^{n} f(i) u_l(i) \rightarrow \boxed{\widehat{f} = U^\top f}$$

Graph Fourier Transform

Given a function $f : V \to \mathbb{R}$ defined on the vertices of G:

Graph Fourier Transform (GFT)

The Graph Fourier Transform of f is defined as

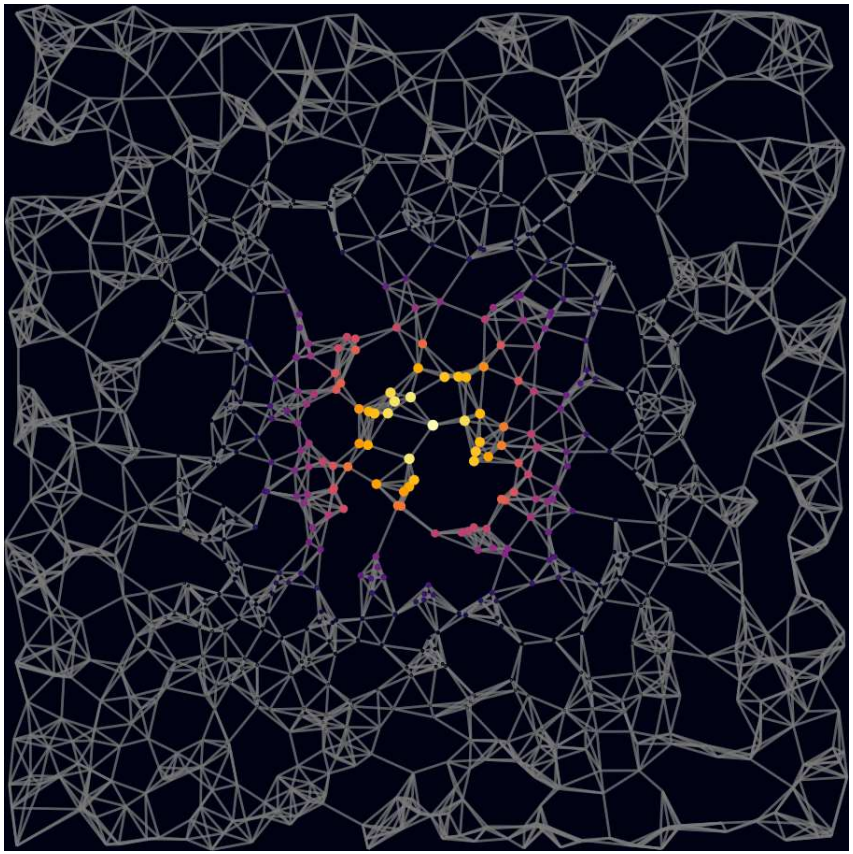
$$GFT[f](\lambda_l) = \widehat{f}(\lambda_l) = < f, u_l > = \sum_{i=1}^{n} f(i) u_l(i) \to \boxed{\widehat{f} = U^\top f}$$

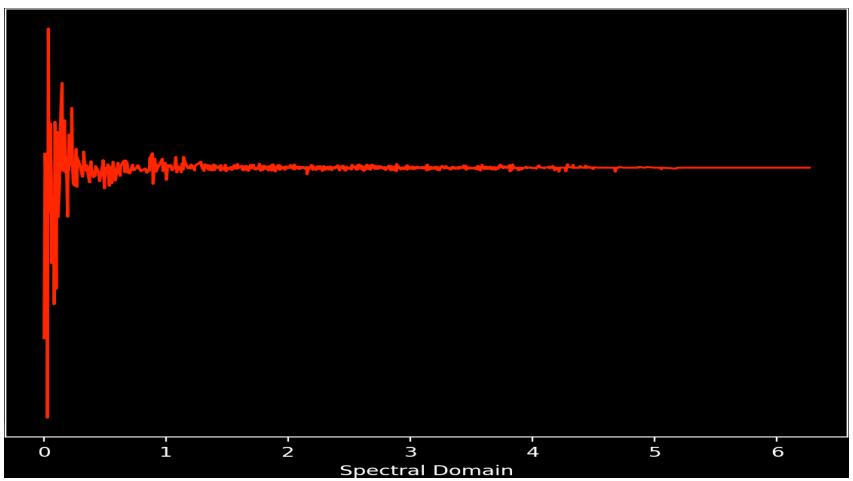
Inverse Graph Fourier Transform

The Inverse Graph Fourier Transform is given by

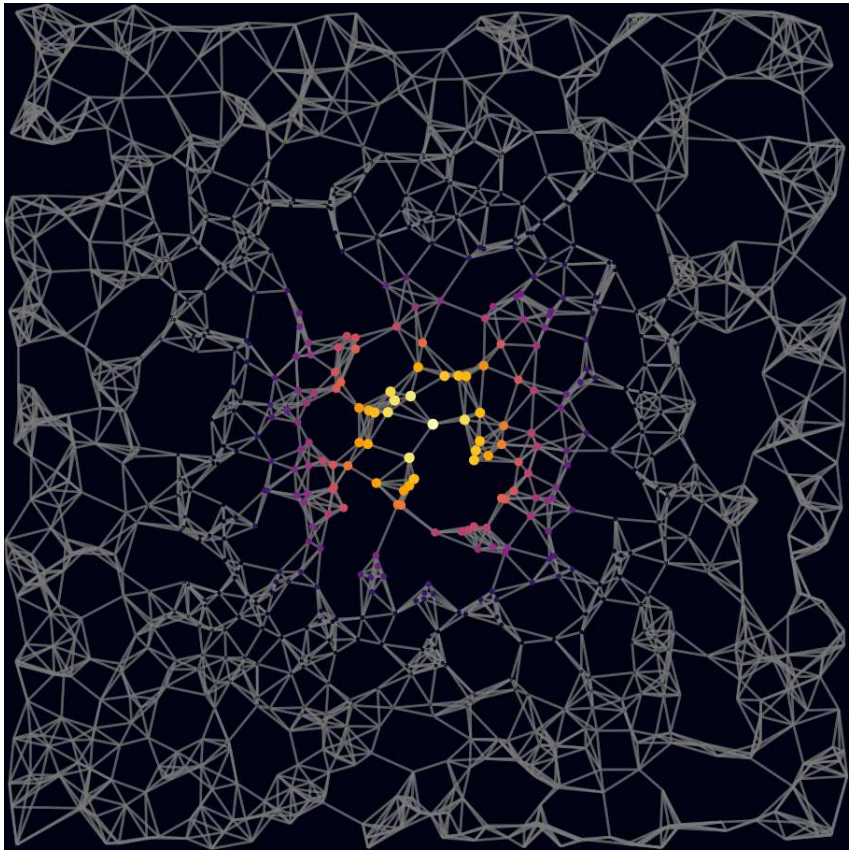
$$iGFT[\widehat{f}](i) = f(i) = \sum_{l=0}^{n-1} \widehat{f}(\lambda_l) u_l(i) \to \boxed{f = U\widehat{f}}$$

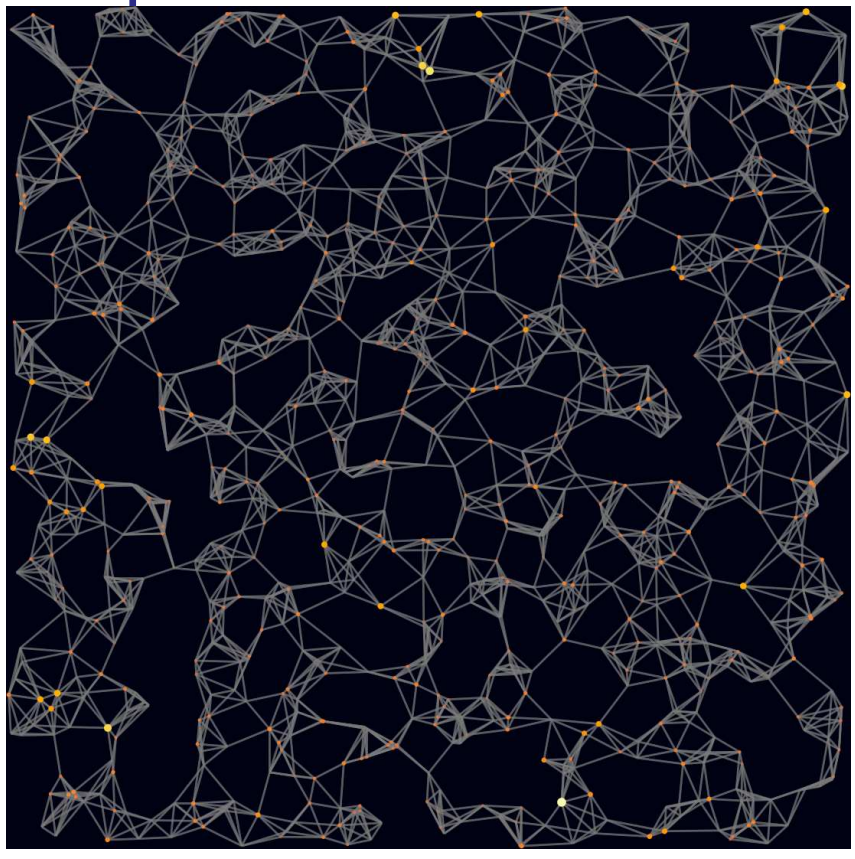
Graph Fourier Transform

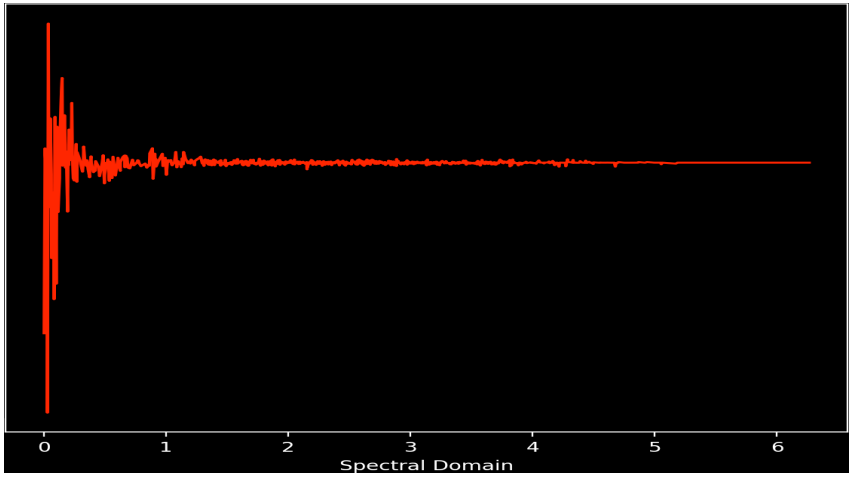


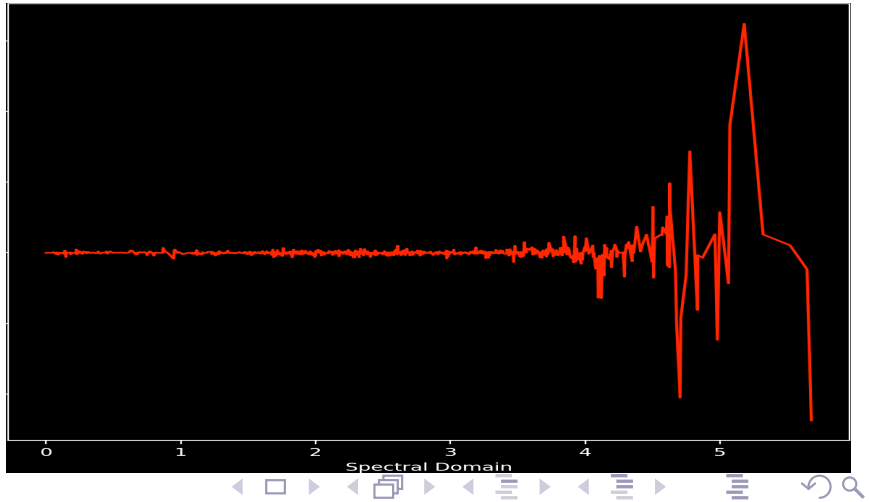


Graph Fourier Transform









Filtering Signals on Graphs

Classical Filtering and Convolution

Convolution

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

$$\mathcal{F}[f * g] = \widehat{f} \cdot \widehat{g}$$

Classical Filtering and Convolution

Convolution

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

$$\mathcal{F}[f * g] = \widehat{f} \cdot \widehat{g}$$

Linear Time Invariant Filter

If $\mathcal{L}$ is a linear time invariant filter then there exists a function h such that

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

Graph Convolution

The definition of convolution can not be directly applied to graphs because the shift operation $f(x-t)$ is not well defined on the graph domain.

Graph Convolution

The definition of convolution can not be directly applied to graphs because the shift operation $f(x-t)$ is not well defined on the graph domain.

However, the *GFT* allows to define convolution as:

Graph Convolution

The definition of convolution can not be directly applied to graphs because the shift operation $f(x-t)$ is not well defined on the graph domain.

However, the *GFT* allows to define convolution as:

Convolution on Graphs

$f, g : V \rightarrow \mathbb{R}$

$$(f * g) = iGFT[\widehat{f} \cdot \widehat{g}]$$

Graph Convolution

The definition of convolution can not be directly applied to graphs because the shift operation $f(x-t)$ is not well defined on the graph domain.

