KU LEUVEN

STADIUS
Center for Dynamical Systems, Signal Processing and Data Analytics

A Unified Framework for Distributed Adaptive Spatial Filtering

IEEE SAM 2024

The 13th IEEE Sensor Array and Multichannel Signal Processing Workshop
Corvallis, Oregon, USA, July 8-11, 2024

Cem Ates Musluoglu

European Research Council
Established by the European Commission

Outline

I - Spatial filtering and distributed signal processing

II - Towards a generic meta-algorithm for distributed data-driven spatial filtering
- II.A - Preliminaries
- II.B - The DASF framework

III - Fully-connected DASF algorithm (FC-DASF)
- III.A- Algorithm derivation
- III.B- Technical properties

IV - Topology-independent DASF (TI-DASF)

V - The DASF toolbox

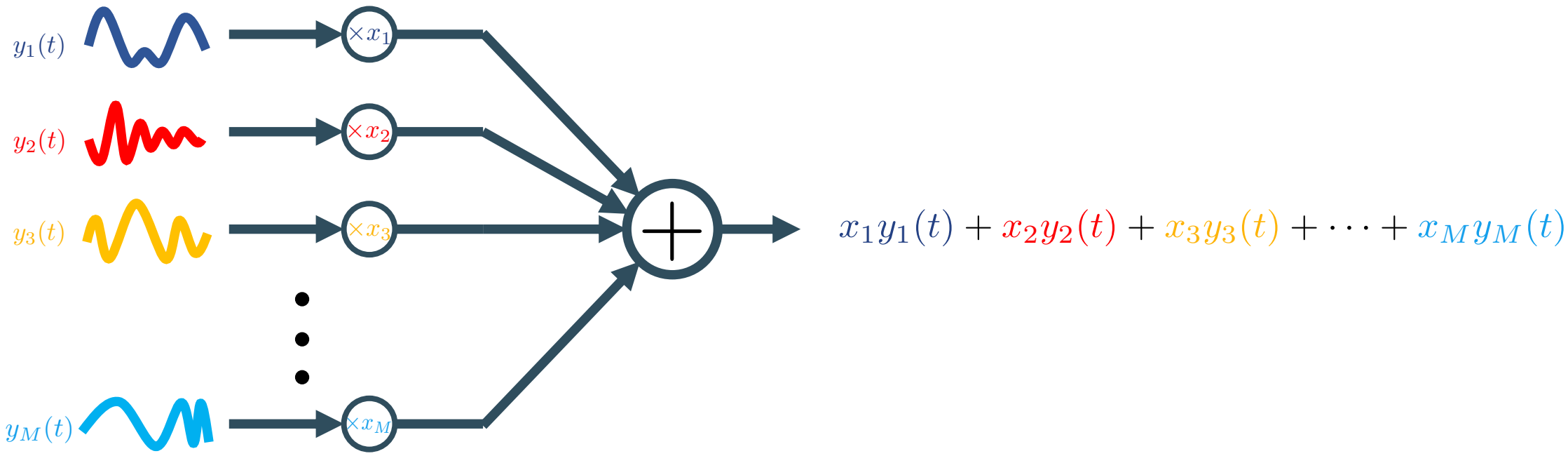
VI - Extensions

I - Spatial filtering and distributed signal processing

Preliminaries

Spatial filtering

- **Spatial filter**: linearly combine channels of a multi-channel signal $\boldsymbol{y}(t) = [y_1(t)\ y_2(t)\ \ldots\ y_M(t)]^T$
  - **MISO**: $z(t) = \boldsymbol{x}^T \boldsymbol{y}(t)$ with $\boldsymbol{x} \in R^M$ → M input channels, 1 output channel



  - **MIMO**: $\boldsymbol{z}(t) = X^T \boldsymbol{y}(t)$ with $X \in R^{M \times Q}$ → M input channels, Q output channels

Data-driven spatial filtering: generic representation (*)

- **Data-driven**: optimize X as a function of the sensor data (statistics)

$$\begin{aligned} \underset{X}{\text{minimize}}\ & f(X^T\mathbf{y}(t)) \\ \text{subject to}\ & g_j(X^T\mathbf{y}(t)) = 0, \\ & h_j(X^T\mathbf{y}(t)) \leq 0 \end{aligned}$$

→ We optimize X, but are generally (more) interested in $\boldsymbol{z}(t) = X^T\boldsymbol{y}(t)$.

Example: minimum mean squared error (MMSE), a.k.a. multi-channel Wiener filter (MWF)

$$\min \mathbb{E}[\|\mathbf{d}(t) - X^T\mathbf{y}(t)\|_F^2]$$

Example: max-SNR filtering

$$\max \frac{\mathbb{E}[\|X^T\mathbf{y}(t)\|_F^2]}{\mathbb{E}[\|X^T\mathbf{n}(t)\|_F^2]}$$

Example: linearly constrained minimum variance (LCMV) beamformer

$$\begin{aligned} &\min \mathbb{E}[\|X^T\mathbf{y}(t)\|_F^2] \\ &\text{s.t. } X^TB = H \end{aligned}$$

Example: principal component analysis (PCA)

$$\begin{aligned} &\max \mathbb{E}[\|X^T\mathbf{y}(t)\|_F^2] \\ &\text{s.t. } X^TX = I \end{aligned}$$

(*) Will assume real-valued signals for simplicity, all results can be extended to complex-valued signals

Data-driven spatial filtering

- $\boldsymbol{y}(t)$ is a stochastic process

$$f(X^T\mathbf{y}(t)) = \mathbb{E}[\Phi(X^T\mathbf{y}(t))]$$

- Typical assumption: $\boldsymbol{y}(t)$ is ergodic and (short-term) stationary

$$\frac{1}{N}\sum_{\tau=t}^{t+N-1}\Phi(X^T\mathbf{y}(\tau)) \approx f(X^T\mathbf{y}(t))$$

- If statistics of $\boldsymbol{y}(t)$ change (slowly) over time → **adaptive** spatial filter

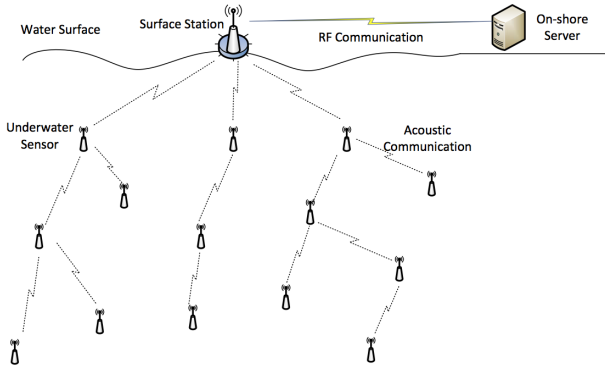
Wireless sensor networks

Wireless Sensor Network:

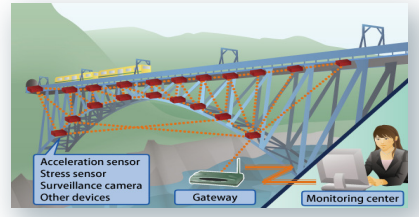
A network of inter-connected devices equipped with **sensing**, **computing** and **communication** capabilities.

Camera networks

Under-water sensing

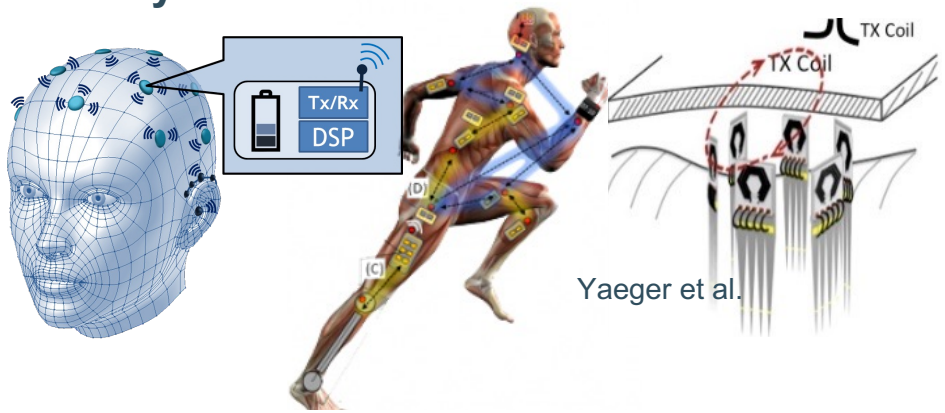


structural health monitoring



Microphone networks (a.k.a. acoustic sensor networks)

Physiological sensor networks & body-area networks



Smart homes & Ambient assisted living

And more…
(radar, environmental monitoring, …)

Distributed signal processing
Motivation

Data centralization in a fusion center:

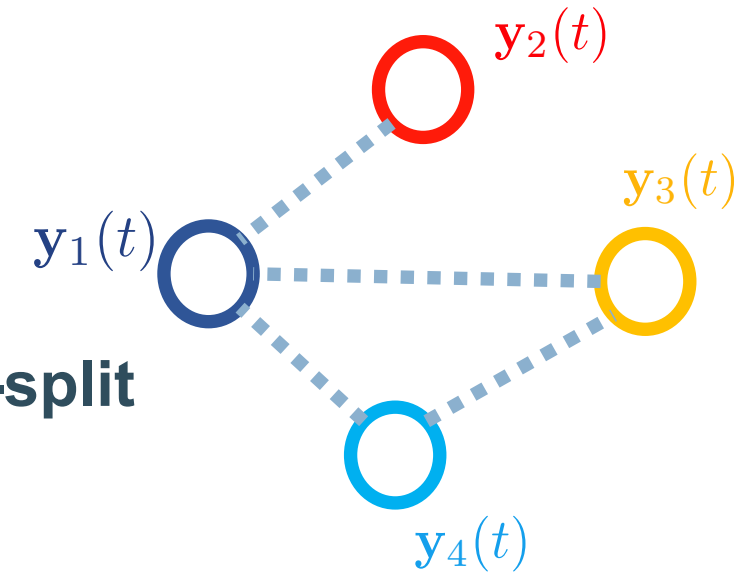
- Simple, use of off-the-shelf SP algorithms
- Wireless transmission of raw sensor data
- Large computational requirement at the fusion center
- Single point of failure

Distributed processing:

No Fusion Center – All nodes equal

- Scalable / compressive data aggregation
- Computational burden is shared
- Robust against node failure
- SP algorithm design is tedious

