Unrolling of Simplicial ElasticNet for Edge Flow Signal Reconstruction

Chengen Liu

Prof. Geert Leus

Dr. Elvin Isufi

11/01/2024

Email: C.Liu-15@tudelft.nl

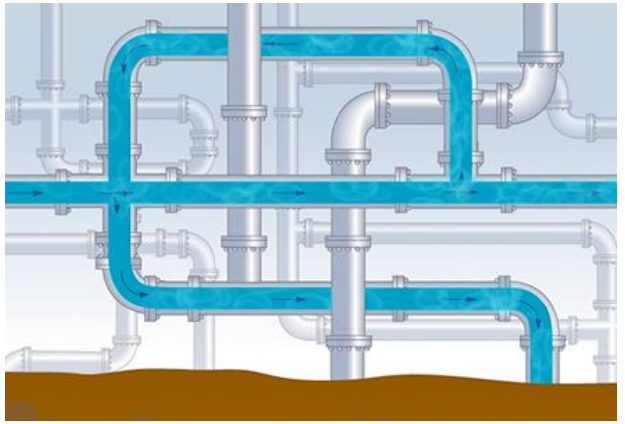
TU Delft

Edge flow

- Many real-world flows can be modeled as edge flows
  a. Traffic flow on roads
  b. Water flow in the water network
  c. Foreign currency exchange flow
  d. ...

- Analysis of edge flow is important for:
  a. making real-time decisions related to these edge flows
  b. monitoring faults on edge flows

Traffic flow

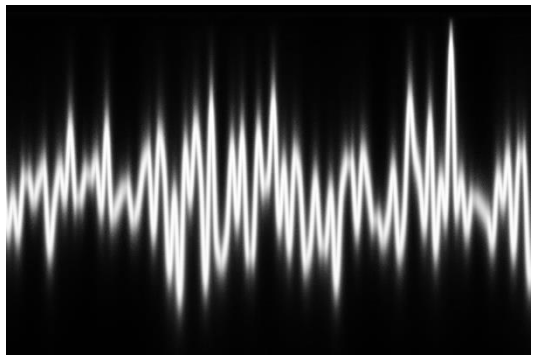


Water flow

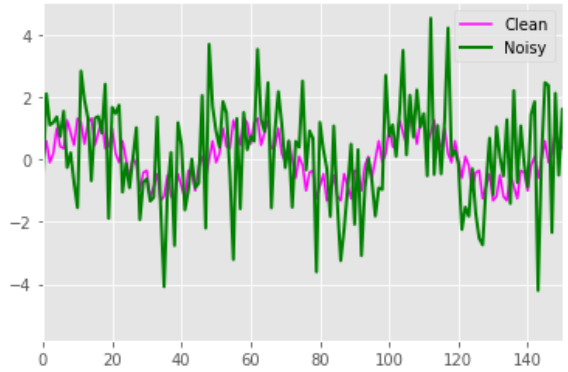
TU Delft

Edge flow reconstruction

- Edge flow data we obtain is defective because of:
  a. noise generated by sensors
  b. lack of sensors in each edge
  c. a portion of the data is lost
  d. ...



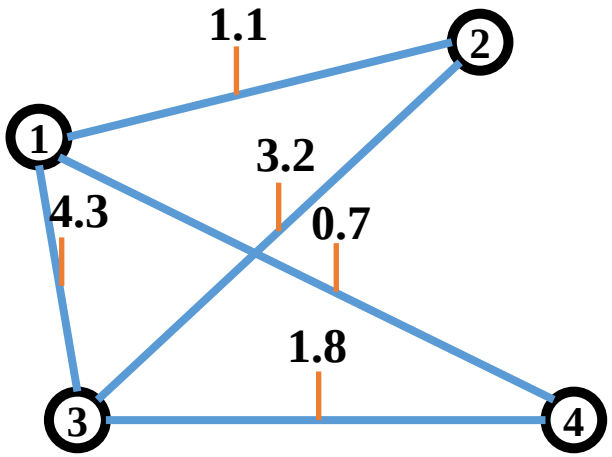
- Reconstructing edge flow data is meaningful because:
  a. it improves the integrity and accuracy of edge flow data
  b. it facilitates downstream tasks
  c. it helps us get a more accurate information about the edge flow



TU Delft

How to model edge flows?

