Fast Topology Identification from Smooth Graph Signals

S. Saman Saboksayr, Gonzalo Mateos, and Mujdat Cetin

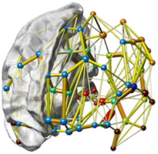
Dept. of ECE and Goergen Institute for Data Science
University of Rochester
gmateosb@ece.rochester.edu
http://www.ece.rochester.edu/~gmateosb/

Ack.: NSF Awards CCF-1750428, ECCS-1809356, CCF-1934962

Novi Sad, Serbia, September 20-22, 2021

What is this talk about?

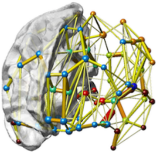
- Learning graphs from nodal observations
- Ex: Central to network neuroscience
  - $\Rightarrow$ Functional network from fMRI signals



What is this talk about?

- Learning graphs from nodal observations
- Ex: Central to network neuroscience

  $\Rightarrow$ Functional network from fMRI signals



- Most GSP works: how known graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ affects signals and filters
  - Feasible for e.g., physical or infrastructure networks
  - Links are tangible and directly observable
- Still, acquisition of updated topology information is challenging

  $\Rightarrow$ Sheer size, reconfiguration, privacy and security

- Here, reverse path: how to use GSP to infer the graph topology?
- **Goal:** fast, scalable algorithm with convergence rate guarantees

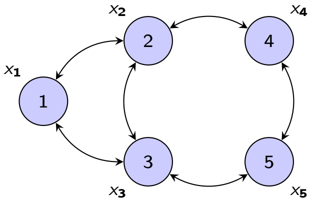
Graph signal processing (GSP)

- Graph $\mathcal{G}$ with adjacency matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$

  $\Rightarrow W_{ij} =$ proximity between i and j

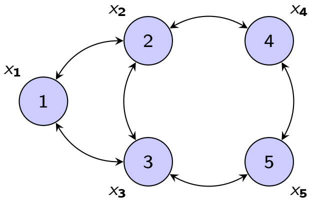
- Define a signal $\mathbf{x} \in \mathbb{R}^N$ on top of the graph

  $\Rightarrow x_i =$ signal value at node $i \in \mathcal{V}$



Graph signal processing (GSP)

- Graph $\mathcal{G}$ with adjacency matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$
  $\Rightarrow W_{ij}$ = proximity between i and j
- Define a signal $\mathbf{x} \in \mathbb{R}^N$ on top of the graph
  $\Rightarrow x_i$ = signal value at node $i \in \mathcal{V}$

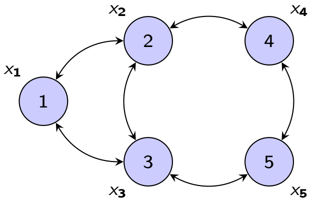


- Total variation of signal $\mathbf{x}$ with respect to Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$

$$\text{TV}(\mathbf{x}) = \mathbf{x}^\top \mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{i \neq j} W_{ij}(x_i - x_j)^2$$

Graph signal processing (GSP)

- Graph $\mathcal{G}$ with adjacency matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$
  $\Rightarrow W_{ij}$ = proximity between i and j
- Define a signal $\mathbf{x} \in \mathbb{R}^N$ on top of the graph
  $\Rightarrow x_i$ = signal value at node $i \in \mathcal{V}$



- Total variation of signal $\mathbf{x}$ with respect to Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$

$$\text{TV}(\mathbf{x}) = \mathbf{x}^\top \mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{i \neq j} W_{ij}(x_i - x_j)^2$$

- Graph Signal Processing $\rightarrow$ Exploit structure encoded in $\mathbf{L}$ to process $\mathbf{x}$
  $\Rightarrow$ Use GSP to learn the underlying $\mathcal{G}$ or a meaningful network model

Problem formulation

Rationale

- Seek graphs on which data admit certain regularities
  - Nearest-neighbor prediction (a.k.a. graph smoothing)
  - Semi-supervised learning
  - Efficient information-processing transforms
- Many real-world graph signals are smooth (i.e., $\text{TV}(\mathbf{x})$ is small)
  - Graphs based on similarities among vertex attributes
  - Network formation driven by homophily, proximity in latent space

Rationale

- Seek graphs on which data admit certain regularities
  - Nearest-neighbor prediction (a.k.a. graph smoothing)
  - Semi-supervised learning
  - Efficient information-processing transforms

- Many real-world graph signals are smooth (i.e., $\text{TV}(\mathbf{x})$ is small)
  - Graphs based on similarities among vertex attributes
  - Network formation driven by homophily, proximity in latent space

Problem statement

Given observations $\mathcal{X} := \{\mathbf{x}_p\}_{p=1}^P$, identify a graph $\mathcal{G}$ such that signals in $\mathcal{X}$ are smooth on $\mathcal{G}$.