However, the *GFT* allows to define convolution as:

Convolution on Graphs

$f, g : V \rightarrow \mathbb{R}$

$$(f * g) = iGFT[\widehat{f} \cdot \widehat{g}]$$

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

Graph Spectral Filtering

$$\mathcal{L}[f] = iGFT[\widehat{f} \cdot \widehat{h}]$$

Graph Spectral Filtering

$$\mathcal{L}[f] = iGFT[\widehat{f} \cdot \widehat{h}]$$

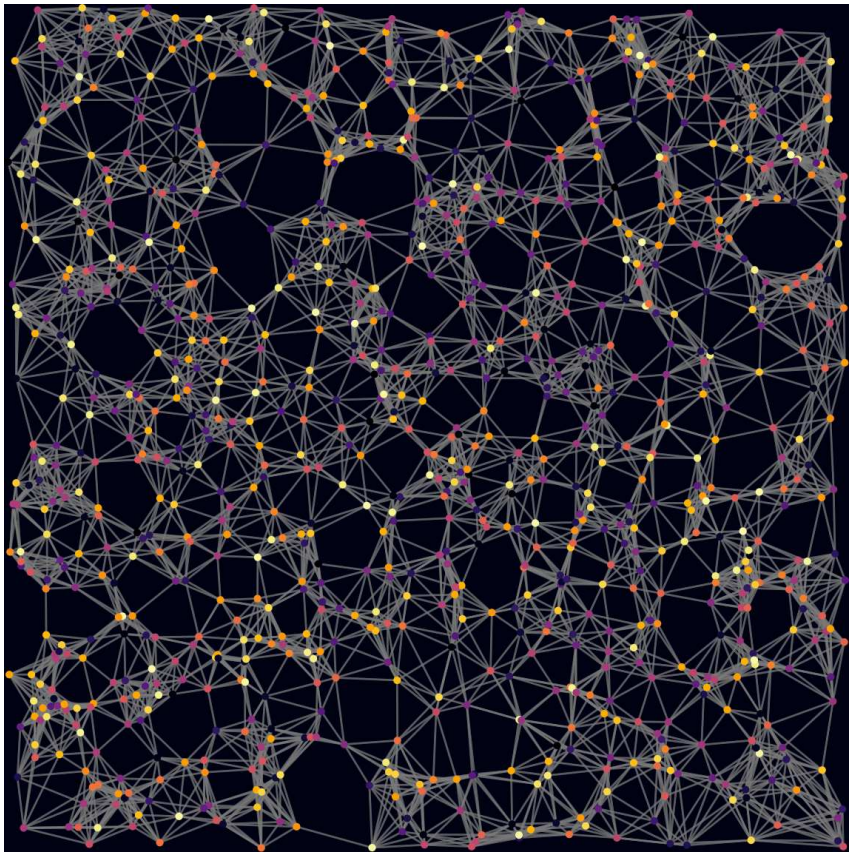
$$\mathcal{L}[f] = \widehat{h}(L)f$$

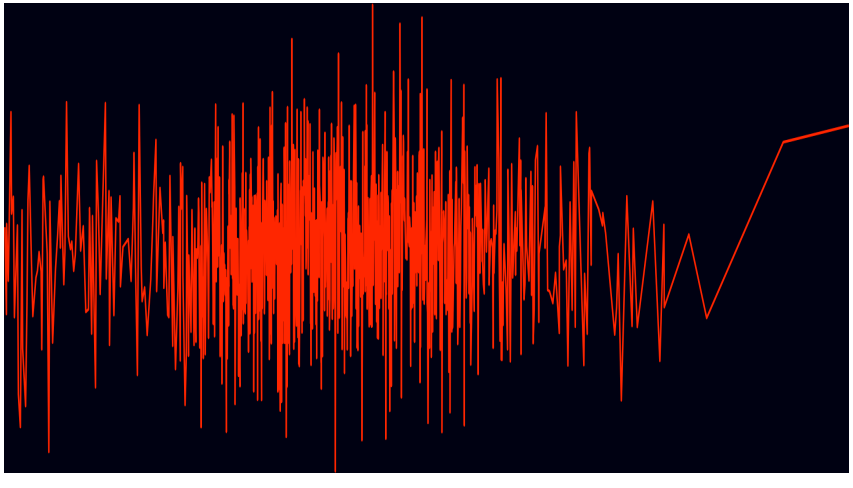
Graph Spectral Filtering

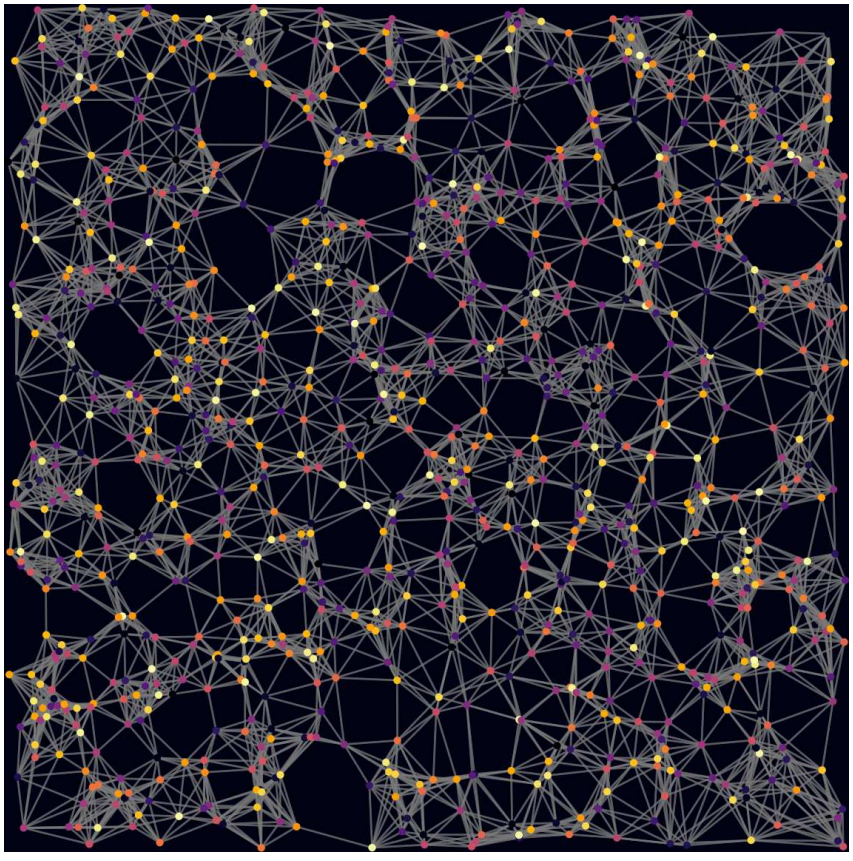
$$\mathcal{L}[f] = iGFT[\widehat{f} \cdot \widehat{h}]$$

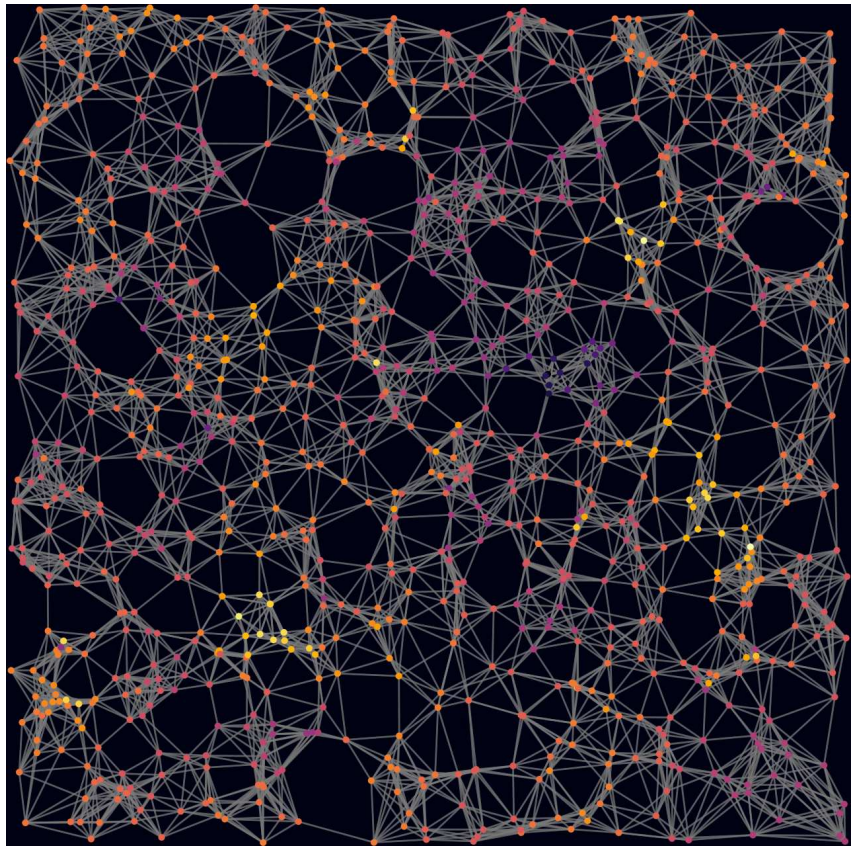
$$\mathcal{L}[f] = \widehat{h}(L)f$$

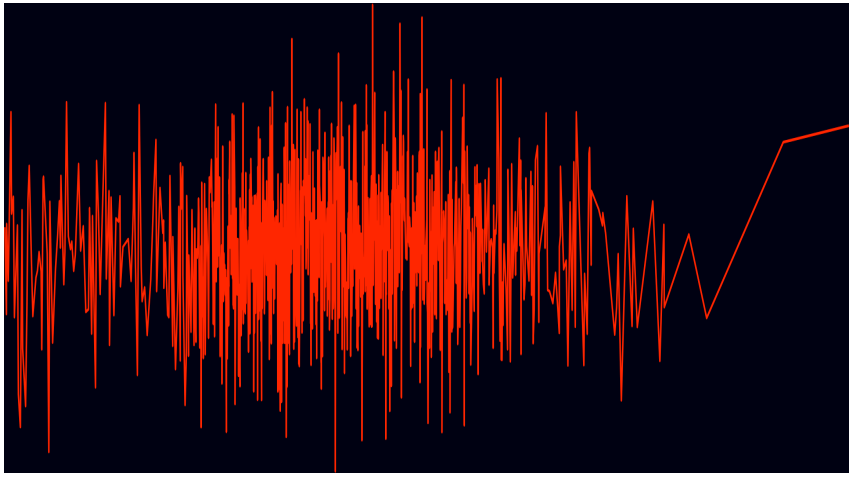
$$\mathcal{L}[f] = U \begin{bmatrix} \widehat{h}(\lambda_1) & & \\ & \ddots & \\ & & \hat{h}(\lambda_n) \end{bmatrix} U^\top f$$

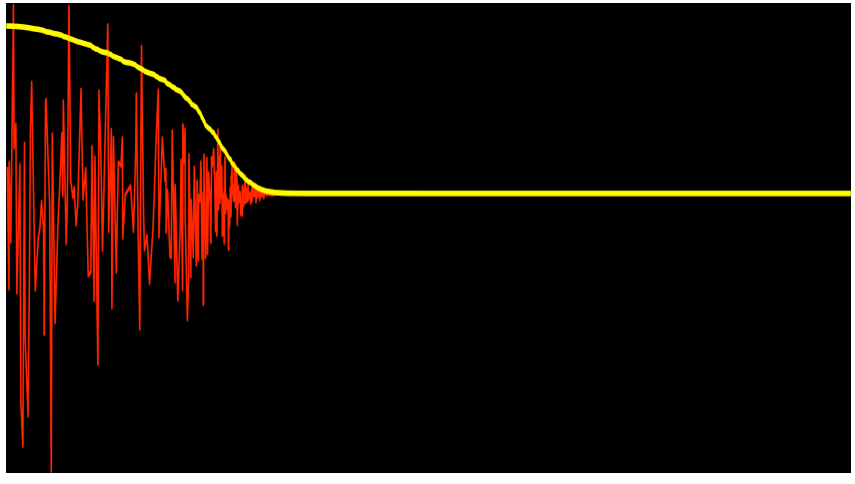












Filter Design

Graph Spectral Filtering: Smoothing

The design of filters with specific properties is a central issue.

Graph Spectral Filtering: Smoothing

The design of filters with specific properties is a central issue.

Smoothing a function f:
Find h such that $g = \mathcal{L}[f] = \widehat{h}(L)f$ minimizes

$$\arg\min_{h}\{\|f-g\|^2 + \gamma g^\top L g\}$$

Graph Spectral Filtering: Smoothing

The design of filters with specific properties is a central issue.

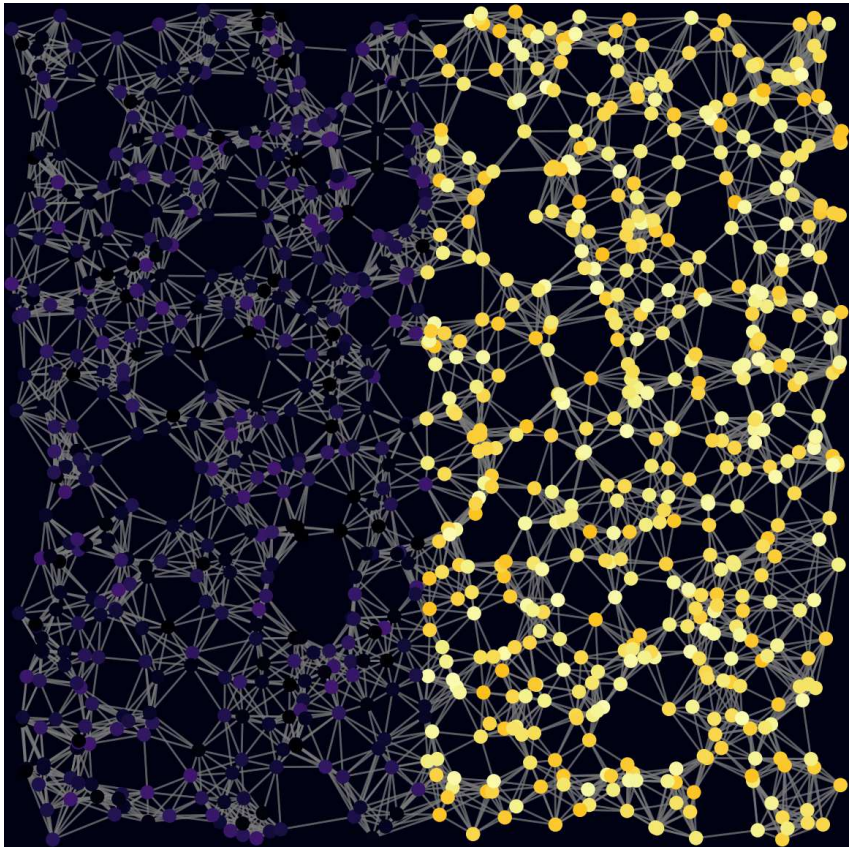
Smoothing a function f:
Find h such that $g = \mathcal{L}[f] = \widehat{h}(L)f$ minimizes

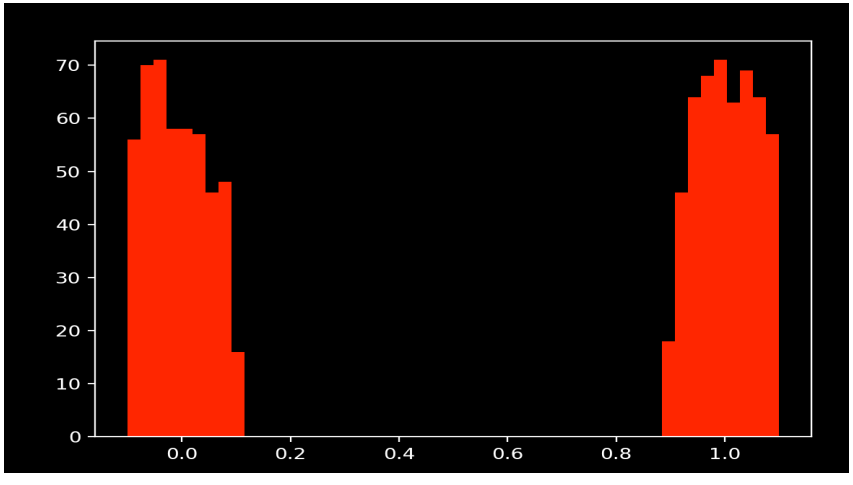
$$\arg\min_{h}\{\|f-g\|^2 + \gamma g^\top L g\}$$