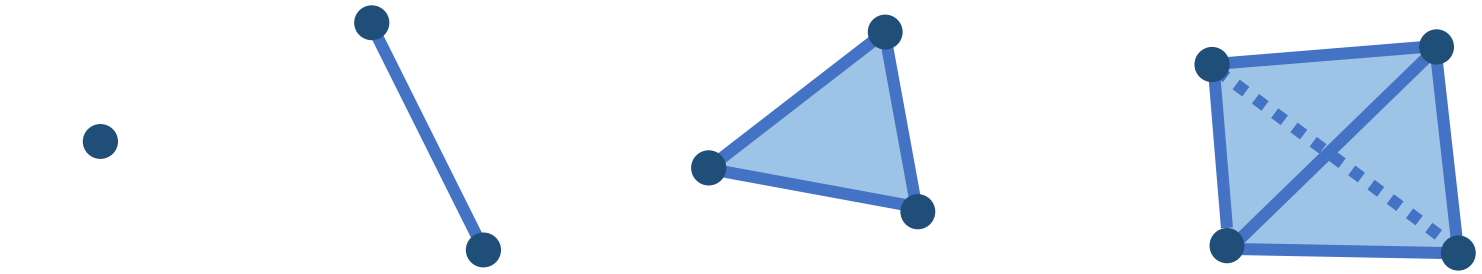
- We can model these edge flows by the edges in the graphs, but:
  a. it only contains nodes and edges
  b. graphs only model pair-wise relationships
  c. the expressive power of graphs is limited
  d. ...



- Thus, simplicial complexes can be considered, which:
  a. is a higher order network
  b. contains nodes, edges, triangles, tetrahedrons...
  c. models multi-way relationships
  d. has stronger expressive power

TU Delft

Simplicial complexes



0-simplex 1-simplex 2-simplex 3-simplex

- Simplicial complexes is a higher-order networks contains different orders of simplices:
  a. 0-simplices are nodes which can model singel element
  b. 1-simplices are edges which can model pair-wise relationships
  c. 2-simplices are triangles which can model relationships between 3 elements
  d. 3-simplices are tetrahedrons which can model relationships between 4 elements
  e. ....

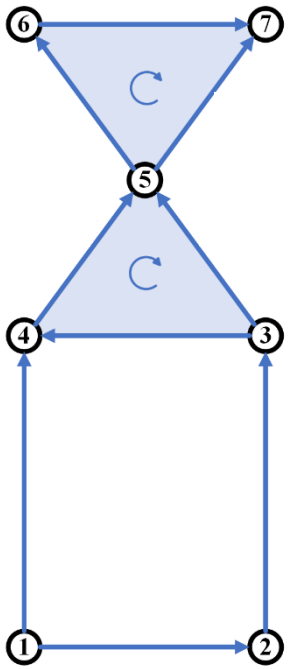
TU Delft

Topology of simplicial complexes

- Topology of simplices can be represented by incidence matrices $\mathbf{B}_k$:
  a. $\mathbf{B}_1$: node-to-edge incidence matrix
  b. $\mathbf{B}_2$: edge-to-triangle incidence matrix
  c. …
- For example, the incidence matrices for this simplicial complexes are:

$$\mathbf{B}_1 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \end{bmatrix}$$

$$\mathbf{B}_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \\ 0 & 1 \end{bmatrix}$$



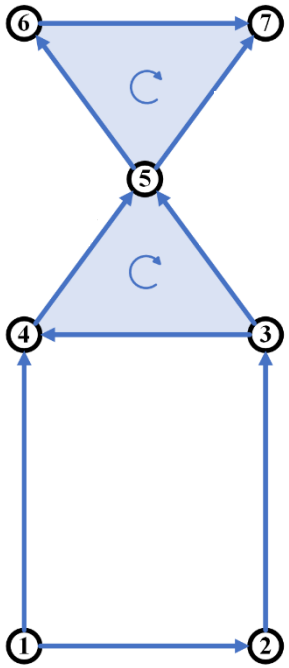
An example of SC

Topology of simplicial complexes

- The whole structure can be represented by Hodge Laplacian matrices :

$$\mathbf{L}_0 = \mathbf{B}_1\mathbf{B}_1^\top$$
$$\mathbf{L}_k = \mathbf{B}_k^\top\mathbf{B}_k + \mathbf{B}_{k+1}\mathbf{B}_{k+1}^\top, k = 1, \ldots, K-1$$
$$\mathbf{L}_K = \mathbf{B}_K^\top\mathbf{B}_K$$

- There are two terms in Hodge Laplacian matrices $\mathbf{L}_k$:
  a. Lower laplacian: $\mathbf{L}_{k,\ell} = \mathbf{B}_k^\top\mathbf{B}_k$ represents lower adjacencies
  b. Upper laplacian: $\mathbf{L}_{k,u} = \mathbf{B}_{k+1}\mathbf{B}_{k+1}^T$ represents upper adjacencies