Signal smoothness meets edge sparsity

- Form $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_P] \in \mathbb{R}^{N \times P}$, let $\bar{\mathbf{x}}_i^\top \in \mathbb{R}^{1 \times P}$ denote its i-th row
  $\Rightarrow$ Euclidean distance matrix $\mathbf{E} \in \mathbb{R}_+^{N \times N}$, where $E_{ij} := \|\bar{\mathbf{x}}_i - \bar{\mathbf{x}}_j\|^2$
- Neat trick: link between smoothness and sparsity [Kalofolias'16]

$$\sum_{p=1}^{P} \mathrm{TV}(\mathbf{x}_p) = \mathrm{trace}(\mathbf{X}^\top \mathbf{L} \mathbf{X}) = \frac{1}{2} \|\mathbf{W} \circ \mathbf{E}\|_1$$

  $\Rightarrow$ Sparse $\mathcal{E}$ when data come from a smooth manifold
  $\Rightarrow$ Favor candidate edges (i, j) associated with small E_{ij}
- Shows that edge sparsity on top of smoothness is redundant

Signal smoothness meets edge sparsity

- Form $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_P] \in \mathbb{R}^{N \times P}$, let $\bar{\mathbf{x}}_i^\top \in \mathbb{R}^{1 \times P}$ denote its i-th row
  $\Rightarrow$ Euclidean distance matrix $\mathbf{E} \in \mathbb{R}_+^{N \times N}$, where $E_{ij} := \|\bar{\mathbf{x}}_i - \bar{\mathbf{x}}_j\|^2$
- Neat trick: link between smoothness and sparsity [Kalofolias'16]

$$\sum_{p=1}^{P} \text{TV}(\mathbf{x}_p) = \text{trace}(\mathbf{X}^\top \mathbf{L} \mathbf{X}) = \frac{1}{2} \|\mathbf{W} \circ \mathbf{E}\|_1$$

  $\Rightarrow$ Sparse $\mathcal{E}$ when data come from a smooth manifold
  $\Rightarrow$ Favor candidate edges (i, j) associated with small E_{ij}
- Shows that edge sparsity on top of smoothness is redundant
- Parameterize graph learning problems in terms of $\mathbf{W}$ (instead of $\mathbf{L}$)
  $\Rightarrow$ Advantageous since constraints on $\mathbf{W}$ are decoupled

Scalable topology identification framework

- General purpose graph-learning framework

$$\min_{\mathbf{W}} \left\{ \|\mathbf{W} \circ \mathbf{E}\|_1 - \alpha \mathbf{1}^\top \log(\mathbf{W}\mathbf{1}) + \frac{\beta}{2}\|\mathbf{W}\|_F^2 \right\}$$
$$\text{s. to} \quad \text{diag}(\mathbf{W}) = \mathbf{0}, \; W_{ij} = W_{ji} \geq 0, \; i \neq j$$

  $\Rightarrow$ Logarithmic barrier forces positive degrees $\mathbf{d} = \mathbf{W}\mathbf{1}$

  $\Rightarrow$ Penalize large edge-weights to control sparsity

- Efficient algorithms incurring $O(N^2)$ cost

  $\Rightarrow$ Primal-dual (PD) [Kalofolias'16] and ADMM [Wang et al'21]

- Cost has no Lipschitz gradient $\rightarrow$ No convergence rates

V. Kalofolias, "How to learn a graph from smooth signals," *AISTATS*, 2016

Equivalent reformulation

- Handle constraints on entries of $\mathbf{W}$
  - Hollow and symmetric $\rightarrow$ Retain $\mathbf{w} := \mathrm{vec}[\mathrm{triu}[\mathbf{W}]] \in \mathbb{R}_+^{N(N-1)/2}$
  - Non-negative $\rightarrow \mathbb{I}\{\mathbf{w} \succeq \mathbf{0}\} = 0$ if $\mathbf{w} \succeq \mathbf{0}$, else $\mathbb{I}\{\mathbf{w} \succeq \mathbf{0}\} = \infty$
- Equivalent unconstrained, non-differentiable reformulation

$$\min_{\mathbf{w}} \Big\{ \underbrace{\mathbb{I}\{\mathbf{w} \succeq \mathbf{0}\} + 2\mathbf{w}^\top \mathbf{e} + \beta\|\mathbf{w}\|_2^2}_{:=f(\mathbf{w})} - \underbrace{\alpha \mathbf{1}^\top \log(\mathbf{S}\mathbf{w})}_{:=-g(\mathbf{S}\mathbf{w})} \Big\}$$