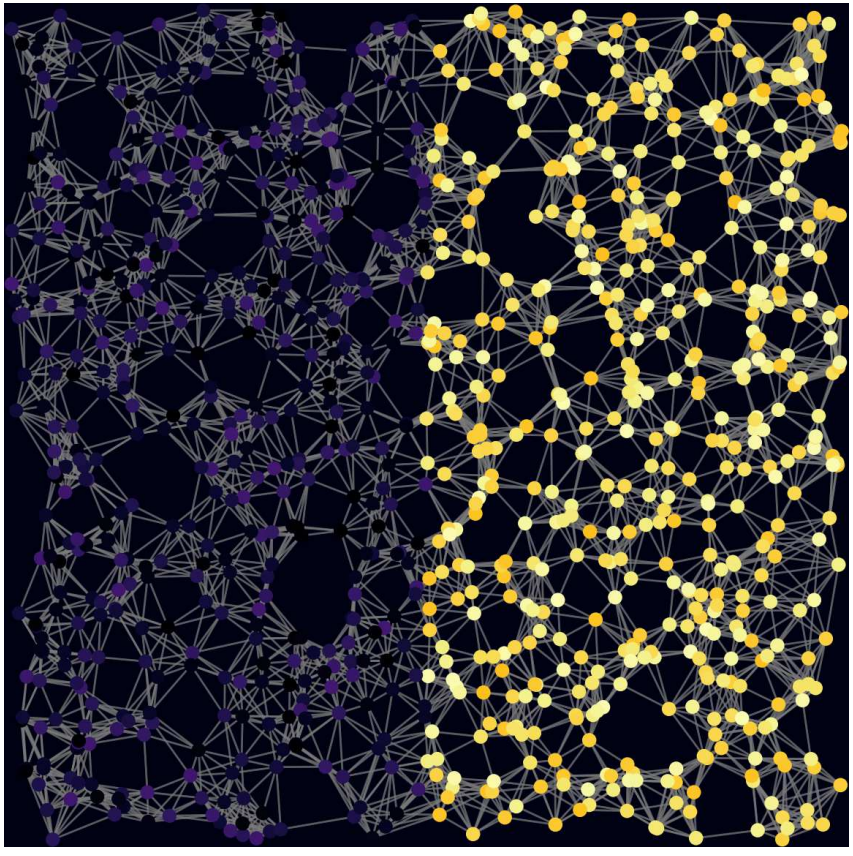
It can be shown that

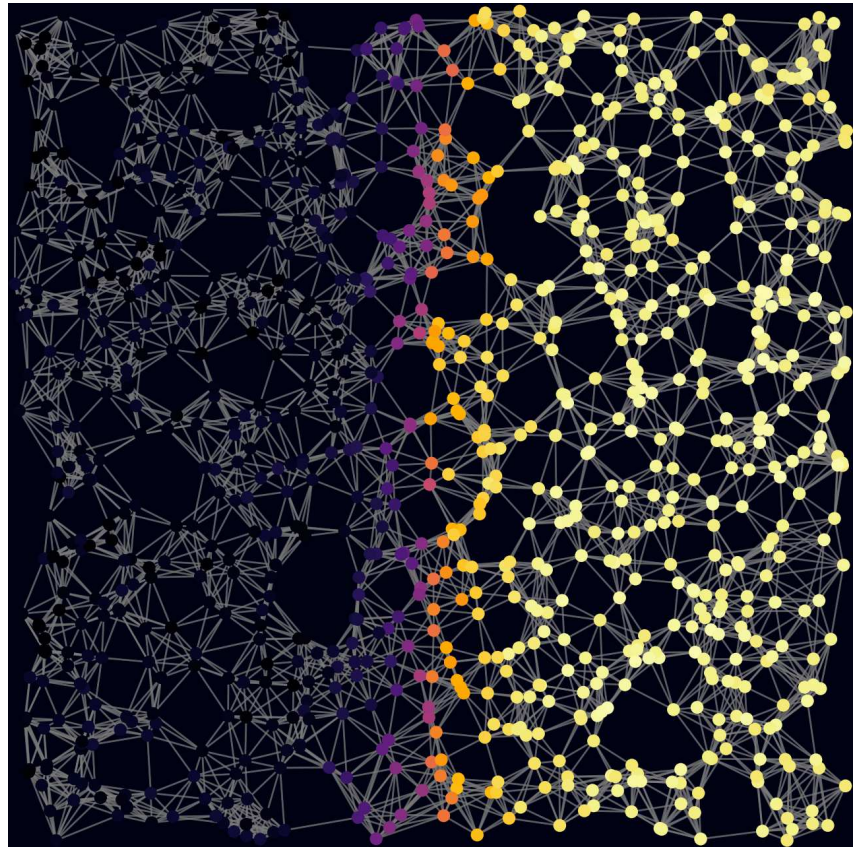
$$\widehat{h}(\lambda) = \frac{1}{1+\gamma\lambda}$$

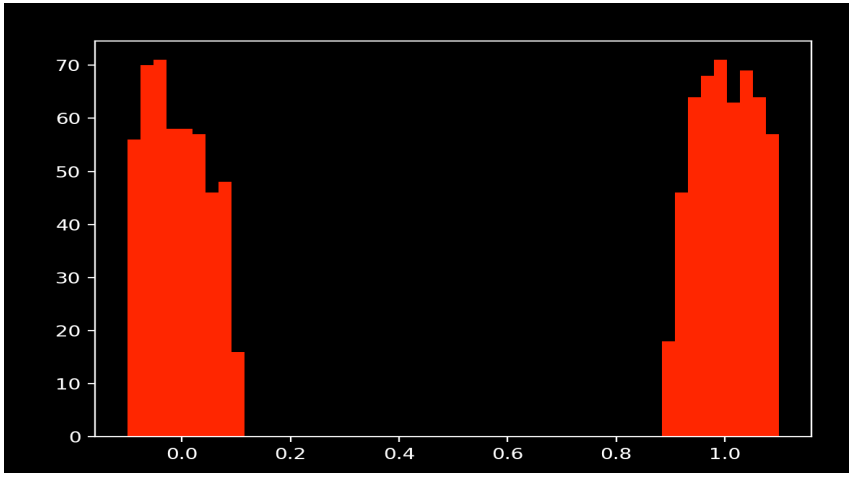
is the sought solution.

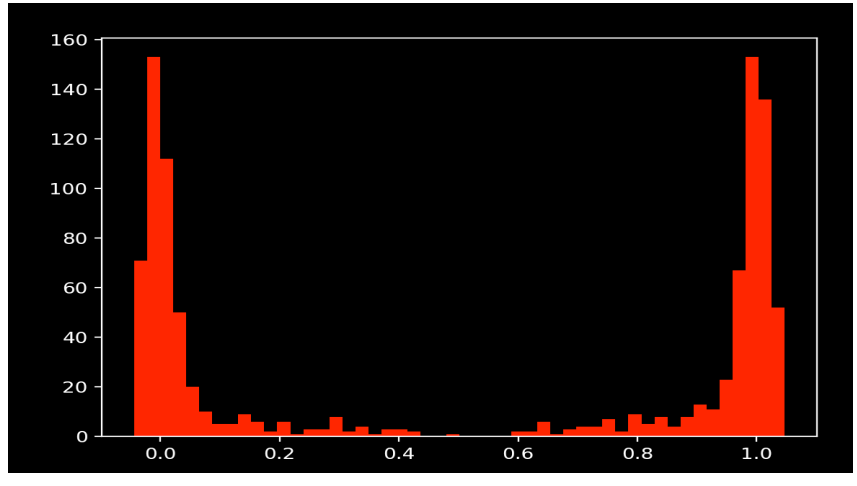



70
60
50
40
30
20
10
0
0.0
0.2
0.4
0.6
0.8
1.0






70
60
50
40
30
20
10
0
0.0
0.2
0.4
0.6
0.8
1.0

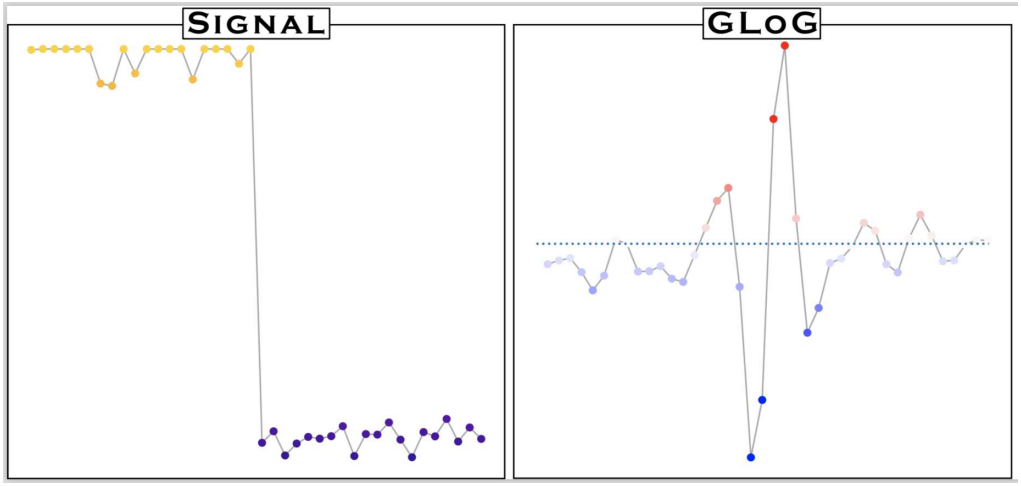

160
140
120
100
80
60
40
20
0
0.0
0.2
0.4
0.6
0.8
1.0

Graph Spectral Filtering: GLoG

Laplacian of Gaussian (LoG)

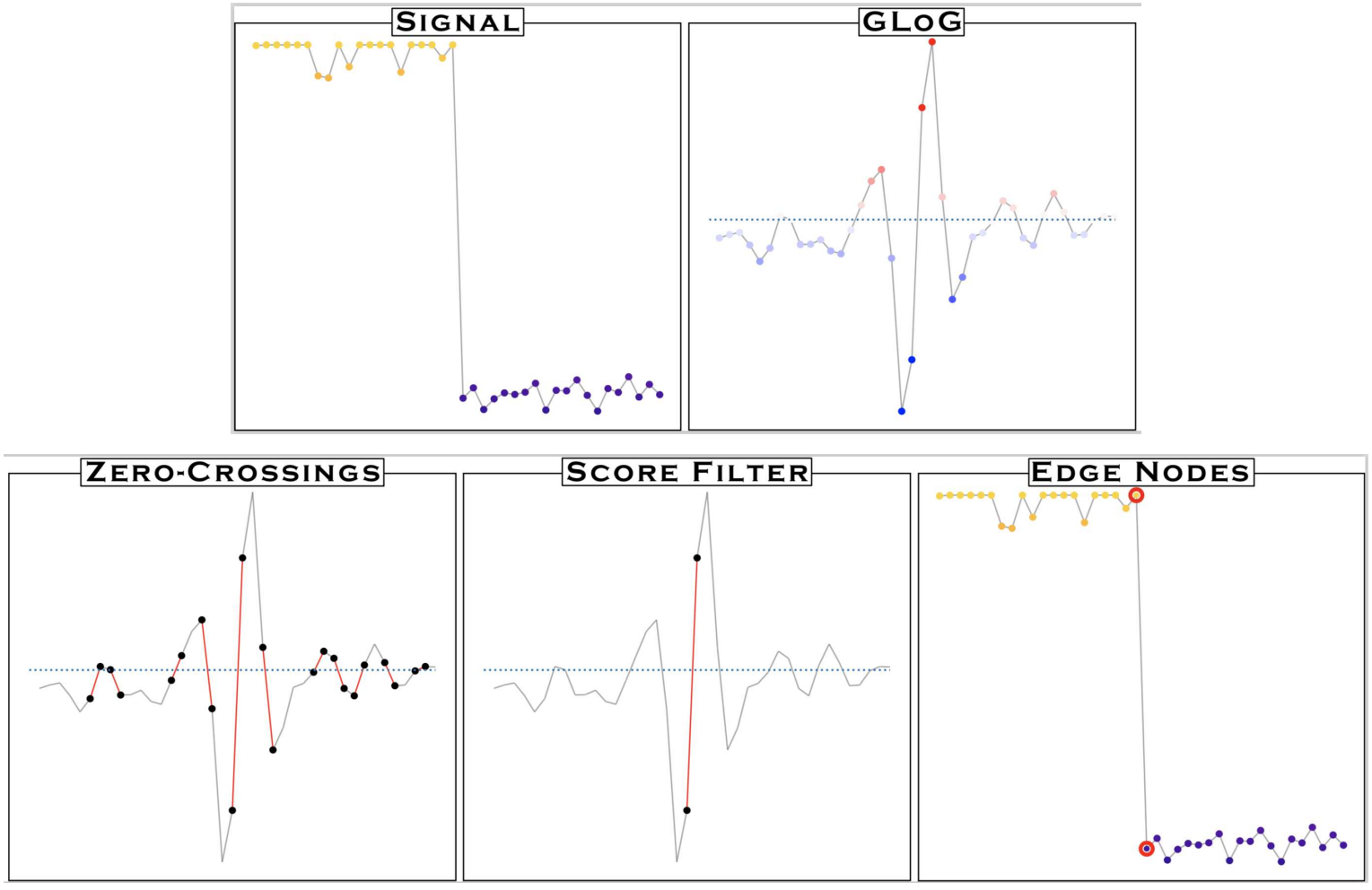
Graph Spectral Filtering: GLoG

Laplacian of Gaussian (LoG)



Graph Spectral Filtering: GLoG

Laplacian of Gaussian (LoG)



$$LoG(f) = \nabla^2[G * f], \quad G \text{ is a Gaussian function}$$

Graph Spectral Filtering: GLoG

Laplacian of Gaussian (LoG)

In the context of GSP, the goal is to define h such that

$$GLoG(f) = \widehat{h}(L)f$$

Graph Spectral Filtering: GLoG

Laplacian of Gaussian (LoG)

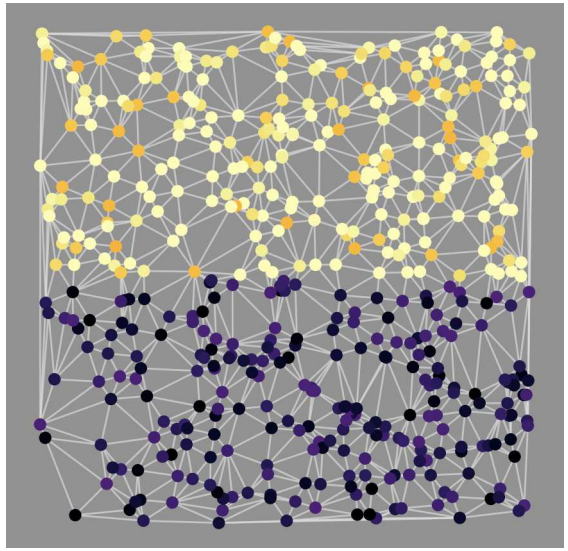
In the context of GSP, the goal is to define h such that

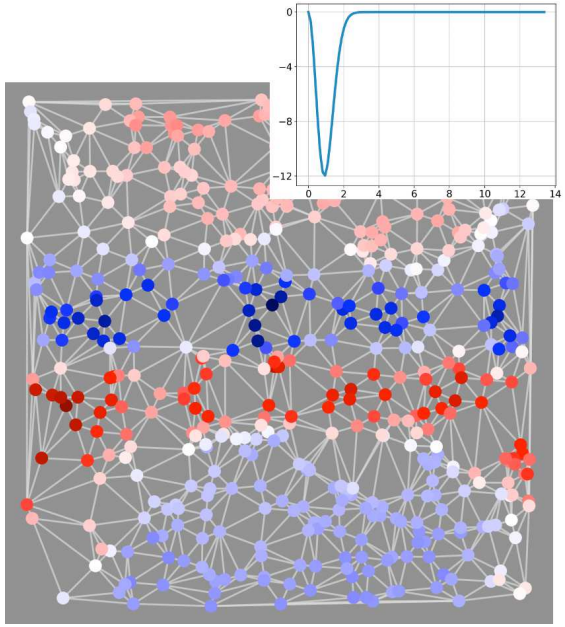
$$GLoG(f) = \widehat{h}(L)f$$

The solution is given by

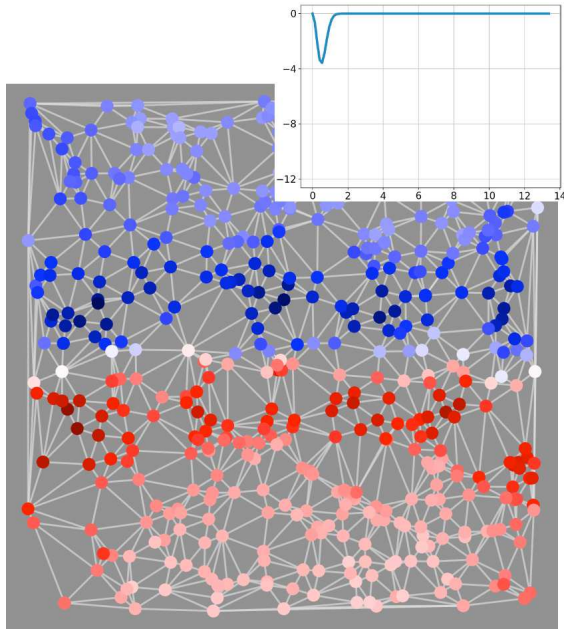
$$\widehat{h}(\lambda_l) = -4\pi^2\lambda_l^2 \exp(\sigma^2\lambda_l^2/2)$$

In the 2D case, the $GLoG$ is equivalent to the LoG filter.