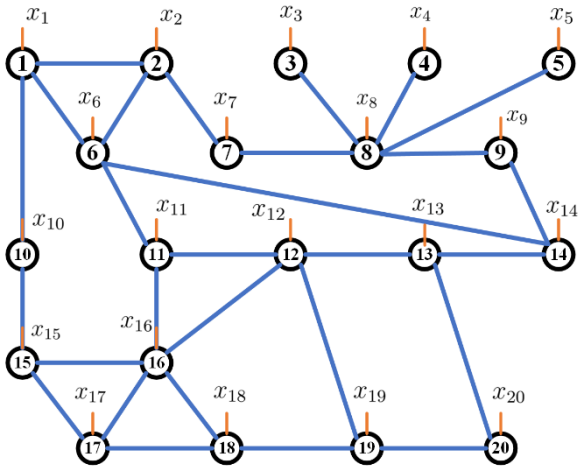
An example of SC

For example: edge (3, 4) and (1, 4) are lower adjacent; edge (3, 4) and (4, 5) are upper adjacent

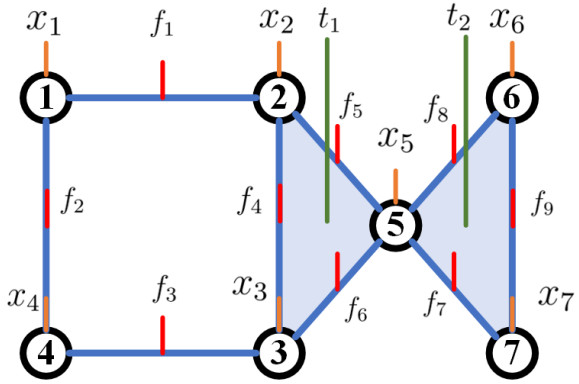
TU Delft

Simplicial signals

- In topological signal processing (TSP):
  a. Signals are defined on nodes, edges, triangles, tetrahedrons…
  b. We collect k−signal into the vector $\mathbf{s}^k = \left[s_1^k, \ldots, s_{N_k}^k\right]^\top$
  c. Simplicial signals with different orders are coupled



Graph signals



Simplicial signals

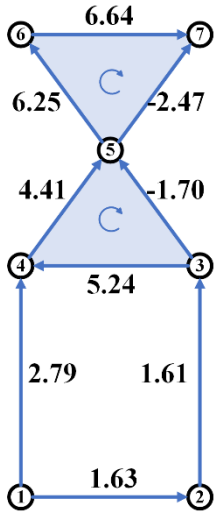
Simplicial signals

- The space of simplicial edge flow signals can be decomposed into three subspaces:

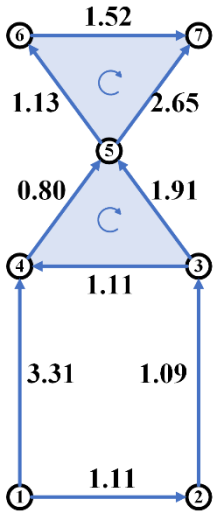
$$\mathbb{R}^{N_1} \equiv \mathrm{im}\left(\mathbf{B}_1^\top\right) \oplus \ker\left(\mathbf{L}_1\right) \oplus \mathrm{im}\left(\mathbf{B}_2\right)$$

- Any simplicial edge flow can be decomposed as:

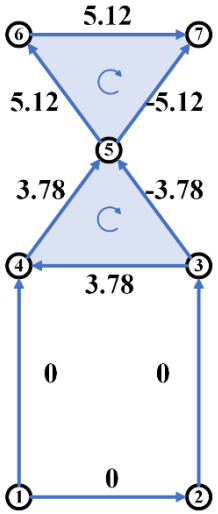
$$\mathbf{f} = \mathbf{f}_\mathrm{G} + \mathbf{f}_\mathrm{C} + \mathbf{f}_\mathrm{H}$$