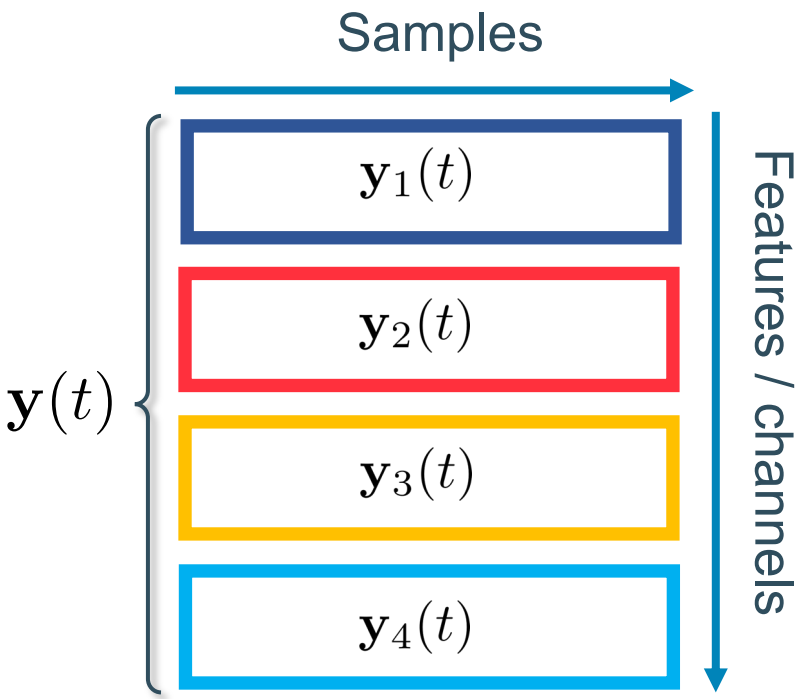
Distributed signal processing
Problem taxonomy

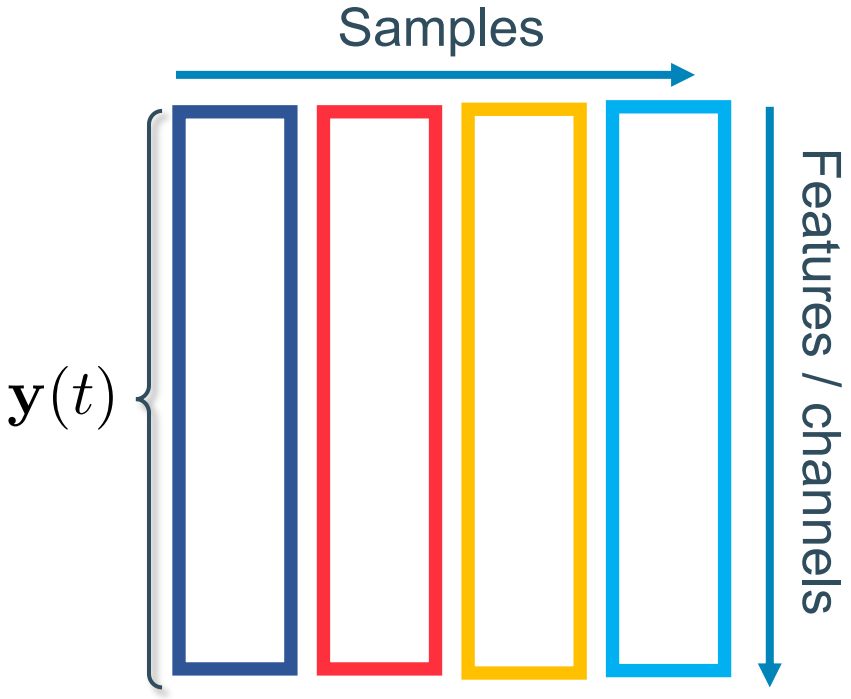


Sample-split

Feature-split

(spatial filtering: features = channels)



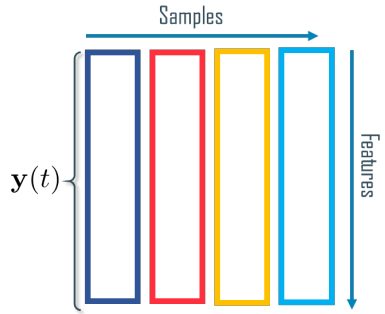


Distributed signal processing
Problem taxonomy

- **Sample-split** typically leads to a **node-separable objective** with a **joint variable** X

$$\min_X \sum_k f_k(X)$$

!! f_k only depends on data from node k !!



- = focus of most distributed SP 'workhorse' algorithms

(consensus, diffusion, ADMM, PDMM, gossip, …)

- Typical strategy:

$$\min_X \sum_k f_k(X_k)$$

$$\text{s.t. } X_i = X_j \ \forall\, i, j$$

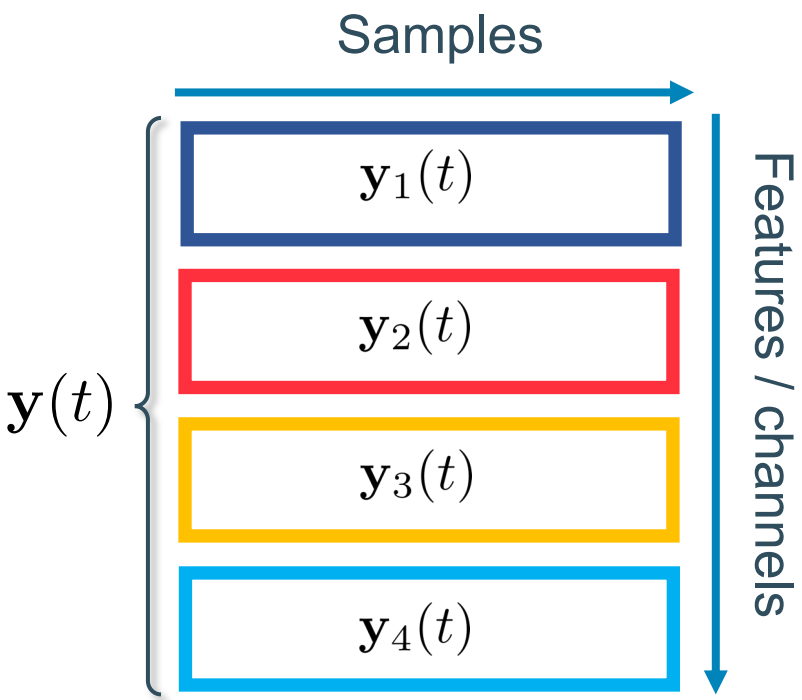
→ Only communicate intermediate estimates X_k^i (usually no sensor data)

Distributed signal processing

Main challenge

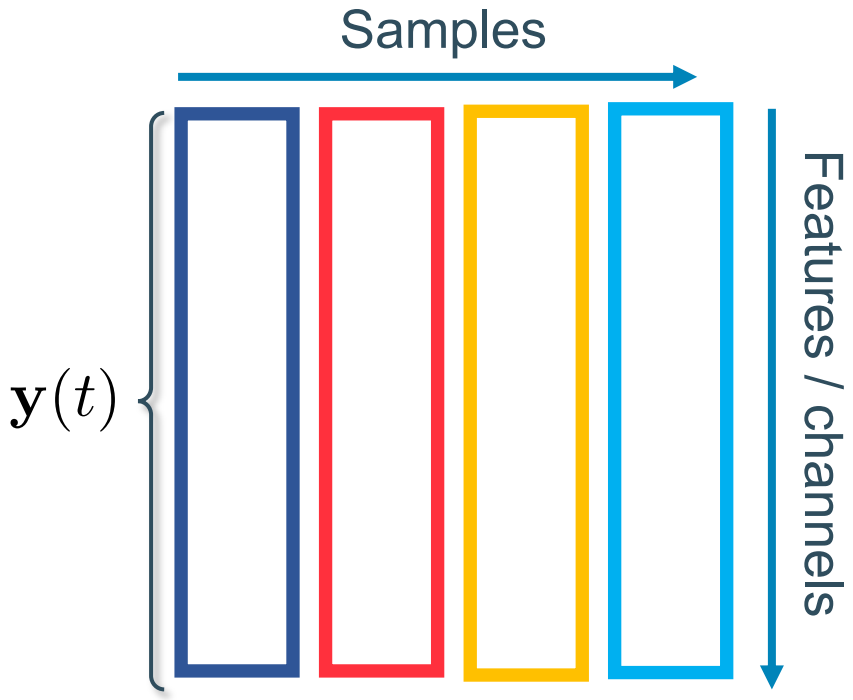
Objective is *not* node-separable

$$\min_X f(X^T \mathbf{y}(t))$$



Node-separable objective

$$\min_X \sum_k f_k(X)$$

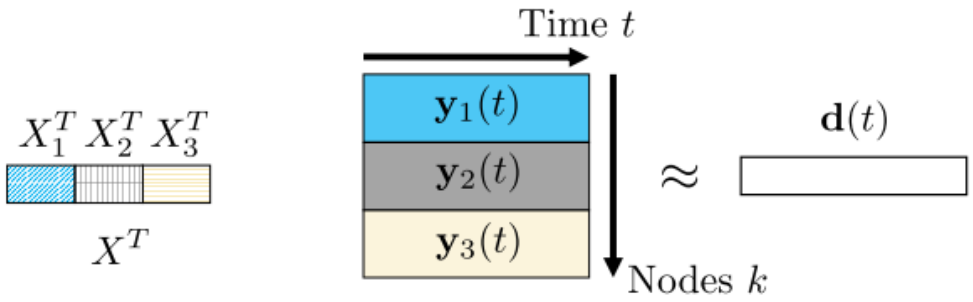


Distributed signal processing
Main challenge

Node-separable argument

MMSE example

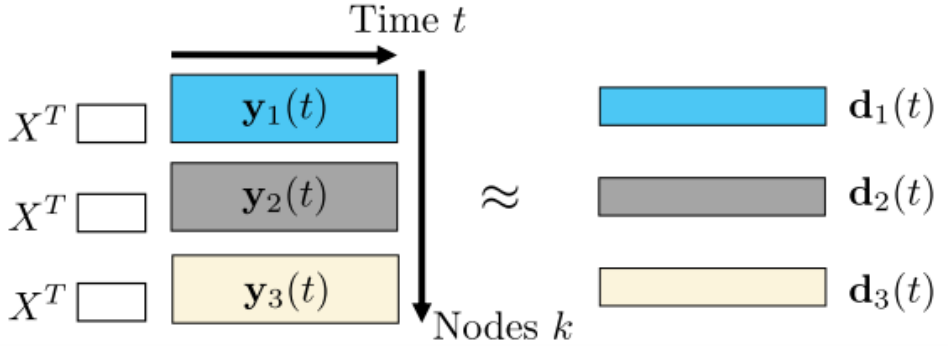
$$\sum_t ||\mathbf{d}(t) - \sum_k X_k^T \mathbf{y}_k(t)||^2$$