$\Rightarrow$ $\mathbf{S}$ maps edge weights to nodal degrees, i.e., $\mathbf{d} = \mathbf{S}\mathbf{w}$

Equivalent reformulation

- Handle constraints on entries of $\mathbf{W}$
  - Hollow and symmetric $\rightarrow$ Retain $\mathbf{w} := \text{vec}[\text{triu}[\mathbf{W}]] \in \mathbb{R}_+^{N(N-1)/2}$
  - Non-negative $\rightarrow \mathbb{I}\{\mathbf{w} \succeq \mathbf{0}\} = 0$ if $\mathbf{w} \succeq \mathbf{0}$, else $\mathbb{I}\{\mathbf{w} \succeq \mathbf{0}\} = \infty$
- Equivalent unconstrained, non-differentiable reformulation

$$\min_{\mathbf{w}} \Big\{ \underbrace{\mathbb{I}\{\mathbf{w} \succeq \mathbf{0}\} + 2\mathbf{w}^\top \mathbf{e} + \beta \|\mathbf{w}\|_2^2}_{:=f(\mathbf{w})} - \underbrace{\alpha \mathbf{1}^\top \log(\mathbf{S}\mathbf{w})}_{:=-g(\mathbf{S}\mathbf{w})} \Big\}$$

$\Rightarrow$ $\mathbf{S}$ maps edge weights to nodal degrees, i.e., $\mathbf{d} = \mathbf{S}\mathbf{w}$

- Non-differentiable $f(\mathbf{w})$ is strongly convex, $g(\mathbf{d})$ is strictly convex
  - Problem $\min_{\mathbf{w}}\{f(\mathbf{w}) + g(\mathbf{S}\mathbf{w})\}$ has a unique optimal solution $\mathbf{w}^\star$
  - Amenable to fast dual-based proximal gradient (FDPG) solver

A. Beck and M. Teboulle, "A fast dual proximal gradient algorithm for convex minimization and applications," *Oper. Res. Lett.*, 2014

Dual problem and its properties

- Variable splitting: $\min_{\mathbf{w},\mathbf{d}} \{f(\mathbf{w}) + g(\mathbf{d})\}$, s. to $\mathbf{d} = \mathbf{S}\mathbf{w}$
  - Attach Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^N$ to equality constraints
  - Lagrangian $\mathcal{L}(\mathbf{w}, \mathbf{d}, \boldsymbol{\lambda}) = f(\mathbf{w}) + g(\mathbf{d}) - \langle \boldsymbol{\lambda}, \mathbf{S}\mathbf{w} - \mathbf{d} \rangle$

- (Minimization form) dual problem is $\min_{\boldsymbol{\lambda}} \{F(\boldsymbol{\lambda}) + G(\boldsymbol{\lambda})\}$, where

$$F(\boldsymbol{\lambda}) := \max_{\mathbf{w}} \left\{ \langle \mathbf{S}^\top \boldsymbol{\lambda}, \mathbf{w} \rangle - f(\mathbf{w}) \right\},$$

$$G(\boldsymbol{\lambda}) := \max_{\mathbf{d}} \{ \langle -\boldsymbol{\lambda}, \mathbf{d} \rangle - g(\mathbf{d}) \}$$

Dual problem and its properties

- Variable splitting: $\min_{\mathbf{w},\mathbf{d}} \{f(\mathbf{w}) + g(\mathbf{d})\}$, s. to $\mathbf{d} = \mathbf{S}\mathbf{w}$
  - Attach Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^N$ to equality constraints
  - Lagrangian $\mathcal{L}(\mathbf{w}, \mathbf{d}, \boldsymbol{\lambda}) = f(\mathbf{w}) + g(\mathbf{d}) - \langle \boldsymbol{\lambda}, \mathbf{S}\mathbf{w} - \mathbf{d} \rangle$
- (Minimization form) dual problem is $\min_{\boldsymbol{\lambda}} \{F(\boldsymbol{\lambda}) + G(\boldsymbol{\lambda})\}$, where

$$F(\boldsymbol{\lambda}) := \max_{\mathbf{w}} \left\{ \langle \mathbf{S}^\top \boldsymbol{\lambda}, \mathbf{w} \rangle - f(\mathbf{w}) \right\},$$
$$G(\boldsymbol{\lambda}) := \max_{\mathbf{d}} \left\{ \langle -\boldsymbol{\lambda}, \mathbf{d} \rangle - g(\mathbf{d}) \right\}$$

- Strong convexity of f implies a Lipschitz gradient property for F

Lemma. Function $F(\boldsymbol{\lambda})$ is smooth, and the gradient $\nabla F(\boldsymbol{\lambda})$ is Lipschitz continuous with constant $L := \frac{N-1}{\beta}$.

