Graph Filter Design for Distributed Network Processing: A Comparison between Adaptive Algorithms

Atiyeh Alinaghi, Stephan Weiss, Vladimir Stankovic, Ian Proudler

Department of Electronic and Electrical Engineering,
University of Strathclyde, Glasgow, UK

15/09/2021
Sensor Signal Processing for Defence 2021

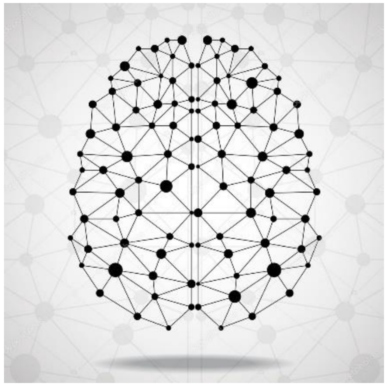
Outline

- Introduction
- Applications
- Graph Signal Processing (GSP)
- Graph Filter (GF) design
- Adaptive algorithms for GF parameter estimation
- Experimental Results
- Conclusion

Introduction

Social media webs

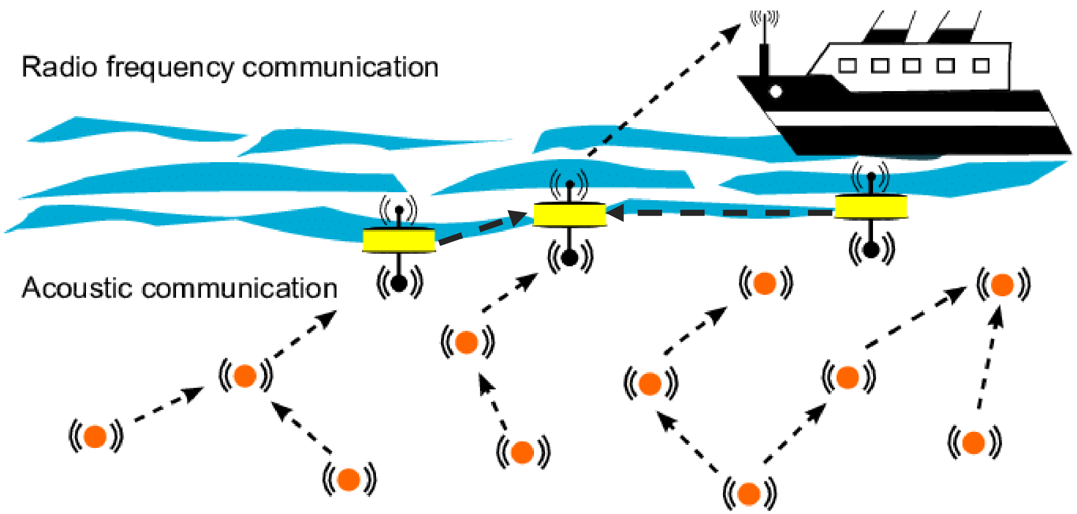
- Graph Signal Processing (GSP)
- Nodes
- Links
- Graph signals
- Graph Filters (GF)
- Graph Fourier Transform (GFT)
- Distributed



Brain activity connections

Applications

- Consider a network of sensors



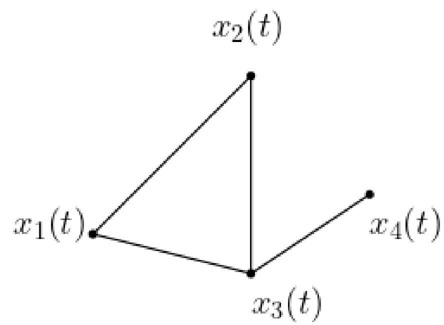
- Sonobuoys
- Wireless sensor networks

- **Centralized** processing:
  - High transmission power
  - Large communication bandwidth
  - Costly energy consumption

- **Distributed** approach:
  - Exchanging the data locally
  - Light processing unit at each node

Graph Signal Processing (GSP)