Node-separable objective

MMSE example

$$\sum_k \sum_t ||\mathbf{d}_k(t) - X^T \mathbf{y}_k(t)||^2$$



Distributed signal processing
Main challenge

Feature / channel split: how to deal with **inter-node signal covariance**?

$$E\left[\|X^T\mathbf{y}(t)\|_F^2\right] = \mathrm{tr}(X^T E\left[\mathbf{y}(t)\mathbf{y}^T(t)\right] X) = \mathrm{tr}(X^T R_{\mathbf{yy}} X)$$

$$E[\mathbf{y}\mathbf{y}^T] = R_{\mathbf{yy}} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & R_{14} \\ R_{21} & R_{22} & R_{23} & R_{24} \\ R_{31} & R_{32} & R_{33} & R_{34} \\ R_{41} & R_{42} & R_{43} & R_{44} \end{bmatrix}$$

Inter-node covariance

Node-specific covariance

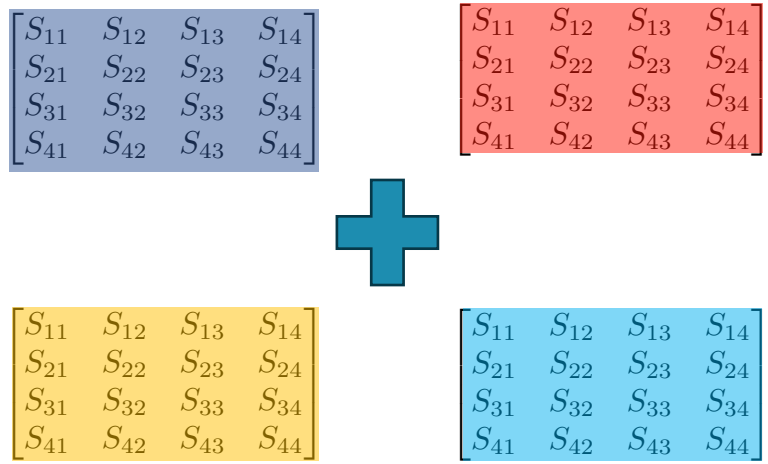
Distributed signal processing
Main challenge

Feature-split
(spatial filtering: features = channels)

$$R_{\mathbf{yy}} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & R_{14} \\ R_{21} & R_{22} & R_{23} & R_{24} \\ R_{31} & R_{32} & R_{33} & R_{34} \\ R_{41} & R_{42} & R_{43} & R_{44} \end{bmatrix}$$

None of the nodes observes the network-wide covariance matrix R_{yy}

Sample-split



- *Each node* observes the **full covariance matrix** R_{yy}
- Aggregated sample covariance can be obtained with an in-network sum

Distributed algorithms for spatial filtering

→ *ad hoc* treatment in literature

IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING, VOL. 5, NO. 4, AUGUST 2011 725

Distributed Principal Subspace Estimation in Wireless Sensor Networks

Lin Li, *Student Member, IEEE*, Anna Scaglione, *Fellow, IEEE*, and Jonathan H. Manton, *Senior Member, IEEE*

260 IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING, VOL. 22, NO. 1, JANUARY 2014

Distributed Delay and Sum Beamformer for Speech Enhancement via Randomized Gossip

Yuan Zeng and Richard C. Hendriks

IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 61, NO. 13, JULY 1, 2013 3447

Distributed LCMV Beamforming in a Wireless Sensor Network With Single-Channel Per-Node Signal Transmission

Alexander Bertrand, *Member, IEEE*, and Marc Moonen, *Fellow, IEEE*

***Abstract*—Linearly constrained minimum variance (LCMV) beamforming is a popular spatial filtering technique for signal estimation or signal enhancement in many different fields. We consider distributed LCMV (D-LCMV) beamforming in wireless sensor networks (WSNs) with either a fully connected or a tree topology. In the D-LCMV beamformer algorithm, each node fuses its multiple sensor signals into a single-channel signal of which observations are then transmitted to other nodes. We envisage an adaptive/time-recursive implementation where each node adapts its local LCMV beamformer coefficients to changes in the local sensor signal statistics, as well as to changes in the statistics of the wirelessly received signals. Although the per-node signal transmission and computational power is greatly reduced compared to a centralized realization, we show that it is possible for each node to generate the centralized LCMV beamformer output as if it had access to all sensor signals in the entire network, without an explicit computation of the network-wide sensor signal covariance matrix. We provide sufficient conditions for convergence and optimality of the D-LCMV beamformer. The theoretical results are validated by means of Monte Carlo simulations, which demonstrate the performance of the D-LCMV beamformer.**

***Index Terms*—Distributed beamforming, distributed signal estimation, LCMV beamforming, signal enhancement, wireless sensor networks (WSNs).**

I. Introduction

A. Distributed Signal Estimation in Wireless Sensor Networks

A wireless sensor network (WSN) consists of a collection of sensor nodes that are connected with each other through wireless links. Each node is equipped with one or more sensors and has computing capabilities for local signal processing. The sensor nodes collect observations of a physical phenomenon and collaborate with each other to perform a certain signal processing task, e.g., localization, detection or estimation of certain signals or parameters. Some approaches require a so-called fusion center (e.g., [1]–[6]) that gathers all the sensor signals, whereas other algorithms are distributed such that all processing happens inside the network (e.g., [7]–[23]). The latter is usually preferred, especially when it is scalable in terms of communication bandwidth and processing power.

Most of the WSN literature focuses on distributed *parameter* estimation (DPE), where a parameter vector with fixed dimension is iteratively estimated by letting nodes exchange intermediate local estimates (see e.g., [7]–[12]). However, in this paper we focus on distributed *signal* estimation (DSE) or signal enhancement, which relies on in-network signal fusion based on spatial filtering or beamforming techniques [4], [5], [14]–[27]. Rather than performing an iterative estimation of each individual sample of the desired signal, DSE algorithms iteratively improve the in-network fusion rules in a time-recursive fashion. DSE algorithms typically operate at higher data rates (compared to DPE algorithms) and often require specific network topologies such as, e.g., fully connected, star, or tree topologies to avoid feedback in the signal fusion paths [1]–[5], [14], [15], [18], [25], [28]. In such topology-controlled networks, the true benefit of a distributed implementation then lies in the in-network fusion/compression of the collected sensor data, i.e., nodes exchange only single-channel (scalar) signal observations instead of multi-channel (vector) signal observations. Even in small-scale networks, this can yield an important reduction in communication bandwidth, in particular for data-intensive tasks such as, e.g., audio processing [20]–[22], [24], [27].

If all raw sensor signals are gathered in a fusion center, traditional (centralized) beamforming techniques can be used to obtain an enhanced signal, based on the estimated covariance between all possible sensor signal pairs. Distributed (in-network) beamforming, on the other hand, is a more challenging problem since each node only has access to a subset of the sensor signals, such that the full covariance matrix cannot be estimated directly. A training phase could be used to construct this matrix, but this approach heavily affects adaptability and flexibility, even in slowly varying scenarios. Therefore, distributed beamforming is often based on suboptimal heuristics to maintain this adaptability and flexibility, and its performance then often heavily depends on the chosen network hierarchy or topology [17]–[19],

Manuscript received December 21, 2012; revised April 10, 2013; accepted April 12, 2013. Date of publication April 23, 2013; date of current version June 10, 2013. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Yimin D. Zhang. The work of A. Bertrand was supported by a Postdoctoral Fellowship of the Research Foundation—Flanders (FWO). This work was carried out at the ESAT Laboratory of KU Leuven, in the frame of KU Leuven Research Council CoE EF/05/006 "Optimization in Engineering" (OPTEC) and PFV/10/002 (OPTEC), Concerted Research Action GOA-MaNet, the Belgian Programme on Interuniversity Attraction Poles initiated by the Belgian Federal Science Policy Office IUAP P6/04 (DYSCO, "Dynamical systems, control and optimization", 2007-2011), Research Project iMinds, and Research Project FWO nr. G.0600.08 ("Signal processing and network design for wireless acoustic sensor networks"). The scientific responsibility is assumed by its authors.

The authors are with the KU Leuven, Department of Electrical Engineering-ESAT, SCD-SISTA, and also with the iMinds Future Health Department, B-3001 Leuven, Belgium (e-mail: alexander.bertrand@esat.kuleuven.be; marc.moonen@esat.kuleuven.be).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2013.2259486

Efficient and Distributed Algorithms for Large-Scale Generalized Canonical Correlations Analysis

Xiao Fu†, Kejun Huang†, Evangelos E. Papalexakis*, Hyun-Ah Song‡,

SIGNAL PROCESSING, VOL. 107, PP. 4-20, FEB. 2015 2

Optimal distributed minimum-variance beamforming approaches for speech enhancement in wireless acoustic sensor networks

Shmulik Markovich-Golan*, Alexander Bertrand†, Marc Moonen†, Sharon Gannot*

*Bar-Ilan University, Faculty of Engineering

Distributed Basis Pursuit

João F. C. Mota, João M. F. Xavier, Pedro M. Q. Aguiar, and Markus Püschel

***Abstract*—We propose a distributed algorithm for solving the optimization problem Basis Pursuit (BP). BP finds the least ℓ_1-norm solution of the underdetermined linear system $Ax = b$ and is used, for example, in compressed sensing for reconstruction. Our algorithm solves BP on a distributed platform such as a sensor network, and is designed to minimize the communication between nodes. The algorithm only requires the network to be connected, has no notion of a central processing node, and no node has access to the entire matrix A at any time. We consider two scenarios in which either the columns or the rows of A are distributed among the compute nodes. Our algorithm, named D-ADMM, is a decentralized implementation of the alternating direction method of multipliers. We show through numerical simulation that our algorithm requires considerably less communications between the nodes than the state-of-the-art algorithms.**

***Index Terms*—Basis pursuit, distributed optimization, sensor networks, augmented Lagrangian**

I. Introduction

Basis Pursuit (BP) is the convex optimization problem [1]

$$\begin{aligned} &\text{minimize} && \|x\|_1 \\ &\text{subject to} && Ax = b, \end{aligned} \tag{BP}$$

where the optimization variable is $x \in \mathbb{R}^n$, $\|x\|_1 = |x_1| + \cdots + |x_n|$ is the ℓ_1 norm of the vector x, and $A \in \mathbb{R}^{m \times n}$ is a matrix with more columns than rows: $m < n$. In words, BP seeks the "smallest" (in the ℓ_1 norm sense) solution of the underdetermined linear system $Ax = b$. To make sure that $Ax = b$ has at least one solution, we require the following.