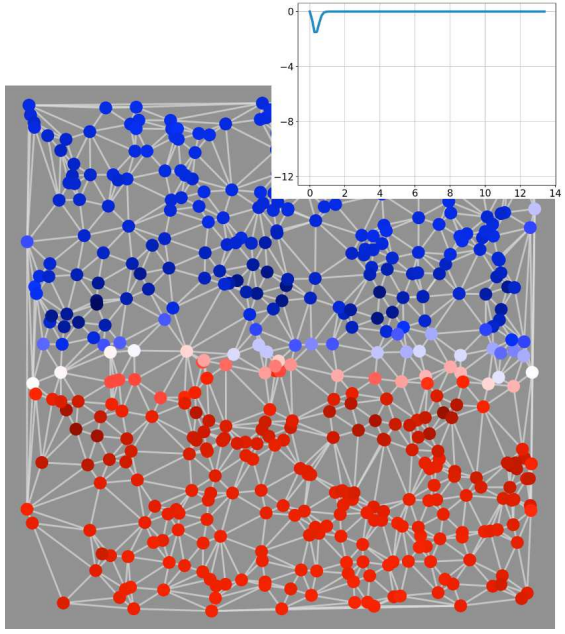




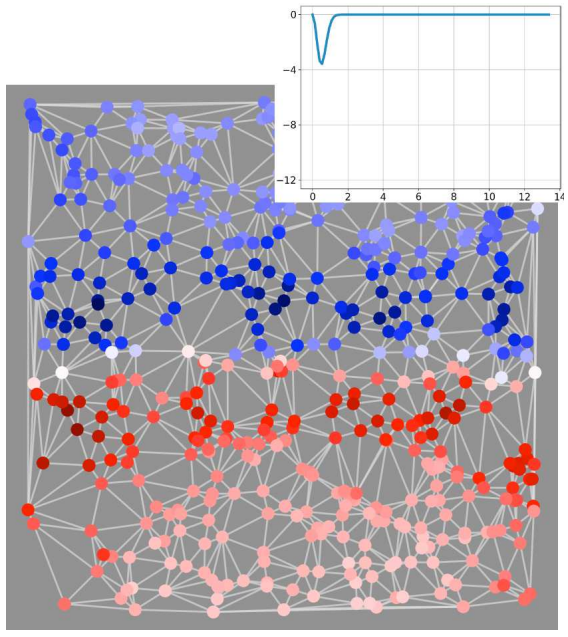
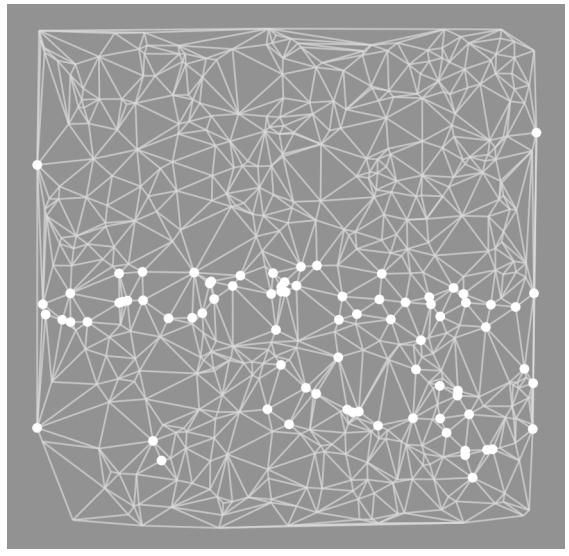
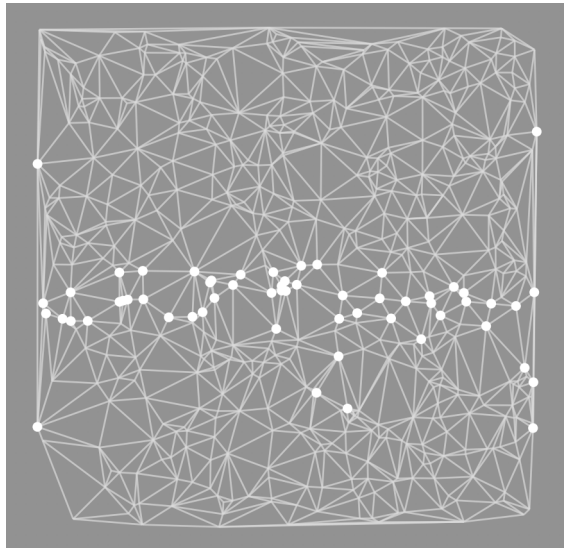
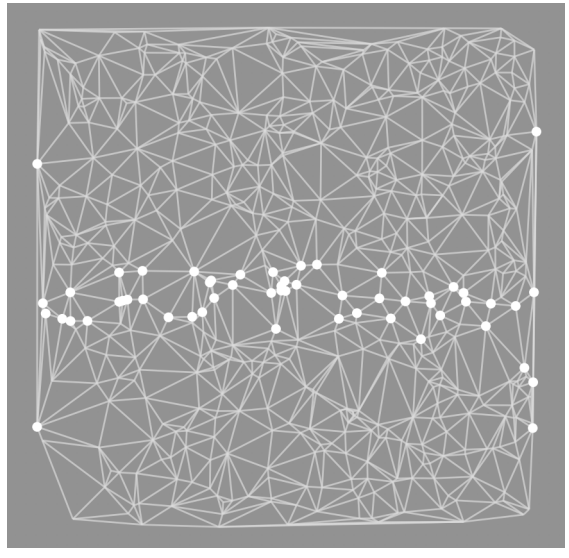
$\sigma = 1$