Fast dual-based proximal gradient method

- **Key:** apply accelerated proximal gradient method to the dual

$$\boldsymbol{\lambda}_k = \mathbf{prox}_{L^{-1}G}\left(\boldsymbol{\omega}_k - \frac{1}{L}\nabla F(\boldsymbol{\omega}_k)\right),$$

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2},$$

$$\boldsymbol{\omega}_{k+1} = \boldsymbol{\lambda}_k + \left(\frac{t_k - 1}{t_{k+1}}\right)[\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k-1}]$$

Fast dual-based proximal gradient method

- **Key:** apply accelerated proximal gradient method to the dual

$$\boldsymbol{\lambda}_k = \mathbf{prox}_{L^{-1}G}\left(\boldsymbol{\omega}_k - \frac{1}{L}\nabla F(\boldsymbol{\omega}_k)\right),$$

$$t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2},$$

$$\boldsymbol{\omega}_{k+1} = \boldsymbol{\lambda}_k + \left(\frac{t_k - 1}{t_{k+1}}\right)[\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k-1}]$$

- Rewrite in terms of problem parameters L, α, β, $\mathbf{S}$, signals in $\mathbf{e}$

Proposition. The dual variable update iteration can be equivalently rewritten as $\boldsymbol{\lambda}_k = \boldsymbol{\omega}_k - L^{-1}(\mathbf{S}\bar{\mathbf{w}}_k - \mathbf{u}_k)$, with

$$\bar{\mathbf{w}}_k = \max\left(\mathbf{0}, \frac{\mathbf{S}^\top\boldsymbol{\omega}_k - 2\mathbf{e}}{2\beta}\right),$$

$$\mathbf{u}_k = \frac{\mathbf{S}\bar{\mathbf{w}}_k - L\boldsymbol{\omega}_k + \sqrt{(\mathbf{S}\bar{\mathbf{w}}_k - L\boldsymbol{\omega}_k)^2 + 4\alpha L\mathbf{1}}}{2}$$

Algorithm summary

Algorithm 1: Topology inference via fast dual PG (FDPG)

Input parameters α, β, data $\mathbf{e}$, set $L = \frac{N-1}{\beta}$.

Initialize $t_1 = 1$ and $\boldsymbol{\omega}_1 = \boldsymbol{\lambda}_0$ at random.

for $k = 1, 2, \ldots,$ **do**

$$\bar{\mathbf{w}}_k = \max\left(\mathbf{0}, \frac{\mathbf{S}^\top \boldsymbol{\omega}_k - 2\mathbf{e}}{2\beta}\right)$$

$$\mathbf{u}_k = \frac{\mathbf{S}\bar{\mathbf{w}}_k - L\boldsymbol{\omega}_k + \sqrt{(\mathbf{S}\bar{\mathbf{w}}_k - L\boldsymbol{\omega}_k)^2 + 4\alpha L \mathbf{1}}}{2}$$

$$\boldsymbol{\lambda}_k = \boldsymbol{\omega}_k - L^{-1}(\mathbf{S}\bar{\mathbf{w}}_k - \mathbf{u}_k)$$

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2}$$

$$\boldsymbol{\omega}_{k+1} = \boldsymbol{\lambda}_k + \left(\frac{t_k - 1}{t_{k+1}}\right)[\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k-1}]$$

end

Output graph estimate $\hat{\mathbf{w}}_k = \max\left(\mathbf{0}, \frac{\mathbf{S}^\top \boldsymbol{\lambda}_k - 2\mathbf{e}}{2\beta}\right)$

- Complexity of $O(N^2)$ in par with state-of-the-art algorithms
- Non-accelerated dual proximal gradient (DPG) method for $t_k \equiv 1,\ k \geq 1$

Convergence rate analysis

- Let $\boldsymbol{\lambda}^\star$ be a minimizer of the dual cost $\varphi(\boldsymbol{\lambda}) := F(\boldsymbol{\lambda}) + G(\boldsymbol{\lambda})$. Then

$$\varphi(\boldsymbol{\lambda}_k) - \varphi(\boldsymbol{\lambda}^\star) \leq \frac{2(N-1)\|\boldsymbol{\lambda}_0 - \boldsymbol{\lambda}^\star\|_2^2}{\beta k^2}$$