- Consider a graph G
- N nodes
- Connected via links or edges
- S: Graph Shift Operator (GSO)
- Adjacency matrix A
- Edge weight
- Graph Fourier Transform
- Graph Filter (GF)



$$\mathbf{S} = \mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

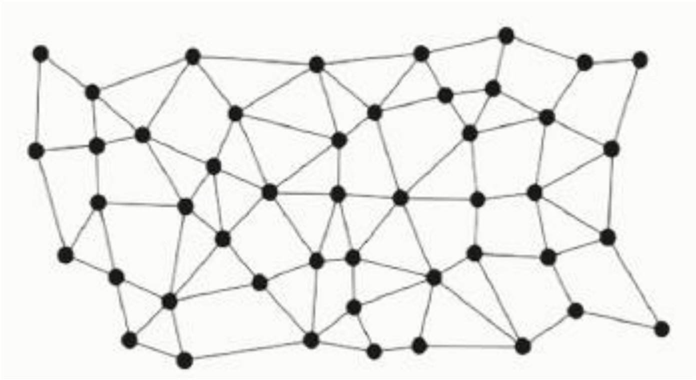
$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_N(t) \end{bmatrix} \qquad \mathbf{y}(t) = \mathbf{T}(\mathbf{x}(t)) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_N(t) \end{bmatrix} \qquad \mathbf{T}(\mathbf{x}) = \mathbf{B}\mathbf{x}$$

Distributed processing

$$\mathbf{y}(t) = \mathbf{T}(\mathbf{x}(t)) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_N(t) \end{bmatrix}$$

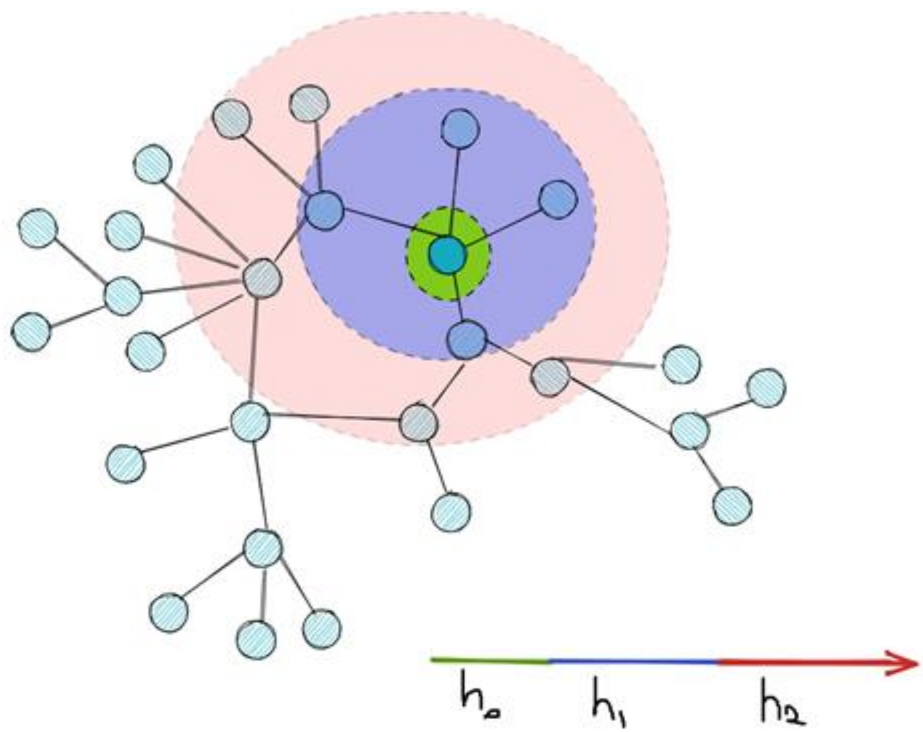
$$\mathbf{T}(\mathbf{x}) = \mathbf{B}\mathbf{x}$$

$$\boxed{\mathbf{B} \approx h_0 + h_1\mathbf{S} + h_2\mathbf{S}^2 + \ldots + h_{M-1}\mathbf{S}^{M-1}}$$



- **S** is the shift matrix of the graph which is given
- Parameters (coefficients) to be estimated: $\{h_m\}_{m=0}^{M-1}$
- **System Identification** in a **distributed manner**
- **Graph Filter (GF)** design

Graph Filter (GF) design



Node-invariant parameters

$$\mathbf{y}(t) = \sum_{m=0}^{M-1} h_m^o \mathbf{S}^m \mathbf{x}(t) + \mathbf{v}(t).$$