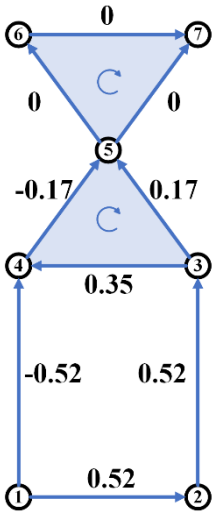
Edge flow $\mathbf{f}$



$\mathbf{f}_G$



$\mathbf{f}_C$

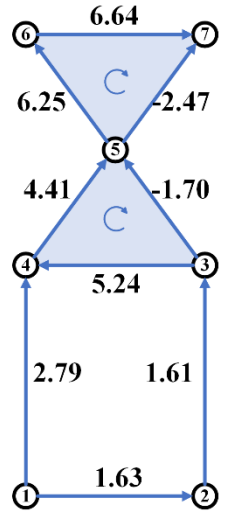


$\mathbf{f}_H$

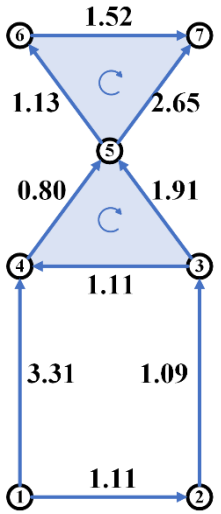
Two operators

- Curl operator:
  a. It is defined as: $\mathrm{curl}(\mathbf{f}) = \mathbf{B}_2^\top \mathbf{f}$
  b. It measures the curl of an edge flow
  c. Curl-free: the curl is zero at each triangle

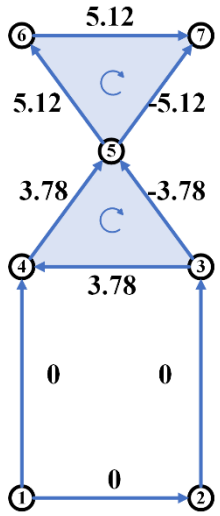
- Divergence operator:
  a. It is defined as: $\mathrm{div}(\mathbf{f}) = \mathbf{B}_1 \mathbf{f}$
  b. It measures the divergence of an edge flow
  c. Divergence-free: the divergence is zero at each nodes



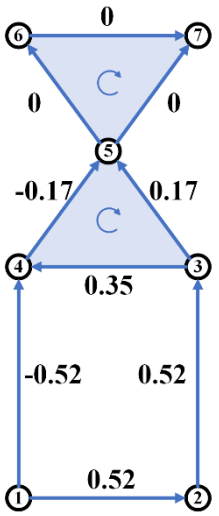
Edge flow $\mathbf{f}$



$\mathbf{f}_G$



$\mathbf{f}_C$



$\mathbf{f}_H$

Problem formulation: Edge flow reconstruction

- The regularized optimization problem for edge flow reconstruction:

$$\underset{\hat{\mathbf{f}} \in \mathbb{R}^{N_1}}{\operatorname{argmin}} \quad \|\mathbf{P}(\hat{\mathbf{f}} - \mathbf{y})\|_2^2 + \sum_{i=1}^{n} r_i(\hat{\mathbf{f}}, \mathcal{S}^1)$$

a. Denoising: When $\mathbf{P} = \mathbf{I}$ and $\mathbf{y} = \mathbf{f}_0 + \mathbf{n}$ is a noisy edge flow

b. Interpolation: When $\mathbf{P} \in \{0, 1\}^{M \times N_1}$ is the sampling matrix and $\mathbf{y}$ is the edge flow with sampled values

c. $\|\mathbf{P}(\hat{\mathbf{f}} - \mathbf{y})\|_2^2$ is a data-fitting term

d. $\sum_{i=1}^{n} r_i(\hat{\mathbf{f}}, \mathcal{S})$ is the regularizer designed based on prior knowledge

TU Delft

Simplicial ElasticNet

- To regularize edge flow reconstruction problem with simplicial prior:

$$\underset{\hat{\mathbf{f}} \in R^{N_1}}{\operatorname{argmin}} \|\mathbf{P}(\hat{\mathbf{f}}-\mathbf{y})\|_2^2 + \alpha_1\|\mathbf{B}_1\hat{\mathbf{f}}\|_1 + \alpha_2\|\mathbf{B}_1\hat{\mathbf{f}}\|_2^2 + \beta_1\|\mathbf{B}_2^\top\hat{\mathbf{f}}\|_1 + \beta_2\|\mathbf{B}_2^\top\hat{\mathbf{f}}\|_2^2 + \gamma_1\|\hat{\mathbf{f}}\|_1 + \gamma_2\|\hat{\mathbf{f}}\|_2^2$$

- There are three groups of ElasticNet regularizers:
  a. $\alpha_1\|\mathbf{B}_1\hat{\mathbf{f}}\|_1 + \alpha_2\|\mathbf{B}_1\hat{\mathbf{f}}\|_2^2$: penalize the divergence
  b. $\beta_1\|\mathbf{B}_2^\top\hat{\mathbf{f}}\|_1 + \beta_2\|\mathbf{B}_2^\top\hat{\mathbf{f}}\|_2^2$: penalize the curl
  c. $\gamma_1\|\hat{\mathbf{f}}\|_1 + \gamma_2\|\hat{\mathbf{f}}\|_2^2$: guarantee the completeness of the problem