$\Rightarrow$ Celebrated $O(1/k^2)$ rate for FISTA [Beck-Teboulle'09]

- Let $\boldsymbol{\lambda}^\star$ be a minimizer of the dual cost $\varphi(\boldsymbol{\lambda}) := F(\boldsymbol{\lambda}) + G(\boldsymbol{\lambda})$. Then

$$\varphi(\boldsymbol{\lambda}_k) - \varphi(\boldsymbol{\lambda}^\star) \leq \frac{2(N-1)\|\boldsymbol{\lambda}_0 - \boldsymbol{\lambda}^\star\|_2^2}{\beta k^2}$$

$\Rightarrow$ Celebrated $O(1/k^2)$ rate for FISTA [Beck-Teboulle'09]

- Construct a primal sequence $\hat{\mathbf{w}}_k = \text{argmin}_{\mathbf{w}}\, \mathcal{L}(\mathbf{w}, \mathbf{d}, \boldsymbol{\lambda}_k)$

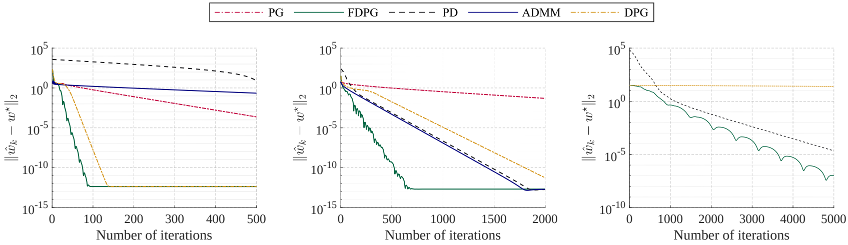
$$\hat{\mathbf{w}}_k = \underset{\mathbf{w}}{\text{argmax}} \left\{ \langle \mathbf{S}^\top \boldsymbol{\lambda}_k, \mathbf{w} \rangle - f(\mathbf{w}) \right\} = \max \left(\mathbf{0}, \frac{\mathbf{S}^\top \boldsymbol{\lambda}_k - 2\mathbf{e}}{2\beta} \right)$$

Theorem. For all $k \geq 1$, the primal sequence $\hat{\mathbf{w}}_k$ defined in terms of dual iterates $\boldsymbol{\lambda}_k$ generated by Algorithm 1 satistifies

$$\|\hat{\mathbf{w}}_k - \mathbf{w}^\star\|_2 \leq \frac{\sqrt{2(N-1)}\|\boldsymbol{\lambda}_0 - \boldsymbol{\lambda}^\star\|_2}{\beta k}.$$

Convergence performance

- Recovery of random and real-world graphs from simulated signals
  - Networks: (a) SBM, $N = 400$; (b) brain, $N = 66$; (c) MN road, $N = 2642$
  - Signals: $P = 1000$ i.i.d. smooth signals $\mathbf{x}_p \sim \mathcal{N}(\mathbf{0}, \mathbf{L}^{\dagger} + 10^{-2}\mathbf{I}_N)$
  - Examine evolution of primal variable error $\|\hat{\mathbf{w}}_k - \mathbf{w}^{\star}\|_2$



(a) (b) (c)

- FDPG converges markedly faster, uniformly across graph classes

Closing remarks

- Network topology inference cornerstone problem in Network Science
  - Most GSP works analyze how $\mathcal{G}$ affect signals and filters
  - Here, reverse path: How to use GSP to infer the graph topology?

- Novel algorithm to learn graphs from observations of smooth signals

  $\Rightarrow$ Cardinal property of many real-world graph signals

  $\Rightarrow$ Ex: sensor measurements, movie ratings, protein annotations

Closing remarks

- Network topology inference cornerstone problem in Network Science
  - Most GSP works analyze how $\mathcal{G}$ affect signals and filters
  - Here, reverse path: How to use GSP to infer the graph topology?
- Novel algorithm to learn graphs from observations of smooth signals
  - $\Rightarrow$ Cardinal property of many real-world graph signals
  - $\Rightarrow$ Ex: sensor measurements, movie ratings, protein annotations
- Fast dual-based proximal gradient (FDPG) iterations
  - $\Rightarrow$ Optimization method so far unexplored for graph learning
  - $\Rightarrow$ Markedly faster than state-of-the-art algorithms
  - $\Rightarrow$ Comes with convergence rate guarantees

Try it out! `http://www.ece.rochester.edu/~gmateosb/code/FDPG.zip`