Assumption 1. *A is full rank.*

BP has recently attracted attention due to its ability to find the sparsest solution of a linear system under certain conditions (see [2], [3]). In particular, BP is a convex relaxation of the combinatorial and nonconvex problem obtained by replacing the ℓ_1 norm in (BP) by the ℓ_0 pseudonorm $\|x\|_0$, which counts the number of nonzero elements of x. Note that the linear system $Ax = b$ has a unique k-sparse solution, i.e., a solution whose ℓ_0 norm is k, if every set of $2k$ columns of A is linearly independent.

João M. F. Xavier, Pedro M. Q. Aguiar, and João F. C. Mota are with Instituto de Sistemas e Robótica (ISR), Instituto Superior Técnico (IST), Technical University of Lisbon, Portugal.

João F. C. Mota is also with the Department of Electrical and Computer Engineering at Carnegie Mellon University, USA.

Markus Püschel is with the Department of Computer Science at ETH Zurich, Switzerland.

This work was supported by the FCT grant CMU-PT/SIA/0026/2009, PTDC/EEA-ACR/73749/2006 and SFRH/BD/33520/2008 (through the Carnegie Mellon/Portugal Program managed by ICTI) from Fundação para a Ciência e Tecnologia and also by ISR/IST pluriannual funding (POSC program, FEDER). This work was also supported by NSF through award 0634967.

Row Partition / Column Partition

$$\begin{bmatrix} A_1 \\ \vdots \\ A_P \end{bmatrix} \qquad \begin{bmatrix} A_1 & A_2 & \cdots & A_P \end{bmatrix}$$

Figure 1. Row partition and column partition of A into P blocks. We assume there are P nodes and the pth node stores A_p. In the row partition a block is a set of rows, while in the column partition a block is a set of columns.

BP belongs to a set of optimization problems that has applications in many areas of engineering. Examples include signal and image denoising and restoration [1], [2], compression, fitting and approximation of functions [4], channel estimation and coding [3] and compressed sensing [5], [6] (for more applications see for example [7], [2] and the references therein). In particular, in the recent field of compressed sensing, BP plays a key role in the reconstruction of a signal.

Notice that Assumption 1 holds with probability one if the entries of A are independent and identically distributed (i.i.d.) and drawn from some (non-degenerate) probability distribution, as commonly seen in compressed sensing [5]. Also in compressed sensing, there are several strategies to deal with noisy observations, i.e., when the observation vector b is corrupted with noise. These include solving variations of (BP), namely BPDN [1] and LASSO [8].

Problem statement and contribution. Consider a network (e.g., a sensor network) with P compute nodes, and partition the matrix A into P blocks. Our goal is to solve BP in a distributed way. By distributed we mean that there is no notion of a central processing node and that the pth node has only access to the block A_p of A during the execution.

We partition A into blocks in two different ways, which we call *row partition* and *column partition*, visualized in Figure 1. In the row partition, the block A_p contains m_p rows of A, which implies $m_1 + \cdots + m_P = m$. In the column partition, A_p contains n_p columns of A, which implies $n_1 + \cdots + n_P = n$.

In summary: *given a network, we solve BP in a distributed way, either in the row partition or in the column partition.*

For the network we only require:

Assumption 2. *The given network is connected and static.*

Connected means that for any two nodes there is a path connecting them. Static means that the network topology does not change over time.

We propose an algorithm to solve this problem and show through extensive simulations that it improves over previous work (discussed below), by reducing the total number of communications to achieve a given solution accuracy. The number of communications in distributed algorithms is an

IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 69, 2021 3653

Distributed Adaptive Trace Ratio Optimization in Wireless Sensor Networks

Cem Ates Musluoglu and Alexander Bertrand, *Senior Member, IEEE*

IEEE SIGNAL PROCESSING LETTERS, VOL. 26, NO. 5, MAY 2019 675

Distributed Rate-Constrained LCMV Beamforming

Jie Zhang, Andreas I. Koutrouvelis, Richard Heusdens, and Richard C. Hendriks

***Abstract*—In this letter, we propose a decentralized framework for rate-distributed linearly constrained minimum variance (LCMV) beamforming in wireless acoustic sensor networks. To save the energy usage within the network, we propose to minimize the transmission cost and put a constraint on the noise reduction** method block-diagonalizes the noise/noisy correlation matrix using linear equality constraints, leading to an efficient distributed implementation for the LCMV beamformer. However, this method does not take into account the quantization noise introduced during the communication between the devices. Nor

4800 IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 63, NO. 18, SEPTEMBER 15, 2015

Distributed Canonical Correlation Analysis in Wireless Sensor Networks With Application to Distributed Blind Source Separation

Alexander Bertrand, *Member, IEEE*, and Marc Moonen, *Fellow, IEEE*

***Abstract*—Canonical correlation analysis (CCA) is a widely used data analysis tool that allows to assess the correlation between two distinct sets of signals. It computes optimal linear combinations of the signals in both sets such that the resulting signals are maximally correlated. The weight vectors defining these optimal linear combinations are referred to as "principal CCA directions". In addition to this particular type of data analysis, CCA is also often used as a blind source separation (BSS) technique, i.e., under certain assumptions, the principal CCA directions have certain demixing properties. In this paper, we propose a distributed CCA (DCCA) algorithm that can operate in wireless sensor networks (WSNs) with a fully connected or a tree topology. The algorithm estimates the Q principal CCA directions from the sensor signal observations collected by the different nodes in the WSN and extracts the corresponding sources. These network-wide principal CCA directions are estimated in a time-recursive fashion without explicitly constructing the corresponding network-wide correlation matrices, i.e., without the need for data centralization. Instead, each node locally computes smaller CCA problems and only transmits compressed sensor signal observations (of dimension Q), which significantly reduces the bit rate over the wireless links of the WSN. We prove convergence and optimality of the DCCA algorithm, and we demonstrate its performance by means of numerical simulations in a blind source separation scenario.**

***Index Terms*—Blind source separation, canonical correlation analysis, distributed estimation, wireless sensor networks (WSNs).**

I. Introduction

Canonical correlation analysis (CCA) is a widely-used data analysis tool to assess the correlation between two distinct sets of data or signals [1], [2]. Basically, it looks for directions in the data with maximal cross-correlation, i.e., it computes optimal linear combinations of the signals in both sets such that the resulting signals are maximally correlated. It is strongly related to partial least squares (PLS) [3], which searches for directions with maximal covariance instead of maximal correlation. Both techniques can be viewed as extensions of principal component analysis (PCA) towards a two-set framework. Further extensions of CCA towards more than two data sets (multi-set CCA or M-CCA), have also been proposed [4], [5], but these are beyond the scope of this paper.

The CCA concept has become a widely-used signal processing (SP) tool, in particular in biomedical signal processing [5]–[13], but also in array processing [14], radar anti-jamming [15], speaker identification [16], SIMO and MIMO equalization [17], [18], and even for analyzing financial data [19]. It is noted that, due to its multi-set framework, CCA is often used for multi-modal signal analysis [7]–[9], [12], [16].

One important SP application of CCA is blind source separation (BSS) [5], [20]–[23]. This CCA-based BSS approach assumes that the hidden sources have a different autocorrelation structure, which is a valid assumption in many applications. For example, CCA-based BSS has been used to remove ocular artifacts [12], muscle artifacts [10], and ballistocardiographic artifacts [7] in electroencephalography (EEG) data. The technique fits in the family of second-order statistics (SOS) BSS techniques [24]–[26], and has a significantly lower computational complexity compared to higher-order statistics BSS techniques, such as independent component analysis [27] and fast implementations thereof [20].

In this paper, we consider the use of CCA to analyze data that is collected by a wireless sensor networks (WSNs), possibly in (slowly-varying) dynamic scenarios where the CCA directions may change over time (e.g., for adaptive BSS). Computing the network-wide CCA requires two matrix inversions and a (generalized) eigenvalue decomposition involving three network-wide correlation matrices that together capture the correlation between every possible sensor signal pair in the WSN. In principle, this would require to forward all the raw sensor signal observations to a fusion center (FC) where the network-wide correlation matrices can then be estimated. However, this usually results in a high bit rate over the wireless links, and it requires a high routing efficiency (in the case of partially-connected networks), and an FC with a large computational power.

Manuscript received March 17, 2015; accepted May 30, 2015. Date of publication June 09, 2015; date of current version August 13, 2015. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Sergios Theodoridis. This work was carried out at the ESAT Laboratory of KU Leuven, in the frame of KU Leuven Research Council BOF/STG-14-005, CoE PFV/10/002 (OPTEC), the Belgian Programme on Interuniversity Attraction Poles initiated by the Belgian Federal Science Policy Office IUAP P7/23 (BESTCOM, 2012-2017), project iMinds Medical IT, Research Projects FWO nr. G.0763.12 "Wireless acoustic sensor networks for extended auditory communication", FWO nr. G.0931.14 "Design of distributed signal processing algorithms and scalable hardware platforms for energy-vs-performance adaptive wireless acoustic sensor networks", and HANDiCAMS. The project HANDiCAMS acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number: 323944. The scientific responsibility is assumed by its authors.

A. Bertrand is with Department of Electrical Engineering (ESAT), Stadius Center for Dynamical Systems, Signal Processing and Data Analytics, KU Leuven, 3001 Leuven, Belgium, and also with iMinds Medical IT (e-mail: alexander.bertrand@esat.kuleuven.be).

M. Moonen is with the Department of Electrical Engineering (ESAT), Stadius Center for Dynamical Systems, Signal Processing and Data Analytics, KU Leuven, 3001 Leuven, Belgium (e-mail: marc.moonen@esat.kuleuven.be).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2015.2443729

ertrand A.