TU Delft

ADMM solution

- Iterative steps of four-block ADMM comprise:

$$
\begin{cases}
\hat{\mathbf{f}}^{(k+1)} = (2\mathbf{P}^\top\mathbf{P} + \rho\mathbf{B}_1^\top\mathbf{B}_1 + \rho\mathbf{B}_2\mathbf{B}_2^\top + \rho\mathbf{I})^{-1} \\
\qquad\qquad (2\mathbf{P}^\top\mathbf{P}\mathbf{y} + \mathbf{B}_1^\top\boldsymbol{\lambda}_1^{(k)} + \mathbf{B}_2\boldsymbol{\lambda}_2^{(k)} + \boldsymbol{\lambda}_3^{(k)} \\
\qquad\qquad + \rho\mathbf{B}_1^\top\mathbf{z}_1^{(k)} + \rho\mathbf{B}_2\mathbf{z}_2^{(k)} + \rho\mathbf{z}_3^{(k)}) \\
\mathbf{z}_1^{(k+1)} = S_{\frac{\alpha_1}{2\alpha_2+\rho}}\left(\frac{1}{2\alpha_2+\rho}(\rho\mathbf{B}_1\hat{\mathbf{f}}^{(k+1)} - \boldsymbol{\lambda}_1^{(k)})\right) \\
\mathbf{z}_2^{(k+1)} = S_{\frac{\beta_1}{2\beta_2+\rho}}\left(\frac{1}{2\beta_2+\rho}(\rho\mathbf{B}_2^\top\hat{\mathbf{f}}^{(k+1)} - \boldsymbol{\lambda}_2^{(k)})\right) \\
\mathbf{z}_3^{(k+1)} = S_{\frac{\gamma_1}{2\gamma_2+\rho}}\left(\frac{1}{2\gamma_2+\rho}(\rho\hat{\mathbf{f}}^{(k+1)} - \boldsymbol{\lambda}_3^{(k)})\right) \\
\boldsymbol{\lambda}_1^{(k+1)} = \boldsymbol{\lambda}_1^{(k)} - \rho(\mathbf{B}_1\hat{\mathbf{f}}^{(k+1)} - \mathbf{z}_1^{(k+1)}) \\
\boldsymbol{\lambda}_2^{(k+1)} = \boldsymbol{\lambda}_2^{(k)} - \rho(\mathbf{B}_2^\top\hat{\mathbf{f}}^{(k+1)} - \mathbf{z}_2^{(k+1)}) \\
\boldsymbol{\lambda}_3^{(k+1)} = \boldsymbol{\lambda}_3^{(k)} - \rho(\hat{\mathbf{f}}^{(k+1)} - \mathbf{z}_3^{(k+1)})
\end{cases}
$$

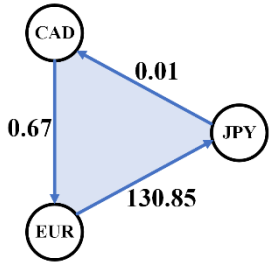
Primal updates

Dual updates

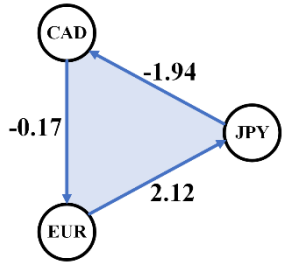
TU Delft

Datasets

- Forex:
  a. 25 nodes, 300 edges, and 2300 triangles
  b. Consider pairwise currency exchanges between 25 different currencies
  c. The no-arbitrage condition is $r^{\mathrm{A/B}} r^{\mathrm{B/C}} = r^{\mathrm{A/C}}$
  d. it is curl-free



Original rate flow



Logarithm rate flow

Datasets

- Lastfm:
  a. 657 nodes, 1997 edges, and 1276 triangles
  b. Records the process of users switching artists while playing music
  c. Switching from artist A to B, add a unit on the edge flow from A to B
  d. Divergence-free

- Chicago Road network:
  a. 546 nodes, 1088 edges and 112 triangles
  b. The traffic network of the city Chicago
  c. Create divergence-free edge flows

TU Delft

ADMM results

- Take the ADMM denoising results (NMSE) as an example:

FOREX	0dB	6dB	10dB
ADMM-STF	0.07	0.02	0.01
ADMM-SEN(ours)	0.07	0.02	0.01
LASTFM	0dB	6dB	10dB
ADMM-STF	0.66	0.16	0.06
ADMM-SEN(ours)	0.66	0.16	0.06
CHICAGO	0dB	6dB	10dB
ADMM-STF	0.49	0.12	0.04
ADMM-SEN(ours)	0.49	0.12	0.04

a. Both simplicial ElasticNet and trend filtering have obvious denoising effect

TU Delft

Motivation of unrolling networks

- Issues with ADMM:
  a. Coefficients of regularizers should be designed in advance based on prior knowledge
  b. Prior knowledge is difficult to obtain
  c. A large number of iterations, thus causes high computation