$\sigma = 2$

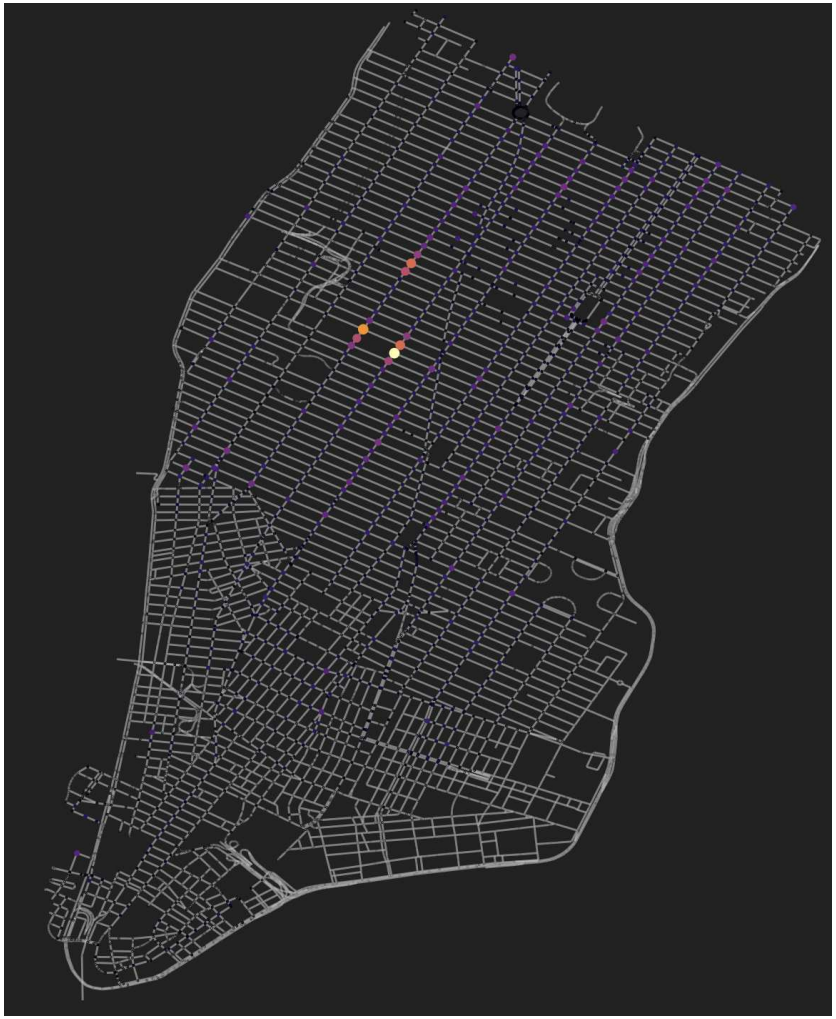


$\sigma = 3$


0
−4
−8
−12
0
2
4
6
8
10
12
14




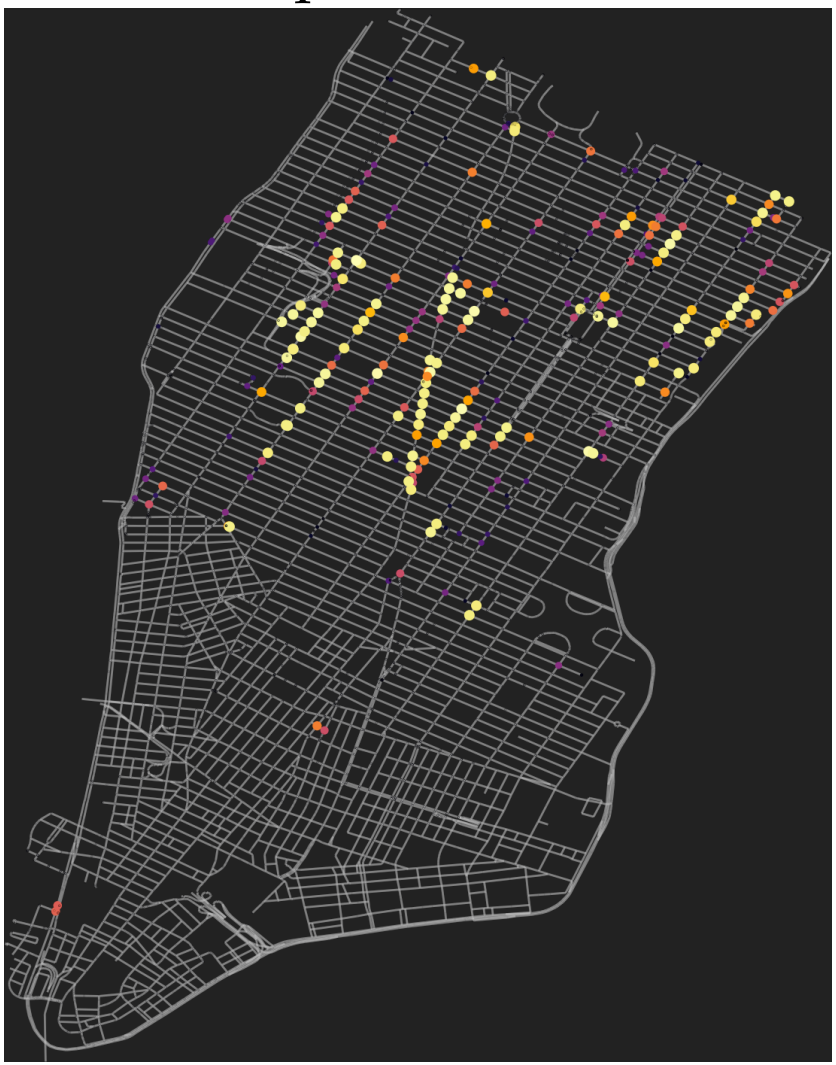
Visual Enhancement

Taxi Pick ups in Downtown Manhattan



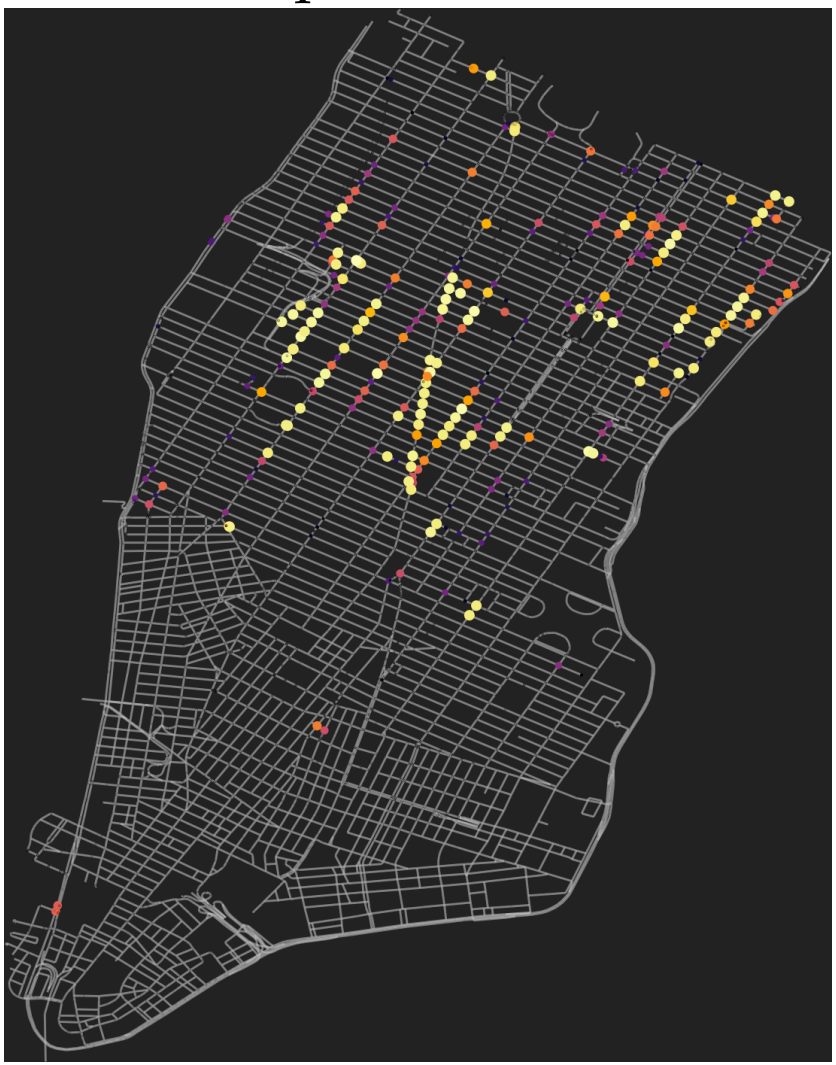
Visual Enhancement

Taxi Pick ups in Downtown Manhattan



Visual Enhancement

Taxi Pick ups in Downtown Manhattan



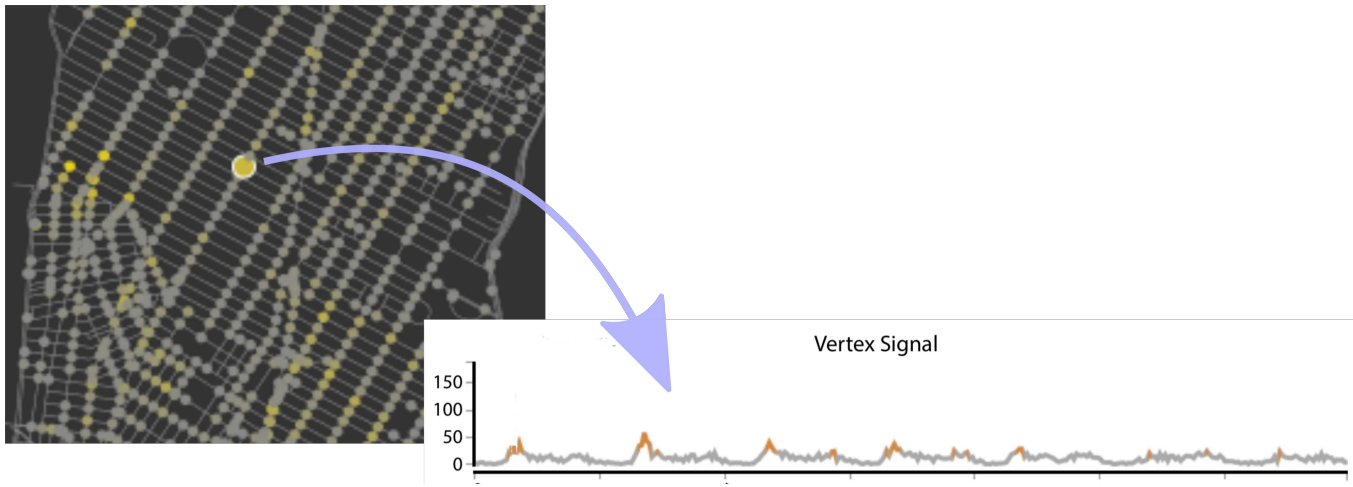
Spatio-Temporal Data Analysis via GLoG

Spatio-Temporal Data Analysis

$$f : V \times T \to \mathbb{R}, \qquad T = \{t_1, t_2, \ldots, t_m\}$$

Spatio-Temporal Data Analysis

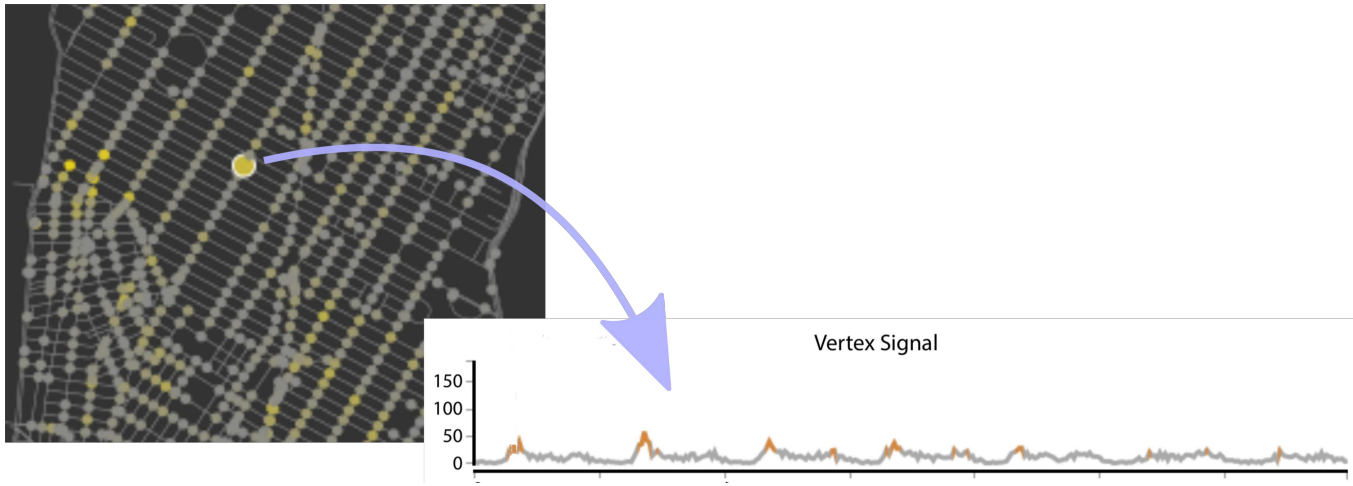
$$f : V \times T \to \mathbb{R}, \qquad T = \{t_1, t_2, \ldots, t_m\}$$



Edge nodes: adjacent nodes where GLoG changes sign

Spatio-Temporal Data Analysis

$$f : V \times T \to \mathbb{R}, \qquad T = \{t_1, t_2, \ldots, t_m\}$$



Edge nodes: adjacent nodes where GLoG changes sign
Probability of edge nodes

$$I(\tau_i, t_k) = \begin{cases} 1 & \text{if } \tau_i \text{ is an edge node in time slice } t_k \\ 0 & \text{otherwise} \end{cases}$$

$$p_e(\tau_i) = \frac{1}{m} \sum_{k=1}^{m} I(\tau_i, t_k)$$

Probability of an observed node configuration:

$$p(I(\tau_i, t_k)) = \begin{cases} p_e(\tau_i) & \text{if } I(\tau_i, t_k) = 1 \\ 1 - p_e(\tau_i) & \text{if } I(\tau_i, t_k) = 0 \end{cases}$$

Probability of an observed node configuration:

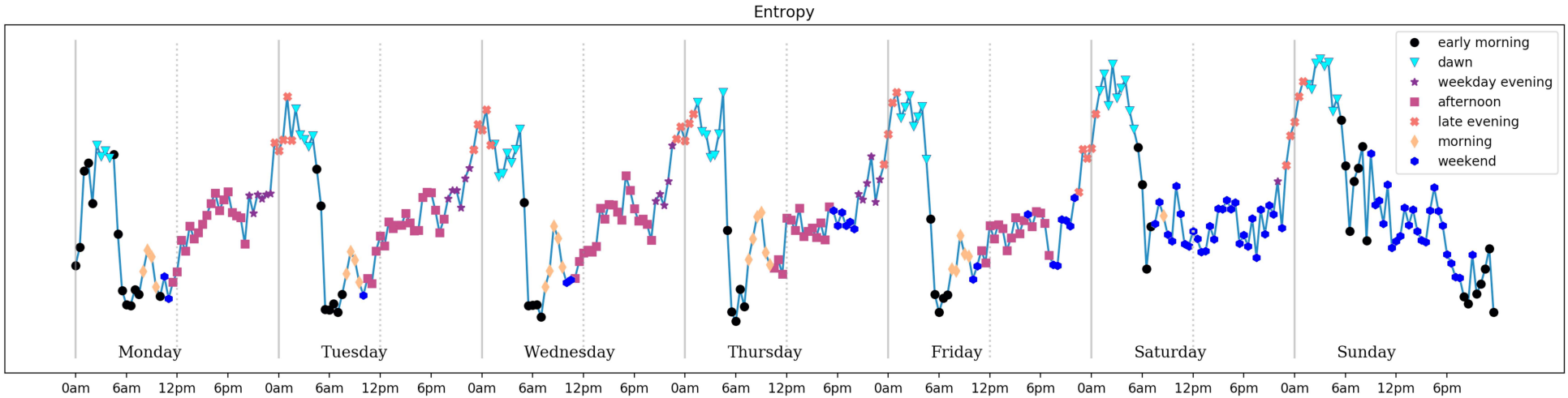
$$p(I(\tau_i, t_k)) = \begin{cases} p_e(\tau_i) & \text{if } I(\tau_i, t_k) = 1 \\ 1 - p_e(\tau_i) & \text{if } I(\tau_i, t_k) = 0 \end{cases}$$