Time-invariant input

Graph Filter (GF) design

- (R. Nassif et al. 2018):
  - Time-variant (dynamic) input signals,
  - but node-invariant graph filter (GF) coefficients

- (S. Segarra et al. 2017):
  - Node-variant GF coefficients => more degrees of freedom
  - but time-invariant (static) input signals

- We considered the case with
  - time-variant inputs
  - and node-variant parameters

Graph Filter (GF) design

- Consider a graph with N nodes
- GF with order of M, with M coefficients on each node
- An $M \times N$ coefficients where $h^o_{m,k}$, is the optimal mth coefficient on the kth node
- An N element vector $\boldsymbol{h}^{(m)}$ is the mth coefficients on all the N nodes

$$\mathbf{y}(t) = \sum_{m=0}^{M-1} \underbrace{\mathrm{diag}(\boldsymbol{h}^{(m)})}_{\text{Node-variant parameters}} \mathbf{S}^m \underbrace{\mathbf{x}(t-m)}_{\text{Time-variant inputs}} + \mathbf{v}(t),$$

Graph Filter (GF) design

- Time-invariant input and node-invariant coefficients

$$y_k(t) = \sum_{m=0}^{M-1} h^o_{m,k}[\mathbf{S}^m \mathbf{x}(t-m)]_k + [\mathbf{v}(t)]_k$$

$$\mathbf{z}(t-m) \triangleq \mathbf{S}^m \mathbf{x}(t-m)$$

$$\mathbf{z}_k(t) \triangleq col\{[\mathbf{z}(t)]_k, [\mathbf{z}(t-1)]_k, ..., [\mathbf{z}(t-M+1)]_k\}$$

- In fact $\mathbf{z}_k(t)$ is all the shifted and distributed values of the inputs on the *k*th node from *M-1* hops
- It incorporates the topology of the graph included in **S**
- An $M \times 1$ vector of the coefficients on the $k\mathrm{th}$ node is represented $\boldsymbol{h}^o_k$

$$\boxed{y_k(t) = \mathbf{z}^T_k(t)\boldsymbol{h}^o_k + v_k(t)}$$

System identification on Graphs

- Least-mean-square (LMS) cost function

$$\mathbf{J}_k(\boldsymbol{h}_k) = \mathbb{E}\{|y_k(t) - \mathbf{z}_k^T(t)\boldsymbol{h}_k|^2\}$$

- Recursive-least-square (RLS) cost function

$$\mathbf{J}_k(\boldsymbol{h}_k) = \sum_{i=1}^{t} \lambda^{t-i}|y_k(i) - \mathbf{z}_k^T(i)\boldsymbol{h}_k|^2$$

- Adapt-then-combine (ATC)

$$\begin{cases} \boldsymbol{\psi}_k(t+1) = \boldsymbol{h}_k(t) + \mu_k \mathbf{z}_k(t)(y_k(t) - \mathbf{z}_k^T(t)\boldsymbol{h}_k(t)) \\ \boldsymbol{h}_k(t+1) = \sum_{l\in\mathcal{N}_k} c_{lk}\boldsymbol{\psi}_l(t+1), \end{cases}$$

Adaptive Filters on Graphs

- Combine-recursive-least-square (CRLS)

$$\hat{\mathbf{R}}_{\mathbf{z}_k}(t) = \sum_{i=1}^{t} \lambda^{t-i} \mathbf{z}_k(i)\mathbf{z}_k^T(i) + \lambda^t \delta \mathbf{I}_M$$

$$\mathbf{P}_k(t) \;=\; \hat{\mathbf{R}}_{\mathbf{z}_k}^{-1}(t)$$

$$\mathbf{P}_k(t+1) = \lambda^{-1}\left(\mathbf{P}_k(t) - \frac{\lambda^{-1}\mathbf{P}_k(t)\mathbf{z}_k(t)\mathbf{z}_k^T(t)\mathbf{P}_k(t)}{1+\lambda^{-1}\mathbf{z}_k^T(t)\mathbf{P}_k(t)\mathbf{z}_k(t)}\right)$$