- Unrolling network:
  a. Each iteration of ADMM is mapped into a layer of unrolling networks
  b. It replaces some fixed parameters in ADMM by some trainable weights in unrolling networks
  c. After the unrolling network is trained, the computation is low

TU Delft

Simplicial unrolling networks for ElasticNet

- The $\hat{\mathbf{f}}$-update of ADMM can be replaced by:

$$\hat{\mathbf{f}}^{(k+1)} = (2\mathbf{P}^{\top}\mathbf{P} + \rho\mathbf{B}_1^{\top}\mathbf{B}_1 + \rho\mathbf{B}_2\mathbf{B}_2^{\top} + \rho\mathbf{I})^{-1} (2\mathbf{P}^{\top}\mathbf{P}\mathbf{y} + \mathbf{B}_1^{\top}\boldsymbol{\lambda}_1^{(k)} + \mathbf{B}_2\boldsymbol{\lambda}_2^{(k)} + \boldsymbol{\lambda}_3^{(k)} + \rho\mathbf{B}_1^{\top}\mathbf{z}_1^{(k)} + \rho\mathbf{B}_2\mathbf{z}_2^{(k)} + \rho\mathbf{z}_3^{(k)})$$

$$\hat{\mathbf{f}}^{(l+1)} = \mathbf{H}_1\mathbf{P}^{\top}\mathbf{P}\mathbf{y} + \mathbf{H}_2\mathbf{B}_1^{\top}\boldsymbol{\lambda}_1^{(l)} + \mathbf{H}_3\mathbf{B}_2\boldsymbol{\lambda}_2^{(l)} + \mathbf{H}_4\boldsymbol{\lambda}_3^{(l)} + \mathbf{H}_5\mathbf{B}_1^{\top}\mathbf{z}_1^{(l)} + \mathbf{H}_6\mathbf{B}_2\mathbf{z}_2^{(l)} + \mathbf{H}_7\mathbf{z}_3^{(l)}$$

a. Simplicial convolutional filters with trainable weights replace fixed parameters
b. SCFs help aggregate information from lower and upper neighbors
c. SCFs help avoid the computation of matrix inverse

TU Delft

Simplicial convolutional filters

- Simplicial convolutional filtering operation:

$$\mathbf{y} = \left(\sum_{l_1=0}^{L_1} \alpha_{l_1} \mathbf{L}_{1,\ell}^{l_1} + \sum_{l_2=0}^{L_2} \beta_{l_2} \mathbf{L}_{1,u}^{l_2} \right) \mathbf{f}$$

- Simplicial convolutional filter [1]:

$$\mathbf{H}(\mathbf{L}_1) := \left(\sum_{l_1=0}^{L_1} \alpha_{l_1} \mathbf{L}_{1,\ell}^{l_1} + \sum_{l_2=0}^{L_2} \beta_{l_2} \mathbf{L}_{1,u}^{l_2} \right)$$

a. The first term $\sum_{l_1=0}^{L_1} \alpha_{l_1} \mathbf{L}_{1,\ell}^{l_1}$ gathers information from lower neighbors

b. The second term $\sum_{l_2=0}^{L_2} \beta_{l_2} \mathbf{L}_{1,u}^{l_2}$ gathers information from upper neighbors

c. L_1 and L_2 are the convolution orders

TU Delft

Simplicial unrolling networks for ElasticNet

- The $\mathbf{z}_i$-update of ADMM can be replaced by:

$$\mathbf{z}_1^{(k+1)} = S_{\frac{\alpha_1}{2\alpha_2+\rho}}\left(\frac{1}{2\alpha_2+\rho}(\rho\mathbf{B}_1\hat{\mathbf{f}}^{(k+1)} - \boldsymbol{\lambda}_1^{(k)})\right) \qquad \mathbf{z}_1^{(l+1)} = S_{\frac{a_1}{2a_2+r_1}}\left(\frac{1}{2a_2+r_1}(r_1\mathbf{B}_1\hat{\mathbf{f}}^{(l+1)} - \boldsymbol{\lambda}_1^{(l)})\right)$$

$$\mathbf{z}_2^{(k+1)} = S_{\frac{\beta_1}{2\beta_2+\rho}}\left(\frac{1}{2\beta_2+\rho}(\rho\mathbf{B}_2^\top\hat{\mathbf{f}}^{(k+1)} - \boldsymbol{\lambda}_2^{(k)})\right) \qquad \mathbf{z}_2^{(l+1)} = S_{\frac{b_1}{2b_2+r_2}}\left(\frac{1}{2b_2+r_2}(r_2\mathbf{B}_2^\top\hat{\mathbf{f}}^{(l+1)} - \boldsymbol{\lambda}_2^{(l)})\right)$$