Distributed algorithms for spatial filtering

Common (ad hoc) strategies and hacks

#1: Use centralized algorithm (e.g. stochastic gradient descent) and compute each network-wide inner product via some type of **consensus iteration**

Example: Oja's algorithm for PCA

$$\mathbf{x}^{i+1} = \mathbf{x}^i + \mu\left(\beta^i \mathbf{y}(i) - \left(\beta^i\right)^2 \mathbf{x}^i\right) \text{ with } \beta^i = \mathbf{y}(i)^T \mathbf{x}^i = \sum_k y_k(i) x_k^i$$

Oja's iterative procedure (outer loop iteration)

Consensus iteration for in-network averaging / summation (nested iteration)

Distributed algorithms for spatial filtering

Common (ad hoc) strategies and hacks

#1: Use centralized algorithm (e.g. stochastic gradient descent) and compute each network-wide inner product via some type of **consensus iteration**

1) **communication-intensive**:

   nested iterations (2 time scales)

   Multiple transmissions for each *single* collected sample

2) **does not scale** well with network size

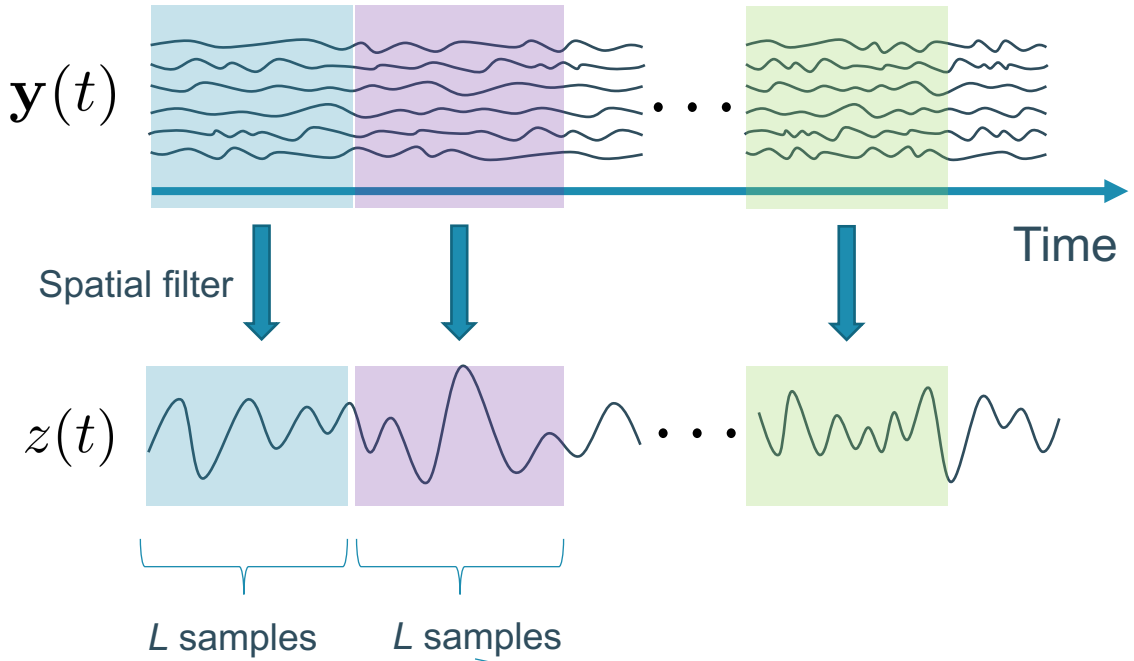
3) strategy not always applicable (depends on chosen centralized algo)

4) often **per-sample transmissions** ($i = t$) → inefficient compared to sample-block transmissions

Distributed algorithms for spatial filtering

Common (ad hoc) strategies and hacks

#2: Treat block of samples of the filter output signal as consensus variable $\boldsymbol{x}$ and run a workhorse distributed algo (consensus, diffusion, ADMM, …) on each separate block



$$\mathbf{x} = [z(0), \ldots, z(L-1)]^T \in \mathbb{R}^L$$

$$\mathbf{x} = [z(L), \ldots, z(2L-1)]^T \in \mathbb{R}^L$$

(start from scratch)

Distributed algorithms for spatial filtering

Common (ad hoc) strategies and hacks

#2: Treat block of samples of the filter output signal as consensus variable x and run a workhorse distributed algorithm (consensus, diffusion, ADMM, …) on each separate block

1) **communication-intensive :**

   - Solving 1 block of L samples = multiple iterations, each one requiring transmission of L-dimensional vector

   - Start from scratch for the next block of L samples

2) **does not scale** well with network size

3) strategy not always applicable (depends on optimization problem)

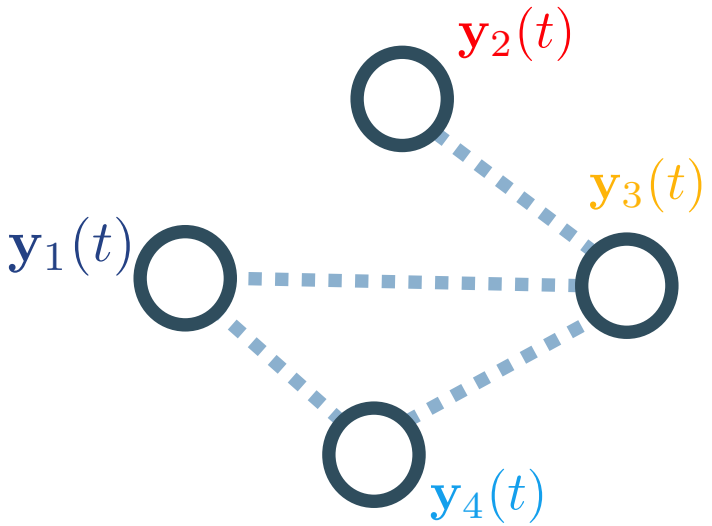
Distributed algorithms for spatial filtering

Common (ad hoc) strategies and hacks

#3: Make (unrealistic) assumptions on covariance matrix, e.g.

- $\boldsymbol{y}(t)$ or noise is spatially white
- $R_{\boldsymbol{yy}}$ has same structure as the communication network

$$R_{\mathbf{yy}} = \begin{bmatrix} R_{11} & 0 & R_{13} & R_{14} \\ 0 & R_{22} & R_{23} & 0 \\ R_{31} & R_{32} & R_{33} & R_{34} \\ R_{41} & 0 & R_{43} & R_{44} \end{bmatrix}$$



Distributed signal processing
Algorithms overview

Algorithm	Exchanged data	Data split	Non-convex problems	Non-smooth problems
Consensus	Opt. variables	Samples		
Diffusion	Opt. variables	Samples		
ADMM or PDMM	Opt. variables	Samples (or Features/channels)		

Problem for spatial filtering
→ requires split in features/channels

Distributed Signal Processing
Algorithms Overview

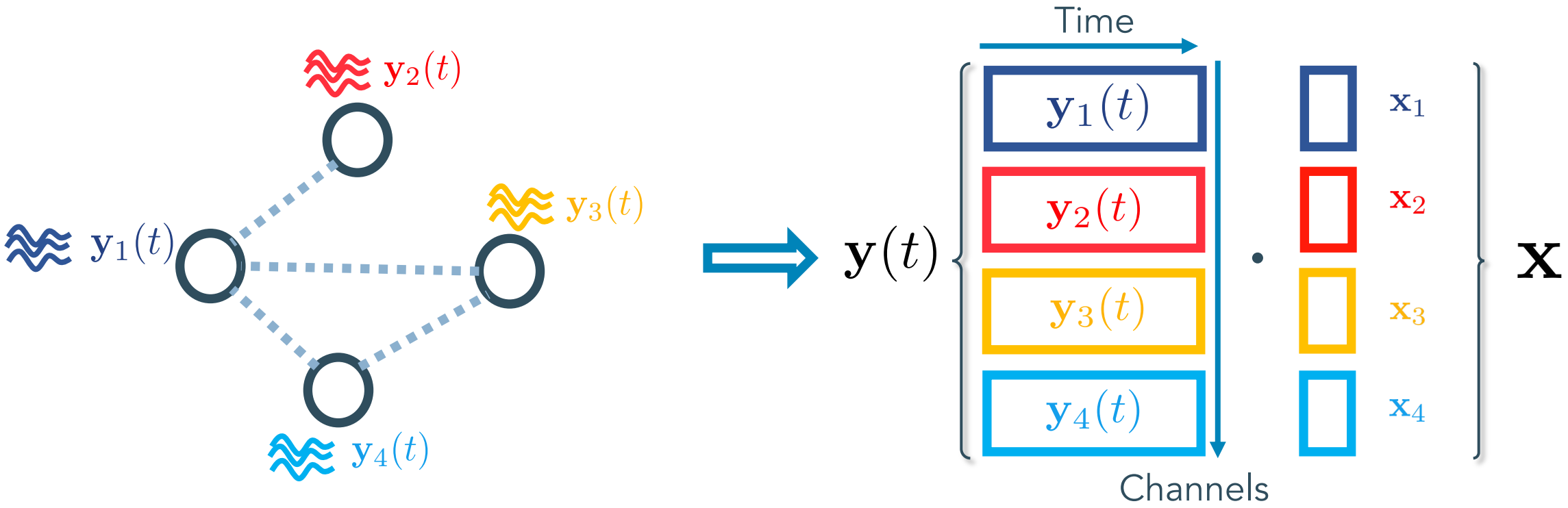
Algorithm	Exchanged data	Data split	Non-convex problems	Non-smooth problems
Consensus	Opt. variables	Samples		
Diffusion	Opt. variables	Samples		
ADMM or PDMM	Opt. variables	Samples (or Features/channels)		
DASF	Signals	Features/channels		

Spoiler Alert !

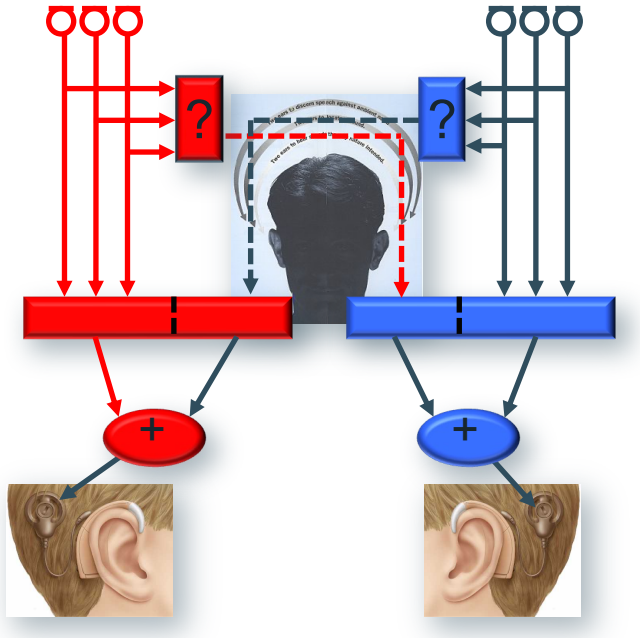
II - Towards a generic meta-algorithm for distributed data-driven spatial filtering