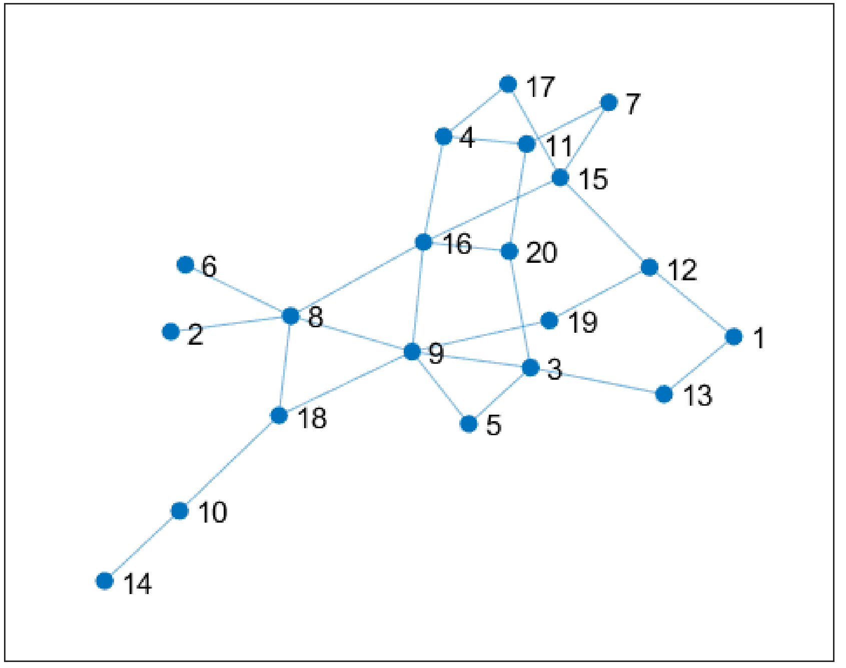
$$\phi_k(t+1) = \boldsymbol{h}_k(t) + \mathbf{P}_k(t)\mathbf{z}_k(t)(y_k(t) - \mathbf{z}_k^T(t)\boldsymbol{h}_k(t)),$$

$$\boxed{\boldsymbol{h}_k(t+1) = \sum_{l\in\mathcal{N}_k} c_{kl}\phi_k(t+1)}.$$

Experimental Results

University of Strathclyde Engineering

- Erdos-Renyi graphs
- Effect of bias in the input signal
- The ground truth GF coefficients $h^o_{m,k}$

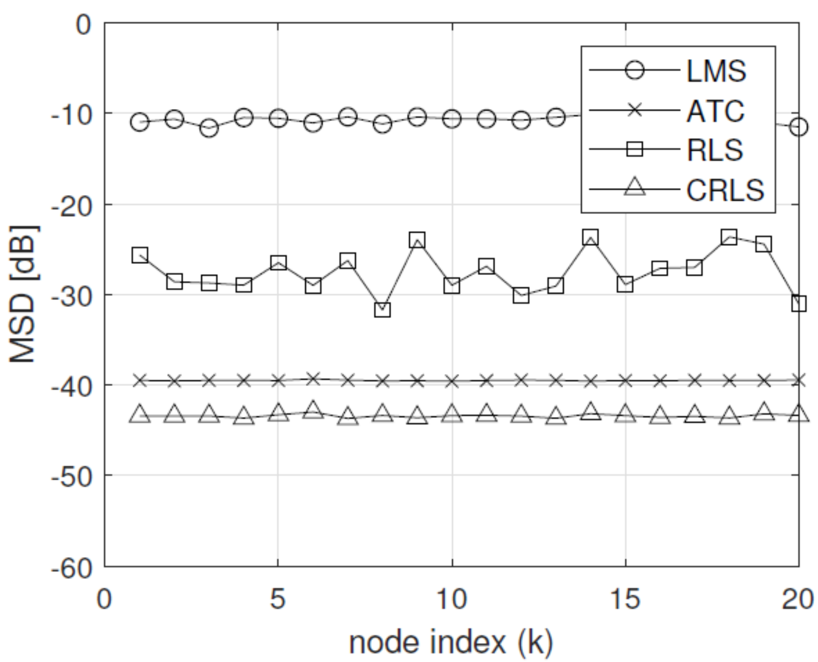


- 200 Monte-Carlo simulations
  - A different random Erdos-Renyi graph for each run
  - A different set of input signals based on the new S
  - A different set of ground truth coefficients at each run
  - Mean-squared-displacement (MSD) at each run