Time Slice Entropy

$$E(t_k) = -\sum_{i=1}^{n} p(I(\tau_i, k)) \log p(I(\tau_i, k))$$

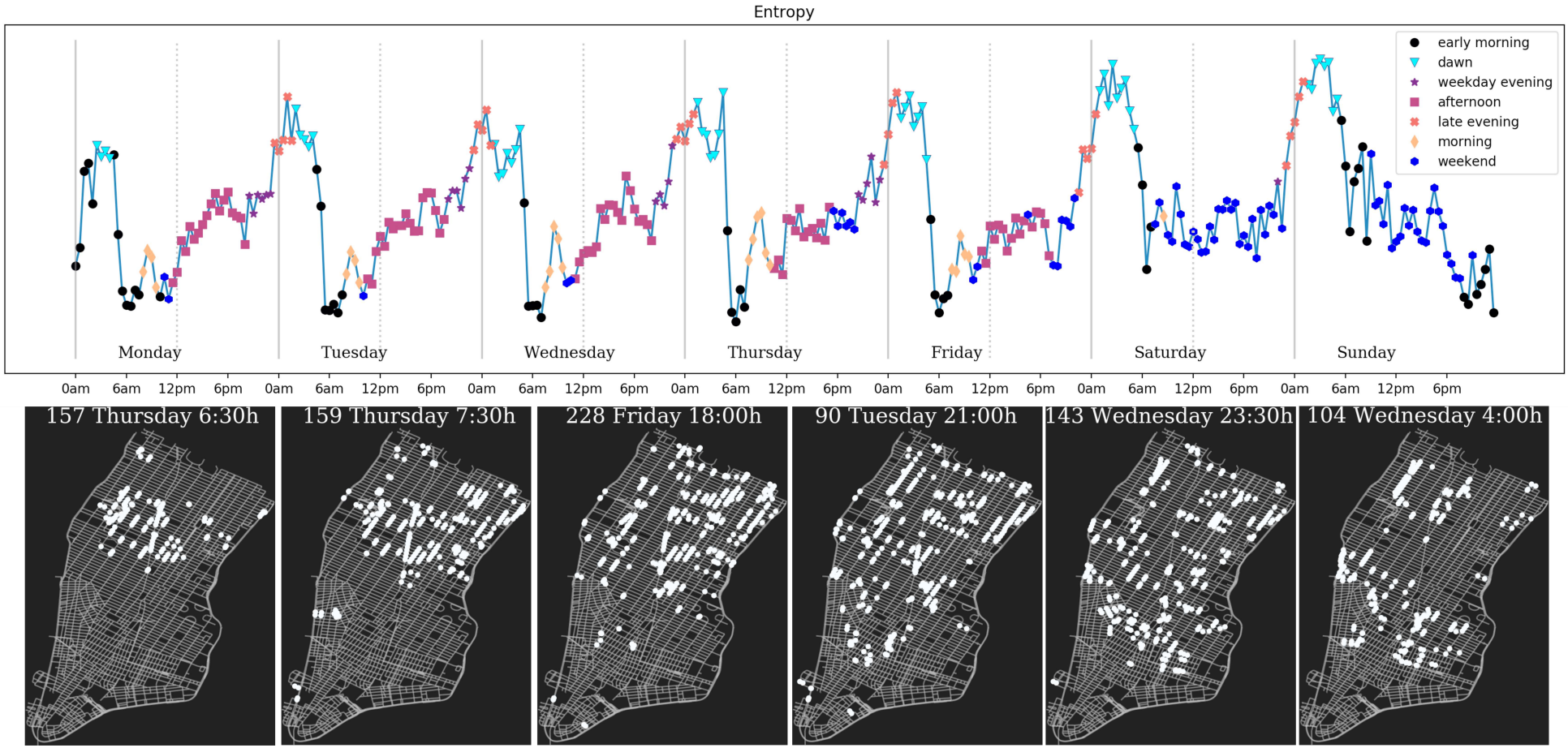
Spatio-Temporal Data Analysis

Analyzing Taxi Pick ups in Downtown Manhattan



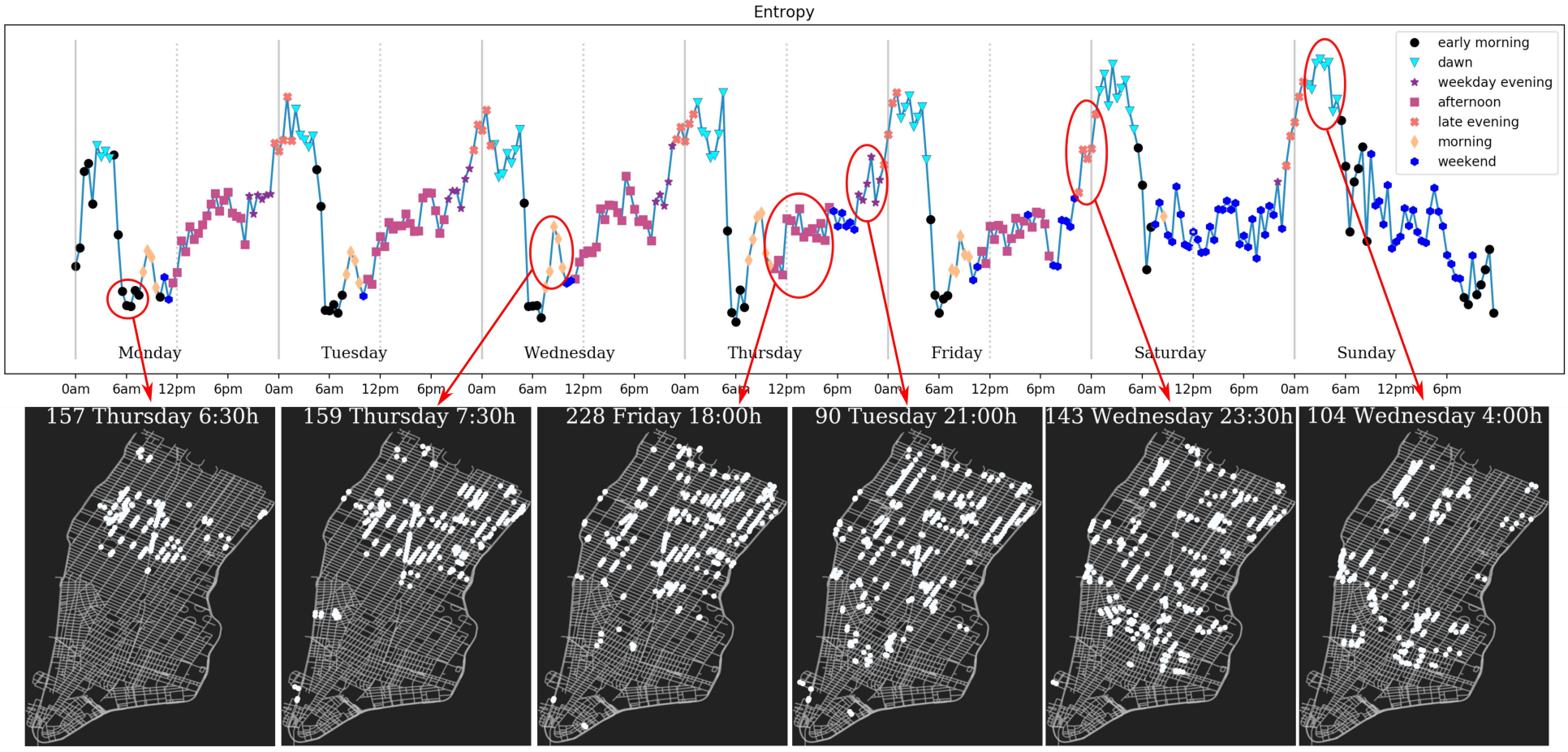
Spatio-Temporal Data Analysis

Analyzing Taxi Pick ups in Downtown Manhattan



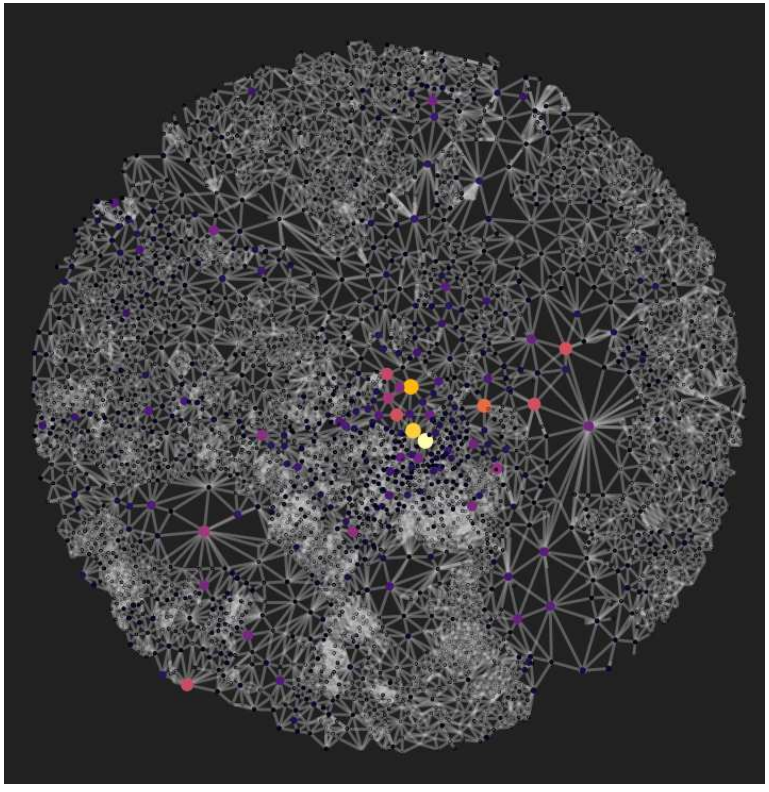
Spatio-Temporal Data Analysis

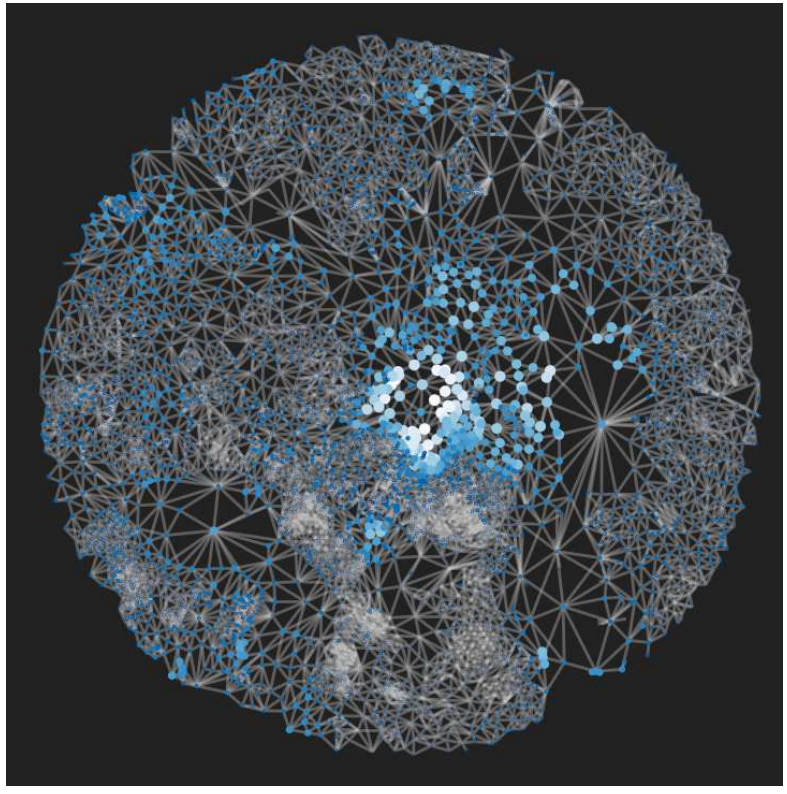
Analyzing Taxi Pick ups in Downtown Manhattan



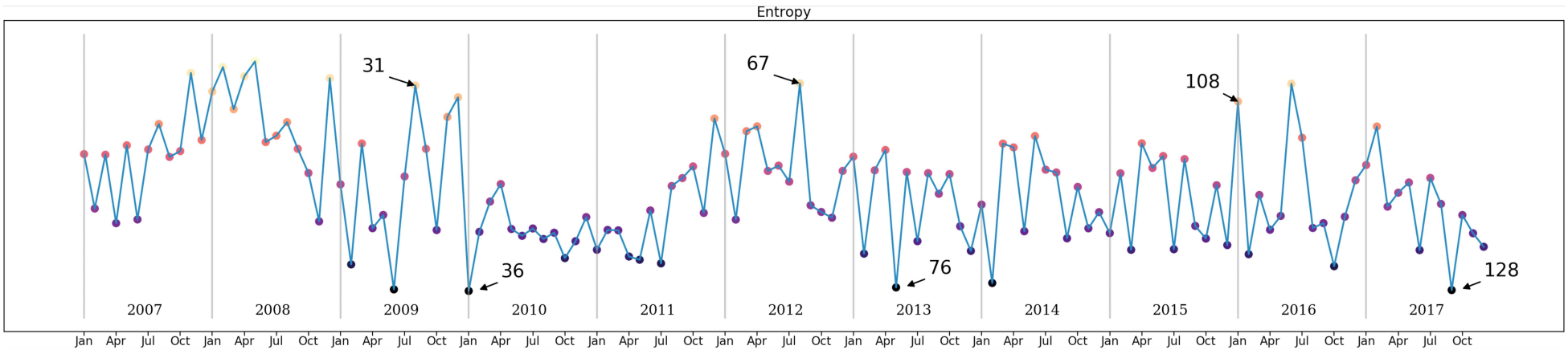
Spatio-Temporal Data Analysis

Analyzing Passerby Robbery in São Paulo City



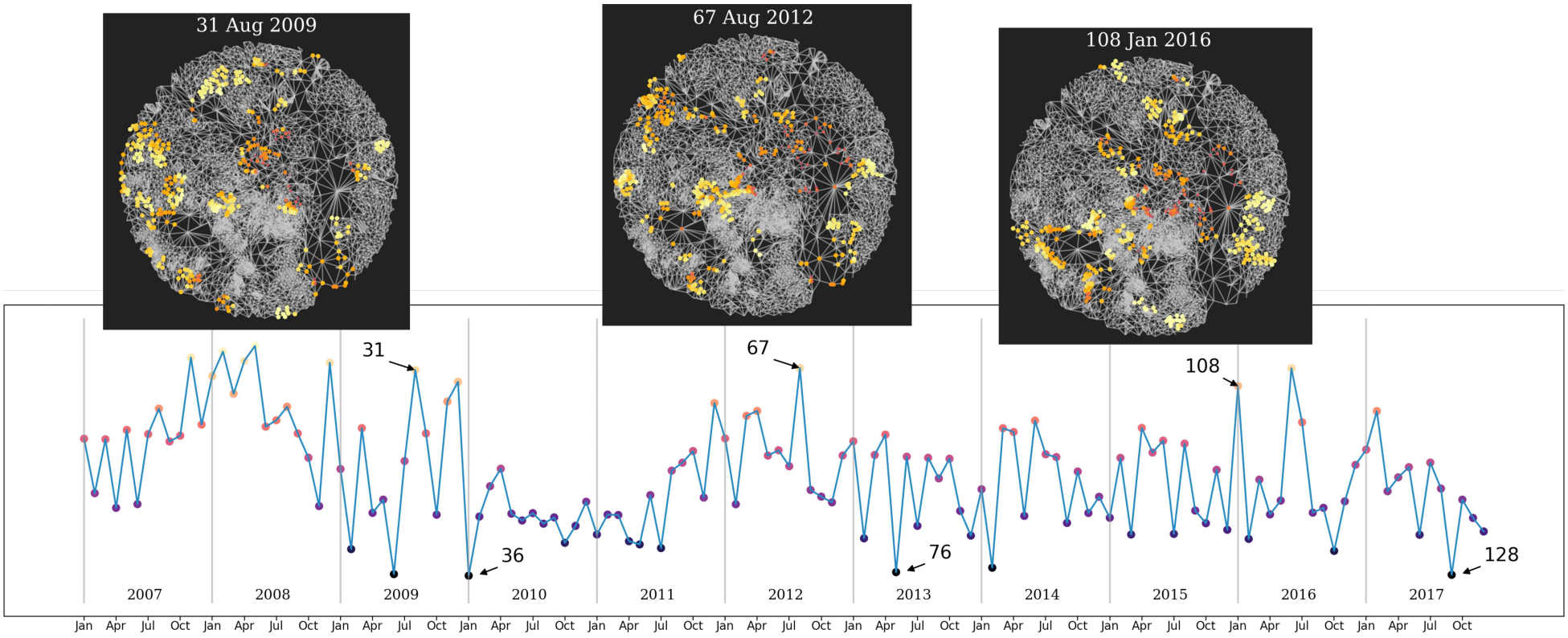


Analyzing Passerby Robbery in São Paulo City



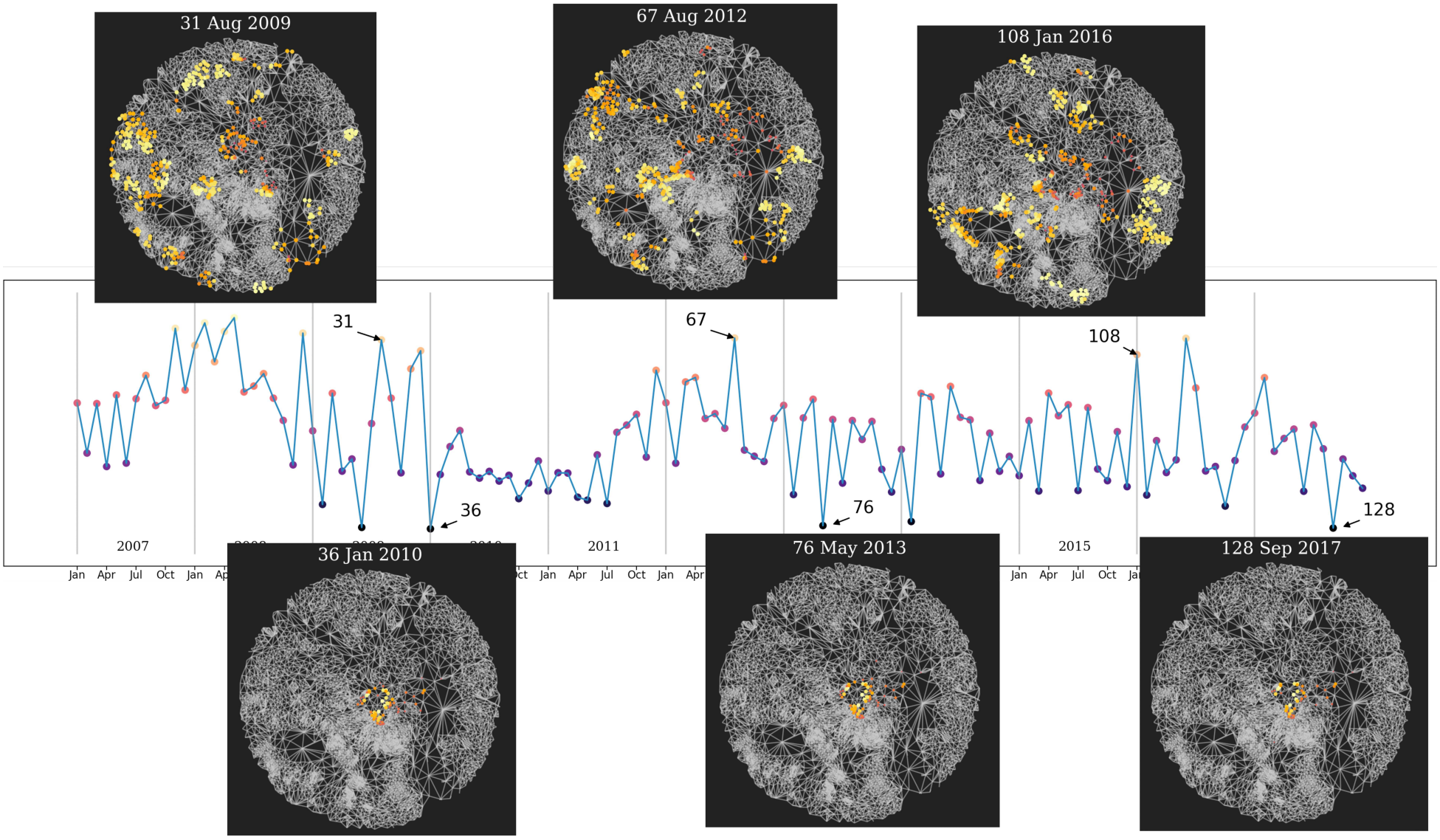
Spatio-Temporal Data Analysis

Analyzing Passerby Robbery in São Paulo City



Spatio-Temporal Data Analysis

Analyzing Passerby Robbery in São Paulo City



Graph Wavelet Transform

Continuous Wavelets Transform

Continuous Wavelets Transform

$$\psi^{s,t}(x) = \frac{1}{s}\psi\left(\frac{x-t}{s}\right) \quad \longrightarrow \quad \mathcal{W}[f](s,t) = \left\langle f, \psi^{s,t}(x) \right\rangle$$

Continuous Wavelets Transform

$$\psi^{s,t}(x) = \frac{1}{s}\psi\left(\frac{x-t}{s}\right) \quad \longrightarrow \quad \mathcal{W}[f](s,t) = \left\langle f, \psi^{s,t}(x) \right\rangle$$

Wavelets should satisfies the admissibility condition:

$$\int \frac{|\widehat{\psi}(\lambda)|^2}{|\lambda|} d\lambda < \infty \quad \longrightarrow \quad |\widehat{\psi}(0)|^2 = 0 \text{ (band-pass filter)}$$

Continuous Wavelets Transform

$$\psi^{s,t}(x) = \frac{1}{s}\psi\left(\frac{x-t}{s}\right) \quad \longrightarrow \quad \mathcal{W}[f](s,t) = \left\langle f, \psi^{s,t}(x) \right\rangle$$

Wavelets should satisfies the admissibility condition:

$$\int \frac{|\widehat{\psi}(\lambda)|^2}{|\lambda|} d\lambda < \infty \quad \longrightarrow \quad |\widehat{\psi}(0)|^2 = 0 \text{ (band-pass filter)}$$

In order to adapt those concepts to the context of graphs we must define translation and dilatation operations on graphs.

Continuous Wavelets Transform

$$\psi^{s,t}(x) = \frac{1}{s}\psi\left(\frac{x-t}{s}\right) \quad \longrightarrow \quad \mathcal{W}[f](s,t) = \left\langle f, \psi^{s,t}(x) \right\rangle$$

Wavelets should satisfies the admissibility condition:

$$\int \frac{|\widehat{\psi}(\lambda)|^2}{|\lambda|} d\lambda < \infty \quad \longrightarrow \quad |\widehat{\psi}(0)|^2 = 0 \text{ (band-pass filter)}$$

In order to adapt those concepts to the context of graphs we must define translation and dilatation operations on graphs.

Moreover, the sought graph wavelet basis must be band pass filters with particular properties.

Graph Signal Translation

The shift operation $x - a$ (subtraction of vertices) is not directly defined on graphs.

Graph Signal Translation

The shift operation $x - a$ (subtraction of vertices) is not directly defined on graphs.

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

$$(T_a f)(x) = (f * \delta_a)(x)$$

Graph Signal Translation

The shift operation $x - a$ (subtraction of vertices) is not directly defined on graphs.