II.A - Preliminaries

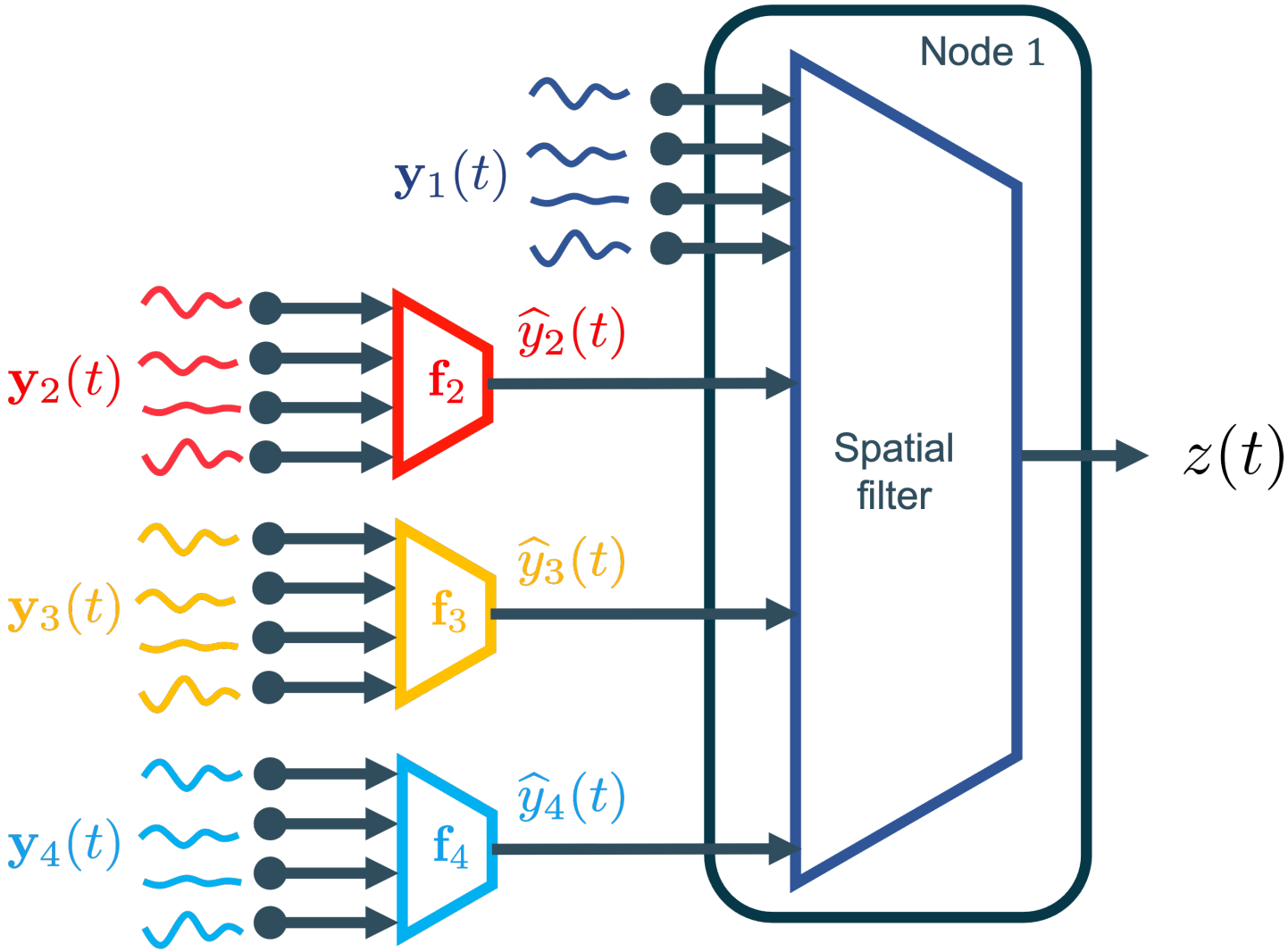
Distributed spatial filtering setting

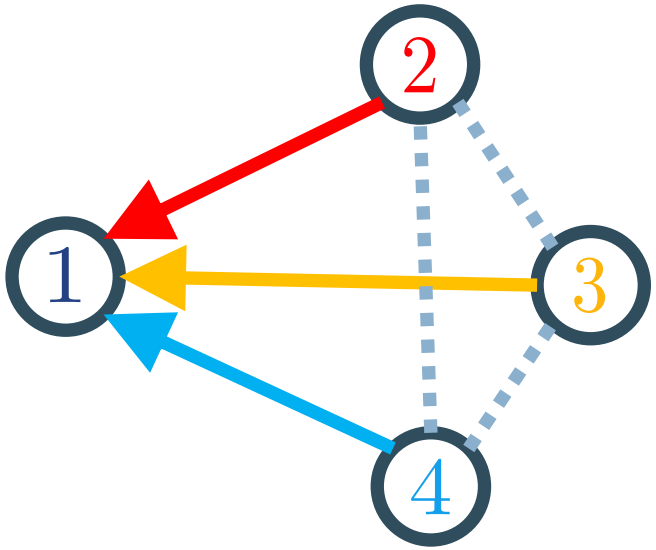


Data flow (2 node example)



Data flow (Fully-connected network)

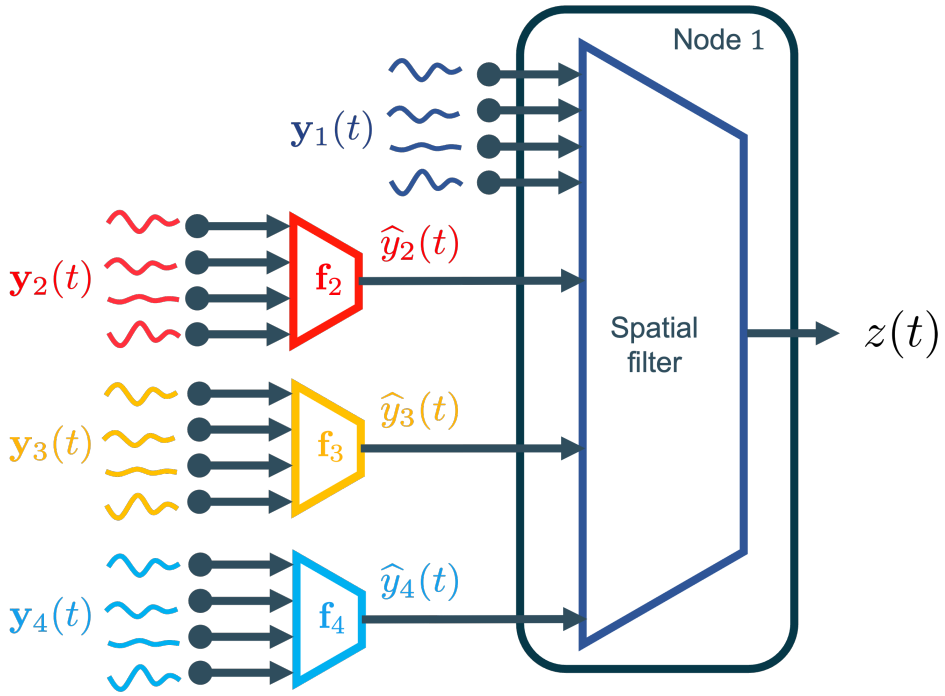




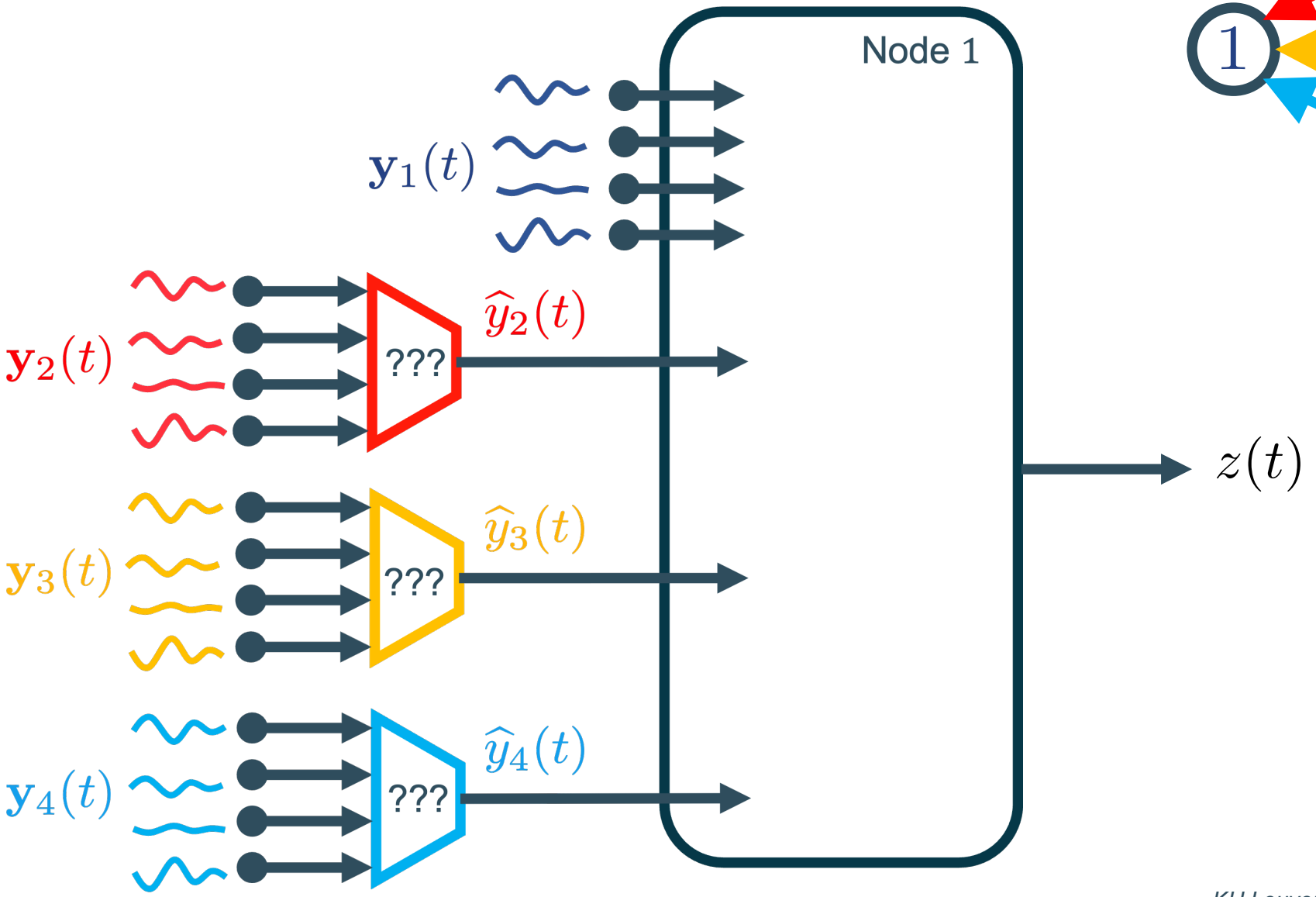
Remember this notation!