MSD: $\mathbb{E}\{||h^o_k - h_k||^2\}$

Experimental Results

- Mean-squared-displacement (MSD):



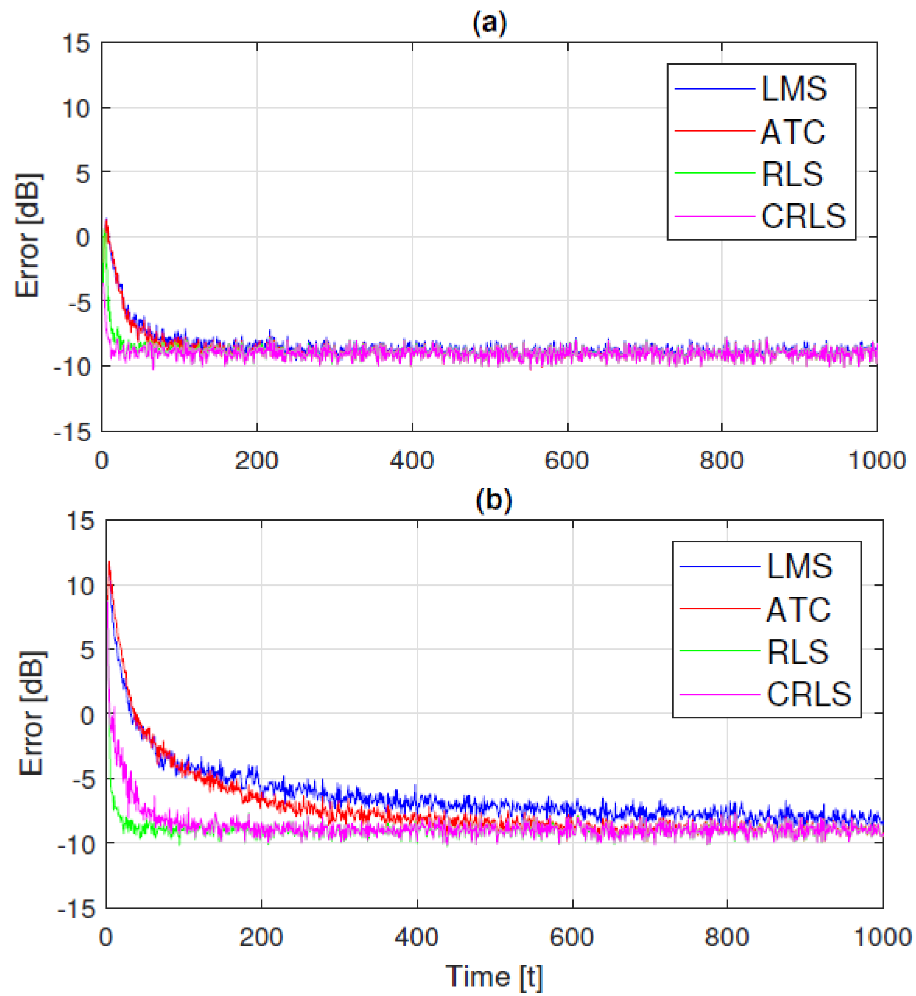
MSD averaged over all coefficients and 200 Monte-Carlo simulations on each node

MSD between the ground-truth and estimations averaged over all the nodes, all the coefficients and 200 Monte-Carlo simulations in [dB]

	node-invariant		node-variant	
	unbiased	biased	unbiased	biased
LMS	-22.72	-15.81	-22.26	-15.76
ATC	-39.39	-42.33	(-10.74)	(-8.78)
RLS	-31.38	-32.60	-30.82	**−32.79**
CRLS	-46.37	**−48.03**	(-10.91)	(-10.89)

Experimental Results

- After each iteration
- On a random node
- With node-invariant coefficients
- (a) Unbiased
- (b) Biased inputs.



Conclusion

- GFs are applied to implement a transformation in a ***distributed*** manner
- A general GF model for both ***time-invariant*** and ***time-varying*** inputs
- Compatible for both ***node-invariant*** and ***node-variant*** coefficients
- Four different adaptive algorithms: LMS, ATC, RLS, CRLS
- Impact of bias in the input signal with severe effect on LMS & ATC
- **RLS** seems to be a better choice with robust performance under different conditions

University of Strathclyde Glasgow