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

$$(T_a f)(x) = (f * \delta_a)(x)$$

Since $\mathcal{F}[\delta_j](\lambda_l) =< \delta_j, u_l >= u_l(j)$, we can define graph signal translation (regarding a fixed vertex τ_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 Signal Translation

The shift operation $x - a$ (subtraction of vertices) is not directly defined on graphs.

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

$$(T_a f)(x) = (f * \delta_a)(x)$$

Since $\mathcal{F}[\delta_j](\lambda_l) =< \delta_j, u_l >= u_l(j)$, we can define graph signal translation (regarding a fixed vertex τ_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 preserves the mean of the signal.

Graph Signal Dilation

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

Graph Signal Dilation

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

$\frac{x}{s}$ is not defined in the graph domain.

Graph Signal Dilation

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

$\frac{x}{s}$ is not defined in the graph domain.

But we can define dilation via GFT:

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

Graph Signal Dilation

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

$\frac{x}{s}$ is not defined in the graph domain.

But we can define dilation via GFT:

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

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

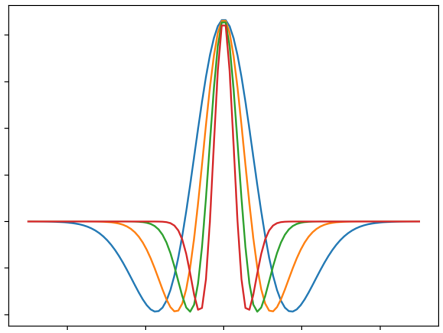
The property

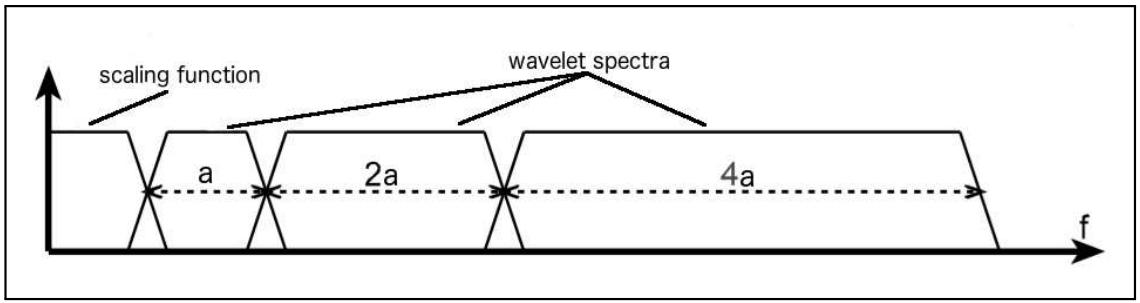
$$\frac{1}{a} f\left(\frac{x}{a}\right) \longleftrightarrow \widehat{f}(a\lambda)$$

The property

$$\frac{1}{a} f\left(\frac{x}{a}\right) \longleftrightarrow \widehat{f}(a\lambda)$$

implies that stretching a band-limited function in the spatial domain shrinks and shifts its spectrum.

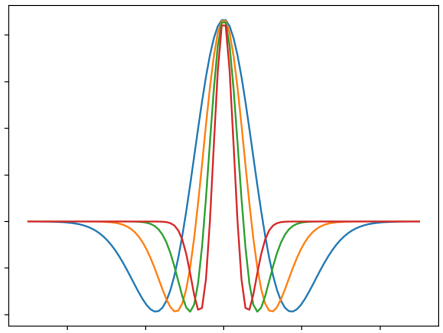


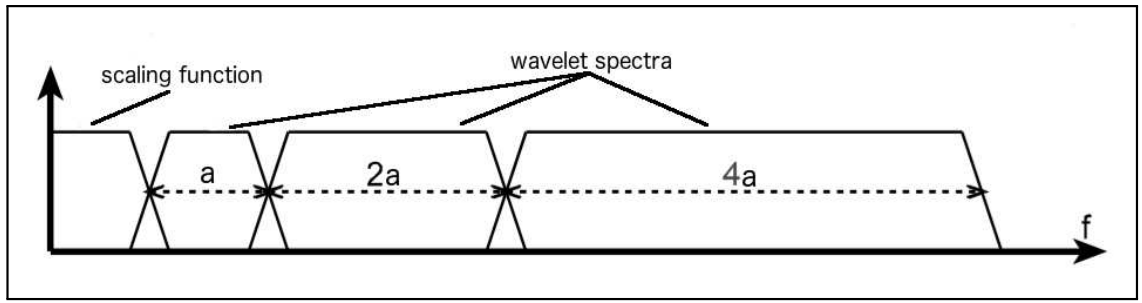


The property

$$\frac{1}{a} f\left(\frac{x}{a}\right) \longleftrightarrow \widehat{f}(a\lambda)$$

implies that stretching a band-limited function in the spatial domain shrinks and shifts its spectrum.





A complete representation demands to complete the basis with a low-pass scaling function.

Hammond et al. have shown that the following set of filters satisfy the shrink and shift property:

$$g(x) = \begin{cases} x^2 & \text{for } x < 1 \\ -5 + 11x - 6x^2 + x^3 & \text{for } 1 \leq x \leq 2 \\ 4x^{-2} & \text{for } 2 < x \end{cases}.$$

and that

$$h(\lambda) = \gamma \exp(-(\frac{10\lambda}{0.3\lambda_n})^4)$$

gives rise to a proper scaling function.

Graph Wavelet Transform

Hammond et al. have shown that the following set of filters satisfy the shrink and shift property:

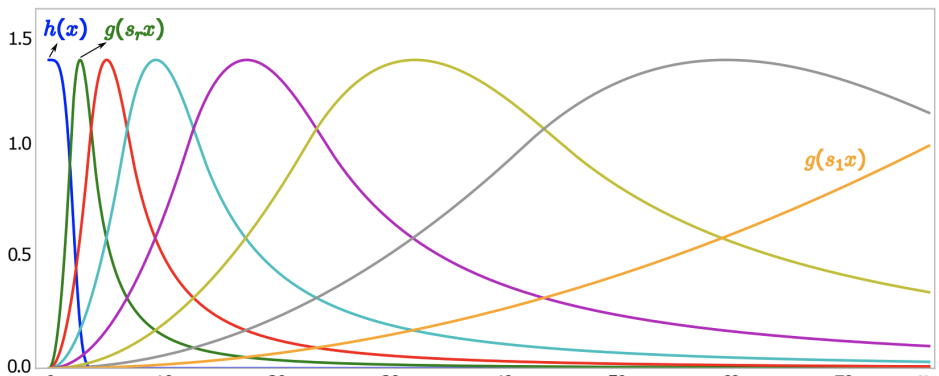
$$g(x) = \begin{cases} x^2 & \text{for } x < 1 \\ -5 + 11x - 6x^2 + x^3 & \text{for } 1 \leq x \leq 2 \\ 4x^{-2} & \text{for } 2 < x \end{cases}.$$

and that

$$h(\lambda) = \gamma \exp(-(\frac{10\lambda}{0.3\lambda_n})^4)$$

gives rise to a proper scaling function.

$g(s\lambda)$ and $h(\lambda)$



Graph Wavelet Transform

Graph Wavelet Basis

$$\psi^{s,j} = \sum_l g(s\lambda_l) u_l(j) u_l$$

Graph Wavelet Transform

Graph Wavelet Basis

$$\psi^{s,j} = \sum_l g(s\lambda_l) u_l(j) u_l$$

Graph Wavelet Transform (GWT)

$$\mathcal{W}[f](s,j) = \left\langle f, \psi^{s,j} \right\rangle = \sum_l g(s\lambda_l) \hat{f}(\lambda_l) u_l(j)$$

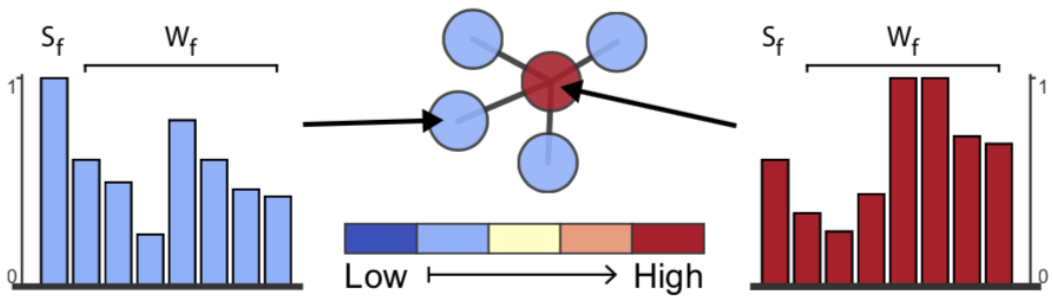
Graph Wavelet Transform

Graph Wavelet Basis

$$\psi^{s,j} = \sum_l g(s\lambda_l) u_l(j) u_l$$

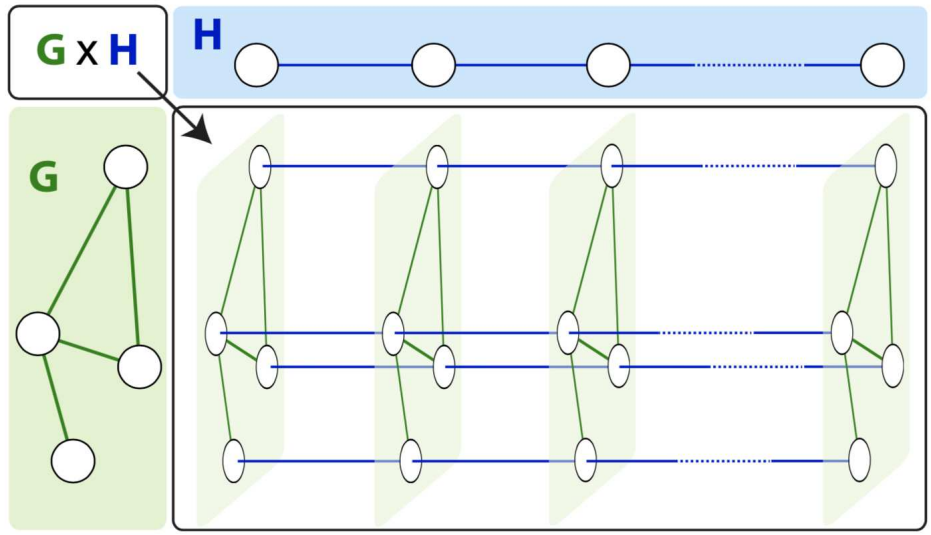
Graph Wavelet Transform (GWT)

$$\mathcal{W}[f](s,j) = \left\langle f, \psi^{s,j} \right\rangle = \sum_l g(s\lambda_l) \hat{f}(\lambda_l) u_l(j)$$

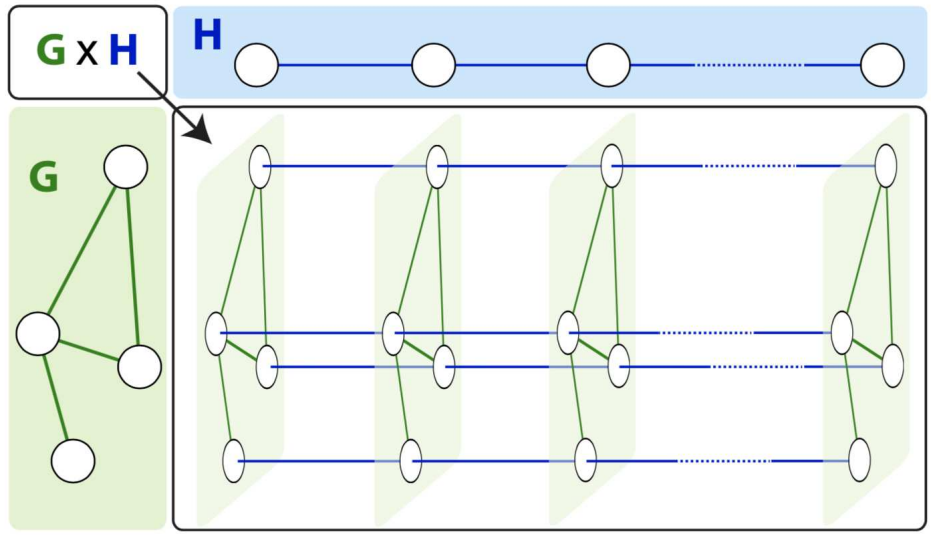


GWT for Spatio-Temporal Data Analysis

Joint Space and Time Analysis



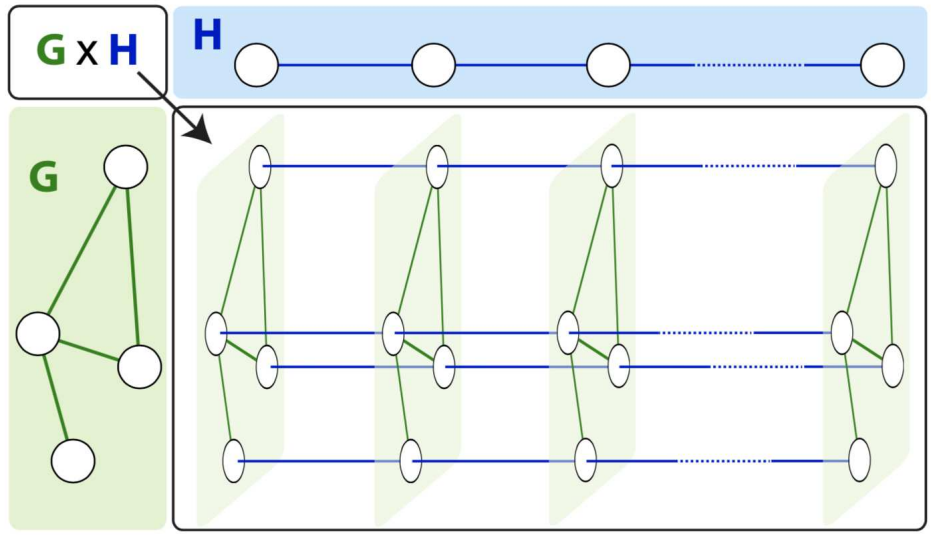
Joint Space and Time Analysis



$$\begin{cases} (u_l, \lambda_l) \text{ eigenpair of } G \\ (v_k, \mu_k) \text{ eigenpair of } H \end{cases} \Rightarrow (u_l \otimes v_k, \lambda_l + \mu_k) \text{ eigenpair of } G \times H$$

where $\otimes$ is the Kronecker product.