$$\mathbf{z}_3^{(k+1)} = S_{\frac{\gamma_1}{2\gamma_2+\rho}}\left(\frac{1}{2\gamma_2+\rho}(\rho\hat{\mathbf{f}}^{(k+1)} - \boldsymbol{\lambda}_3^{(k)})\right) \qquad \mathbf{z}_3^{(l+1)} = S_{\frac{c_1}{2c_2+r_3}}\left(\frac{1}{2c_2+r_3}(r_3\hat{\mathbf{f}}^{(l+1)} - \boldsymbol{\lambda}_3^{(l)})\right)$$

a. Fixed penalty parameter ρ are replaced by 3 different trainable weights

b. Fixed regularizers coefficients are replaced by trainable weights

TU Delft

Simplicial unrolling networks for ElasticNet

- The $\boldsymbol{\lambda}_i$-update of ADMM can be replaced by:

$$\boldsymbol{\lambda}_1^{(k+1)} = \boldsymbol{\lambda}_1^{(k)} - \rho(\mathbf{B}_1\hat{\mathbf{f}}^{(k+1)} - \mathbf{z}_1^{(k+1)}) \qquad \boldsymbol{\lambda}_1^{(l+1)} = \boldsymbol{\lambda}_1^{(l)} - r_1(\mathbf{B}_1\hat{\mathbf{f}}^{(l+1)} - \mathbf{z}_1^{(l+1)})$$

$$\boldsymbol{\lambda}_2^{(k+1)} = \boldsymbol{\lambda}_2^{(k)} - \rho(\mathbf{B}_2^\top\hat{\mathbf{f}}^{(k+1)} - \mathbf{z}_2^{(k+1)}) \quad \Rightarrow \quad \boldsymbol{\lambda}_2^{(l+1)} = \boldsymbol{\lambda}_2^{(l)} - r_2(\mathbf{B}_2^\top\hat{\mathbf{f}}^{(l+1)} - \mathbf{z}_2^{(l+1)})$$

$$\boldsymbol{\lambda}_3^{(k+1)} = \boldsymbol{\lambda}_3^{(k)} - \rho(\hat{\mathbf{f}}^{(k+1)} - \mathbf{z}_3^{(k+1)}) \qquad \boldsymbol{\lambda}_3^{(l+1)} = \boldsymbol{\lambda}_3^{(l)} - r_3(\hat{\mathbf{f}}^{(l+1)} - \mathbf{z}_3^{(l+1)})$$

a. Fixed penalty parameter ρ are replaced by 3 different trainable weights

The k-th iteration of ADMM is mapped into l-th layer of unrolling networks

TU Delft

Simplicial unrolling networks vs. ADMM

- The denoising result (NMSE) of unrolling networks and ADMM:

a. Overall, unrolling networks perform better than ADMM
b. When SNR is small, unrolling network is better
c. Layers of unrolling network are much less than iterations of ADMM

FOREX	0dB	6dB	10dB
ADMM-STF	0.07	0.02	0.01
ADMM-SEN(ours)	0.07	0.02	0.01
USEN(ours)	0.07	0.02	0.01
USTF(ours)	**0.07**	**0.02**	**0.01**
LASTFM	0dB	6dB	10dB
ADMM-STF	0.66	0.16	0.06
ADMM-SEN(ours)	0.66	0.16	0.06
USEN(ours)	**0.09**	**0.05**	**0.03**
USTF(ours)	0.09	0.06	0.04
CHICAGO	0dB	6dB	10dB
ADMM-STF	0.49	0.12	0.04
ADMM-SEN(ours)	0.49	0.12	0.04
USEN(ours)	**0.34**	**0.11**	**0.05**
USTF(ours)	0.35	0.11	0.05

TU Delft

Simplicial unrolling networks vs. other neural networks

- Other neural networks:
  a. MLP [2]: Multilayer perceptrons
  b. SNN [3]: Simplicial neural networks
  c. SCNN [4]: Simplicial convolutional neural networks
  d. GUTF [5]: Graph unrolling network for trend filtering