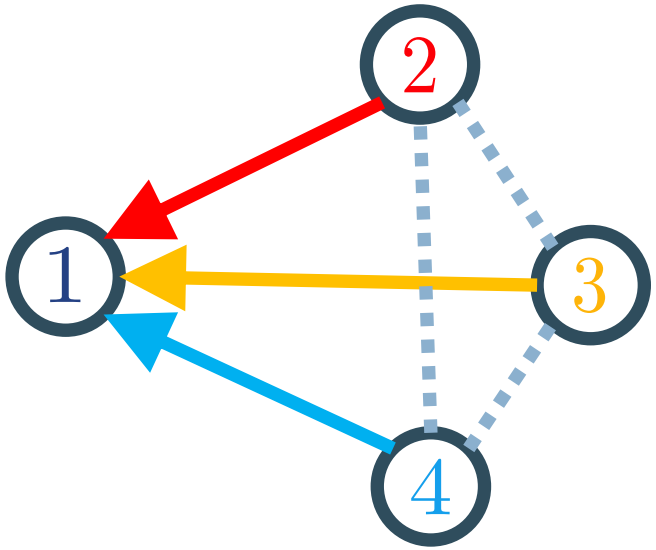
$\boldsymbol{y}_k$ = **original** (raw) signals of node k

$\widehat{\boldsymbol{y}}_k$ = **fused** signals of node k

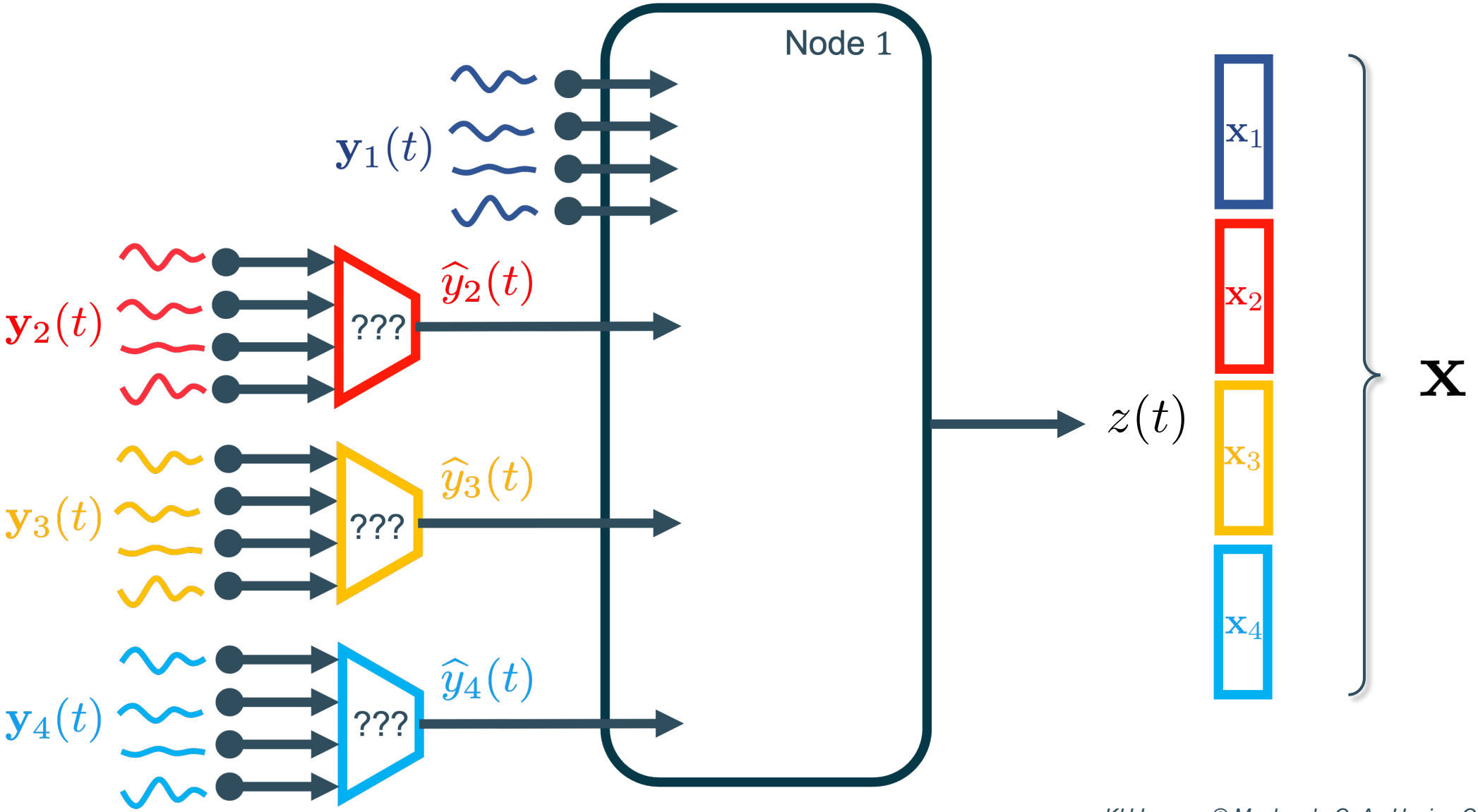


Data flow (Fully-connected network)

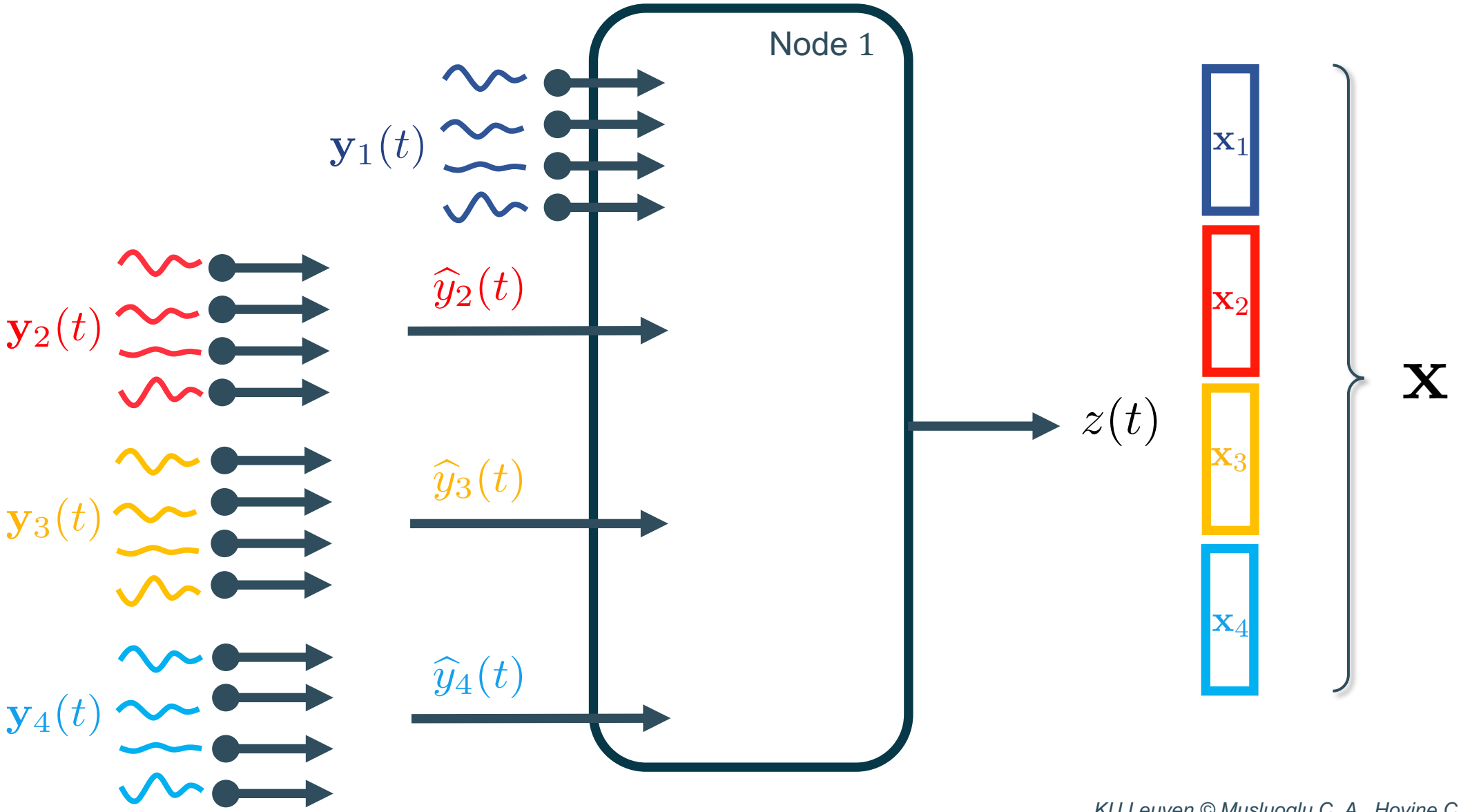




Data flow (Fully-connected network)



Data flow (Fully-connected network)

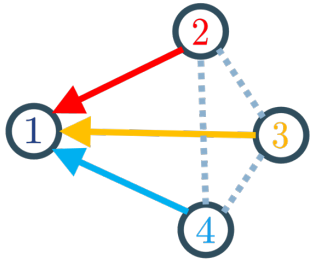


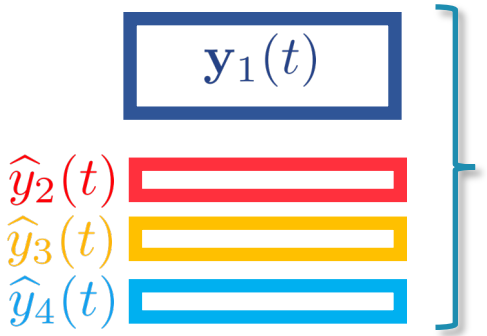
First attempt:
alternating optimization / block-coordinate descent / non-linear Gauss-Seidel

MMSE Example

- Centralized problem: $\min_{\mathbf{x}} \mathbb{E}\left[\left|d(t) - \mathbf{x}^T\mathbf{y}(t)\right|^2\right] = \min_{\mathbf{x}} \mathbb{E}\left[\left|d(t) - \sum_k \mathbf{x}_k^T\mathbf{y}_k(t)\right|^2\right]$