Joint Space and Time Analysis

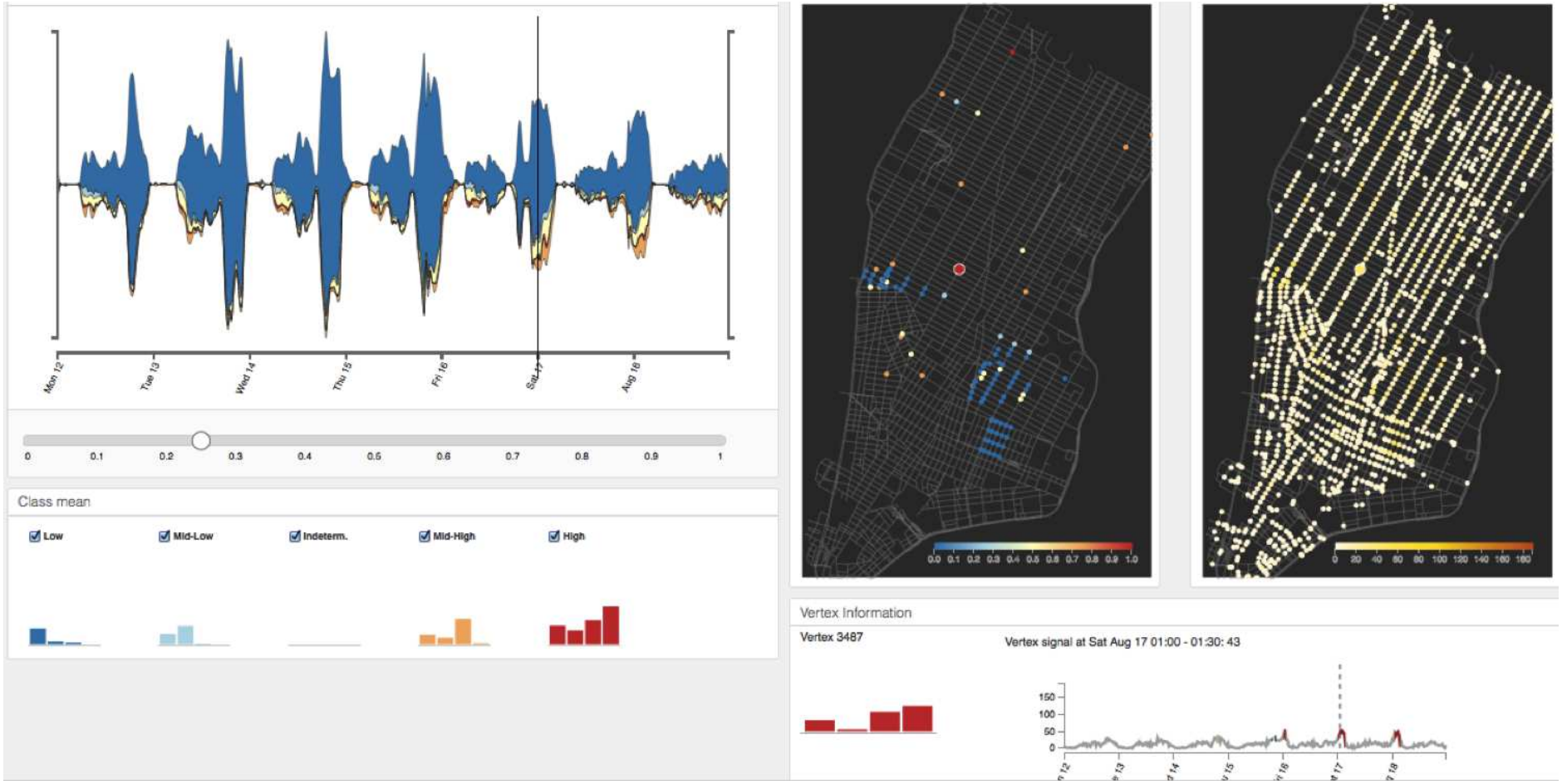


$$\begin{cases} (u_l, \lambda_l) \text{ eigenpair of } G \\ (v_k, \mu_k) \text{ eigenpair of } H \end{cases} \Rightarrow (u_l \otimes v_k, \lambda_l + \mu_k) \text{ eigenpair of } G \times H$$

where $\otimes$ is the Kronecker product.

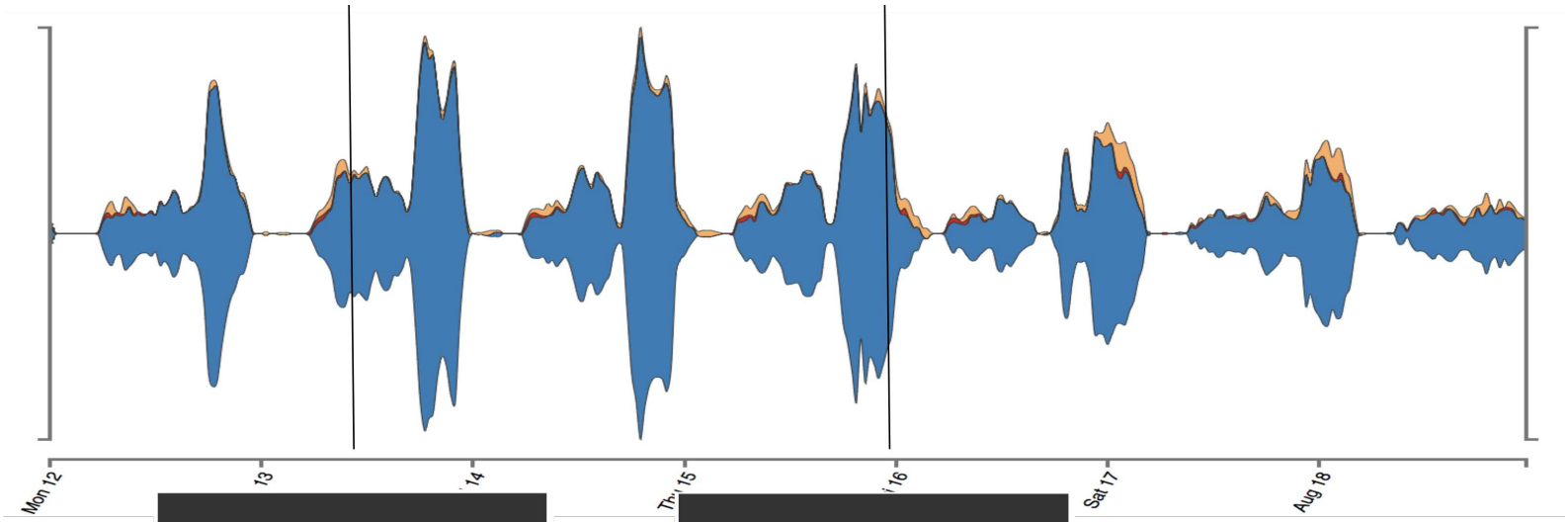
This property is extremely useful to make the joint space and time analysis computationally feasible.

Analyzing taxi pickups

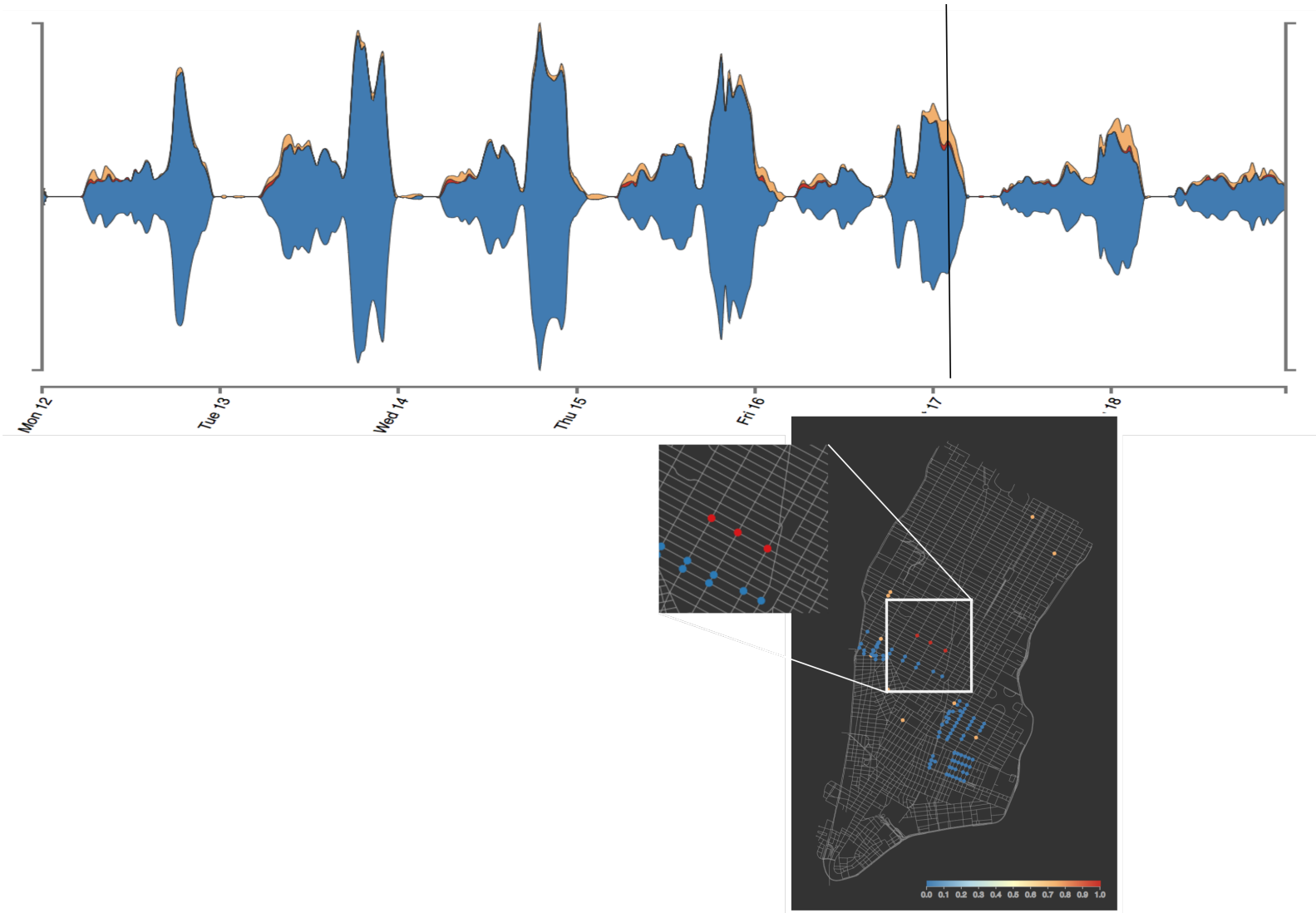


More than 1.5 million nodes and 2 million edges.

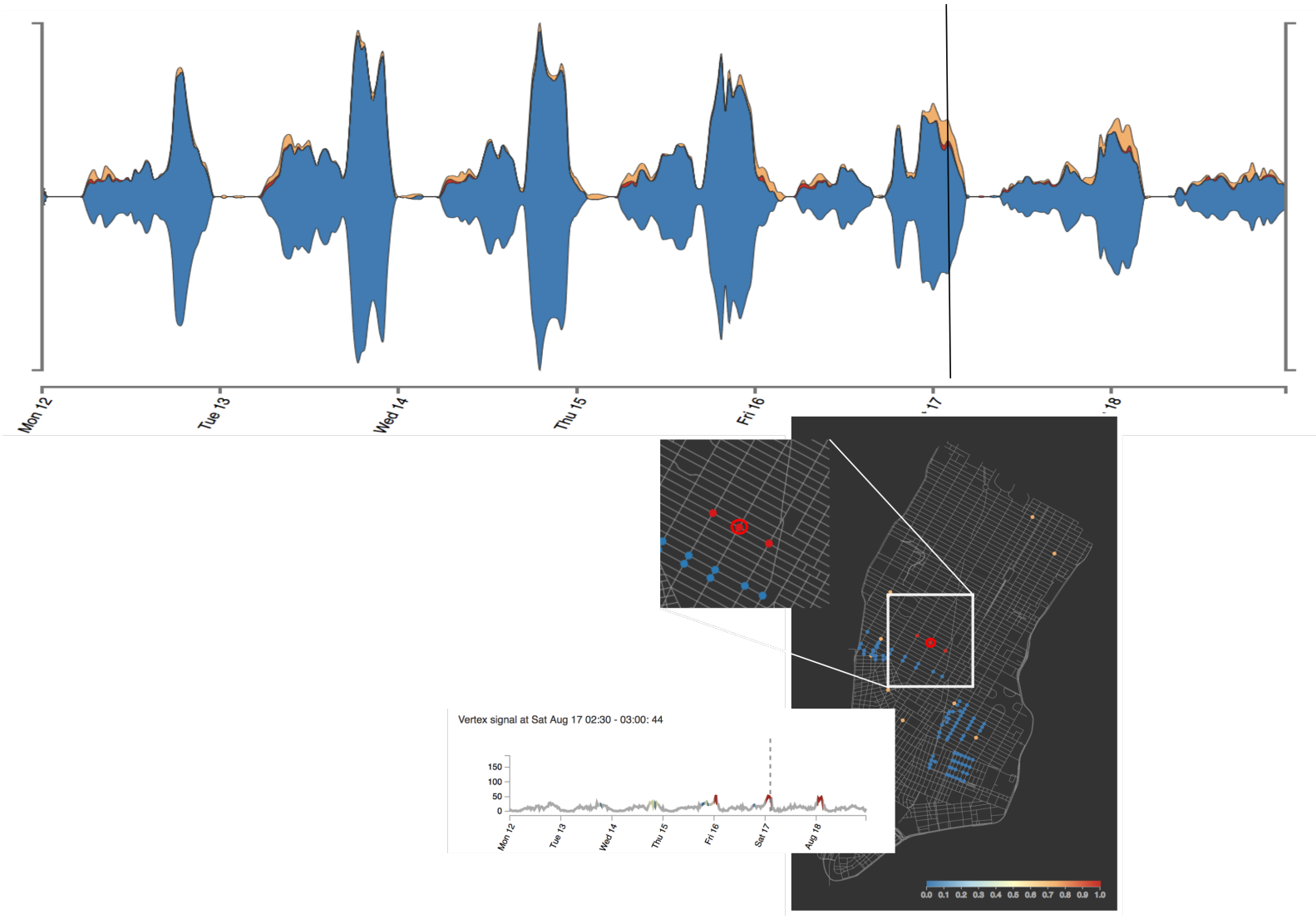
Analyzing taxi pickups



Analyzing taxi pickups



Analyzing taxi pickups



Conclusions

Conclusions

- GSP is a powerful tool to assist spatio-temporal data analysis

Conclusions

- GSP is a powerful tool to assist spatio-temporal data analysis
- The proposed edge detection filter has tuned out to be quite effective to uncover spatial and temporal patterns

Conclusions

- GSP is a powerful tool to assist spatio-temporal data analysis
- The proposed edge detection filter has tuned out to be quite effective to uncover spatial and temporal patterns
- GWT enables the simultaneous analysis of space and time, which is hard to be done with other techniques, mainly when dealing with large data sets.

Conclusions

- GSP is a powerful tool to assist spatio-temporal data analysis
- The proposed edge detection filter has tuned out to be quite effective to uncover spatial and temporal patterns
- GWT enables the simultaneous analysis of space and time, which is hard to be done with other techniques, mainly when dealing with large data sets.
- *Future*:

Conclusions

- GSP is a powerful tool to assist spatio-temporal data analysis
- The proposed edge detection filter has tuned out to be quite effective to uncover spatial and temporal patterns
- GWT enables the simultaneous analysis of space and time, which is hard to be done with other techniques, mainly when dealing with large data sets.
- *Future*:
  - the use of machine learning to design optimal filters is an almost unexplored problem

Conclusions

- GSP is a powerful tool to assist spatio-temporal data analysis
- The proposed edge detection filter has tuned out to be quite effective to uncover spatial and temporal patterns
- GWT enables the simultaneous analysis of space and time, which is hard to be done with other techniques, mainly when dealing with large data sets.
- *Future*:
  - the use of machine learning to design optimal filters is an almost unexplored problem
  - a better understanding on how space and time individually contribute to the GWT coefficients is an important issue (not properly tackled yet).

Conclusions

- GSP is a powerful tool to assist spatio-temporal data analysis
- The proposed edge detection filter has tuned out to be quite effective to uncover spatial and temporal patterns
- GWT enables the simultaneous analysis of space and time, which is hard to be done with other techniques, mainly when dealing with large data sets.
- *Future*:
  - the use of machine learning to design optimal filters is an almost unexplored problem
  - a better understanding on how space and time individually contribute to the GWT coefficients is an important issue (not properly tackled yet).
  - very hot topic is the interplay between GSP + Deep Learning

THANKS !!