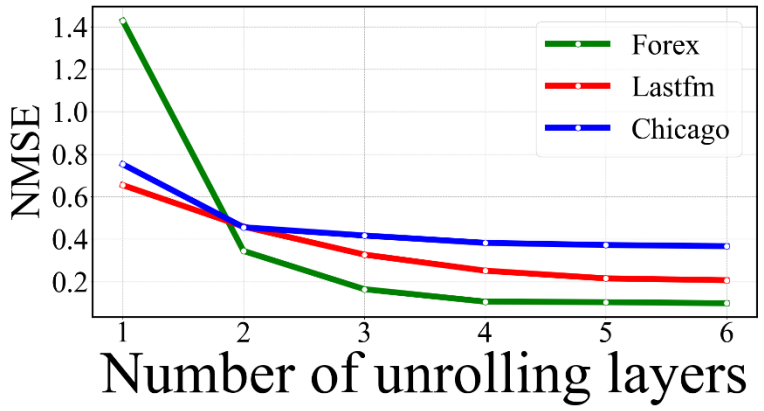
- Comments on results (NMSE):
  a. Trained in one-shot learning
  b. Simplicial unrolling networks perform the best
  c. GUTF performs the worst

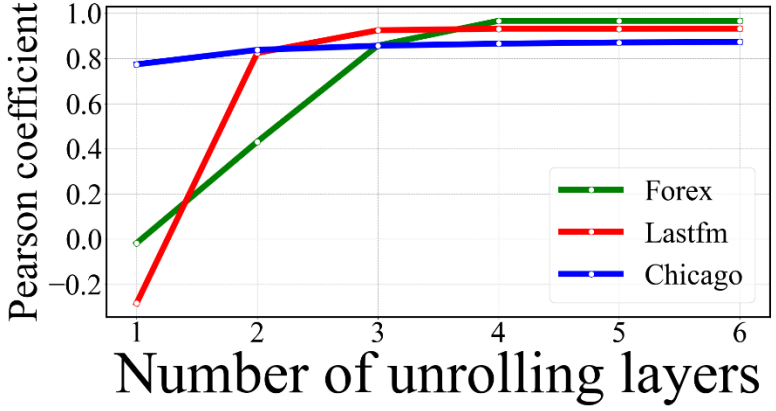
FOREX	0dB	6dB	10dB
MLP	0.71	0.28	0.12
SNN	0.49	0.20	0.09
SCNN	0.11	0.04	0.01
GUTF	0.46	0.19	0.09
USEN(ours)	0.07	0.02	0.01
USTF(ours)	**0.07**	**0.02**	**0.01**
LASTFM	0dB	6dB	10dB
MLP	0.39	0.16	0.07
SNN	0.27	0.12	0.07
SCNN	0.21	0.11	0.05
GUTF	0.87	0.87	0.86
USEN(ours)	**0.09**	**0.05**	**0.03**
USTF(ours)	0.09	0.06	0.04
CHICAGO	0dB	6dB	10dB
MLP	0.53	0.21	0.09
SNN	0.44	0.18	0.10
SCNN	0.37	0.15	0.05
GUTF	0.76	0.61	0.55
USEN(ours)	**0.34**	**0.11**	**0.05**
USTF(ours)	0.35	0.11	0.05

TU Delft

Convergence of simplicial unrolling networks

- If we check the output at each layer of the unrolling networks:





a. The NMSE and Pearson coefficient converge gradually

b. This is because each layer of unrolling networks can be regarded as an iteration in ADMM

Conclusion

- Simplicial ElasticNet for the edge flow reconstruction
  a. Considers both ℓ_1 and ℓ_2 norm
  b. Solved by multi-block ADMM
- Simplicial unrolling networks
  a. Fixed parameters are replaced by learnable weights
  b. Simplicial convolutional filters
- Numerical experiments
  a. Real-world and artificial datasets
  b. Conventional iterative algorithms, non-model-based neural networks and unrolling networks

TU Delft

Reference

[1] M. Yang, E. Isufi, M. T. Schaub, and G. Leus,“Simplicial convolutional filters,” IEEE Transactions on Signal Processing, vol. 70, pp. 4633–4648, 2022.

[2] M. W. Gardner and S. Dorling, “Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences,” Atmospheric environment, vol. 32, no. 14-15, pp. 2627–2636, 1998.

[3] S. Ebli, M. Defferrard, and G. Spreemann, “Simplicial neural networks,” arXiv preprint arXiv:2010.03633, 2020.

[4] M. Yang, E. Isufi, and G. Leus, “Simplicial convolutional neural networks,” in ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2022, pp. 8847–8851.

[5] S. Chen, Y. C. Eldar, and L. Zhao, “Graph unrolling networks: Interpretable neural networks for graph signal denoising,” IEEE Transactions on Signal Processing, vol. 69, pp. 3699–3713, 2021.

TU Delft

Thanks for your listening

Chengen Liu

Email: C.Liu-15@tudelft.nl

TU Delft