- Local MMSE problem at node 1: $\min_{\mathbf{x}_1} \mathbb{E}\left[\left|d(t) - \mathbf{x}_1^T\mathbf{y}_1(t) - \sum_{k\neq 1} \mathbf{x}_k^T\underbrace{\mathbf{y}_k(t)}_{\widehat{y}_k(t)}\right|^2\right]$

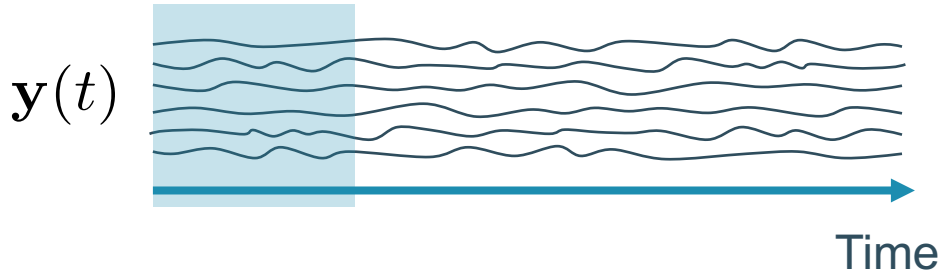


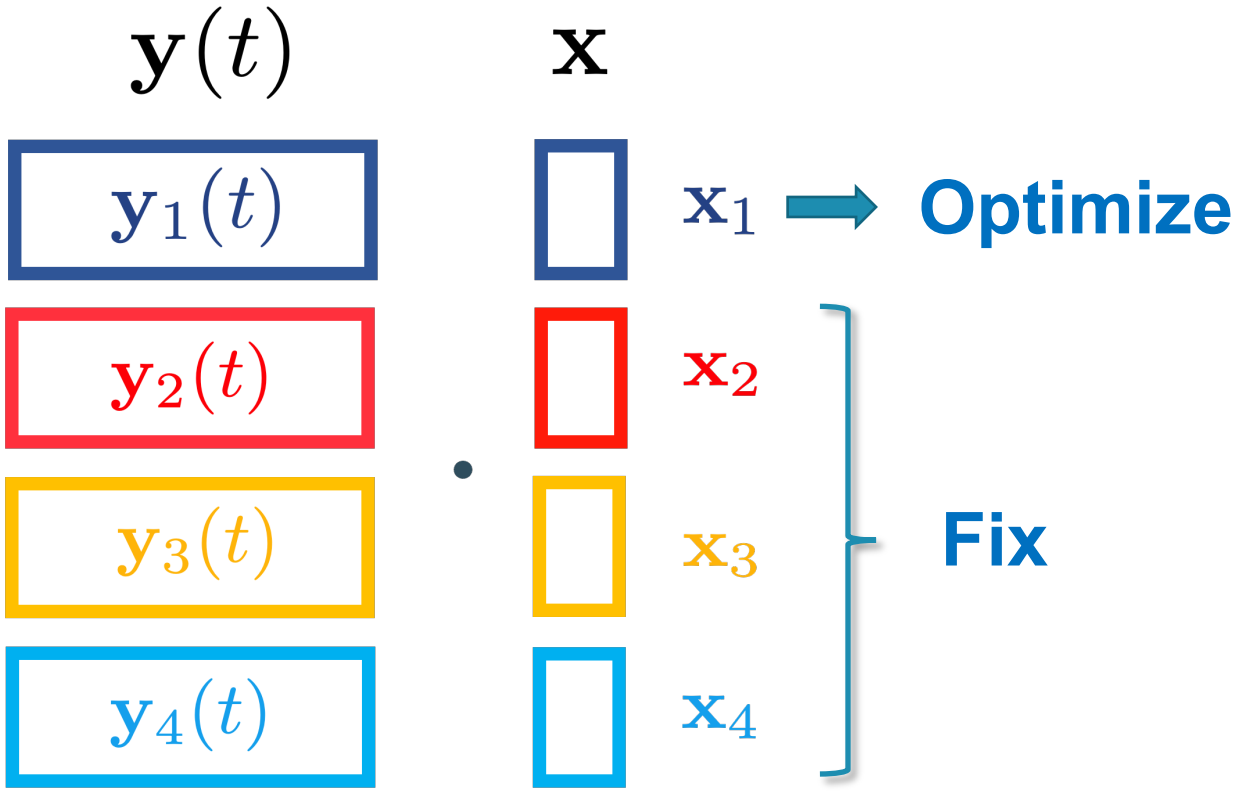


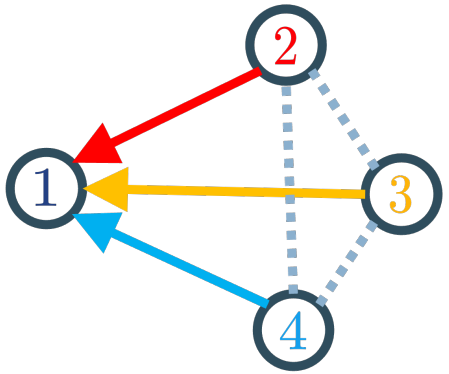
Data available at node 1

First attempt:
alternating optimization / block-coordinate descent / non-linear Gauss-Seidel

- Update in alternating fashion
- Spread iterations over different sample blocks

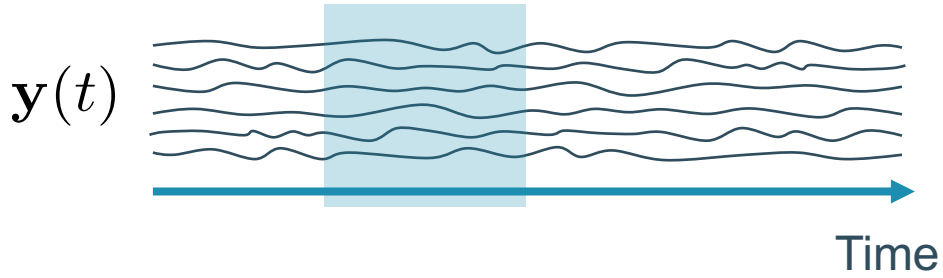


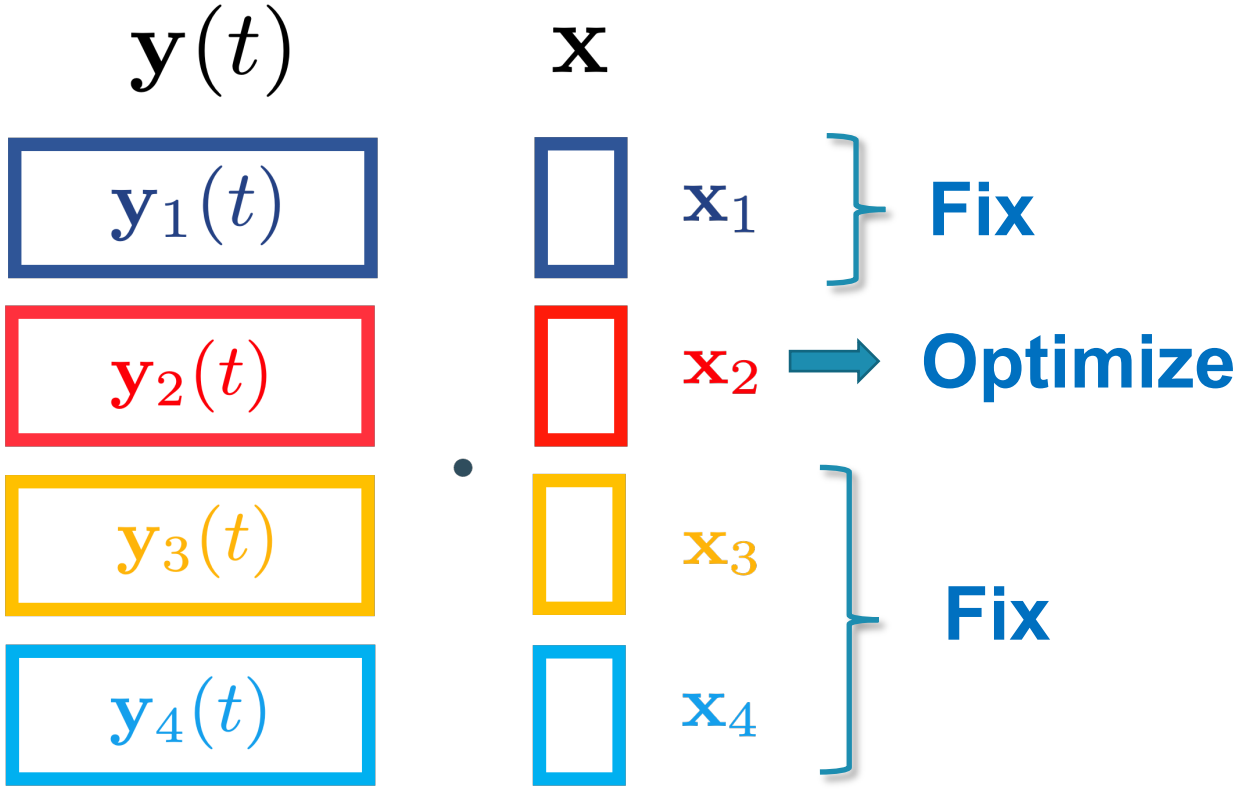


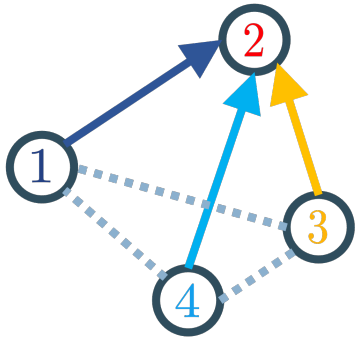


First attempt:
alternating optimization / block-coordinate descent / non-linear Gauss-Seidel

- Update in alternating fashion
- Spread iterations over different sample blocks

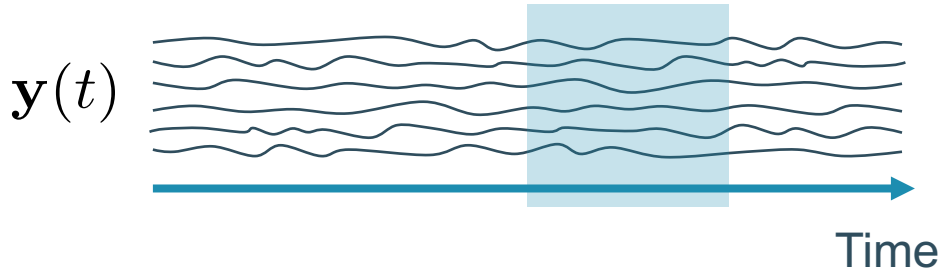


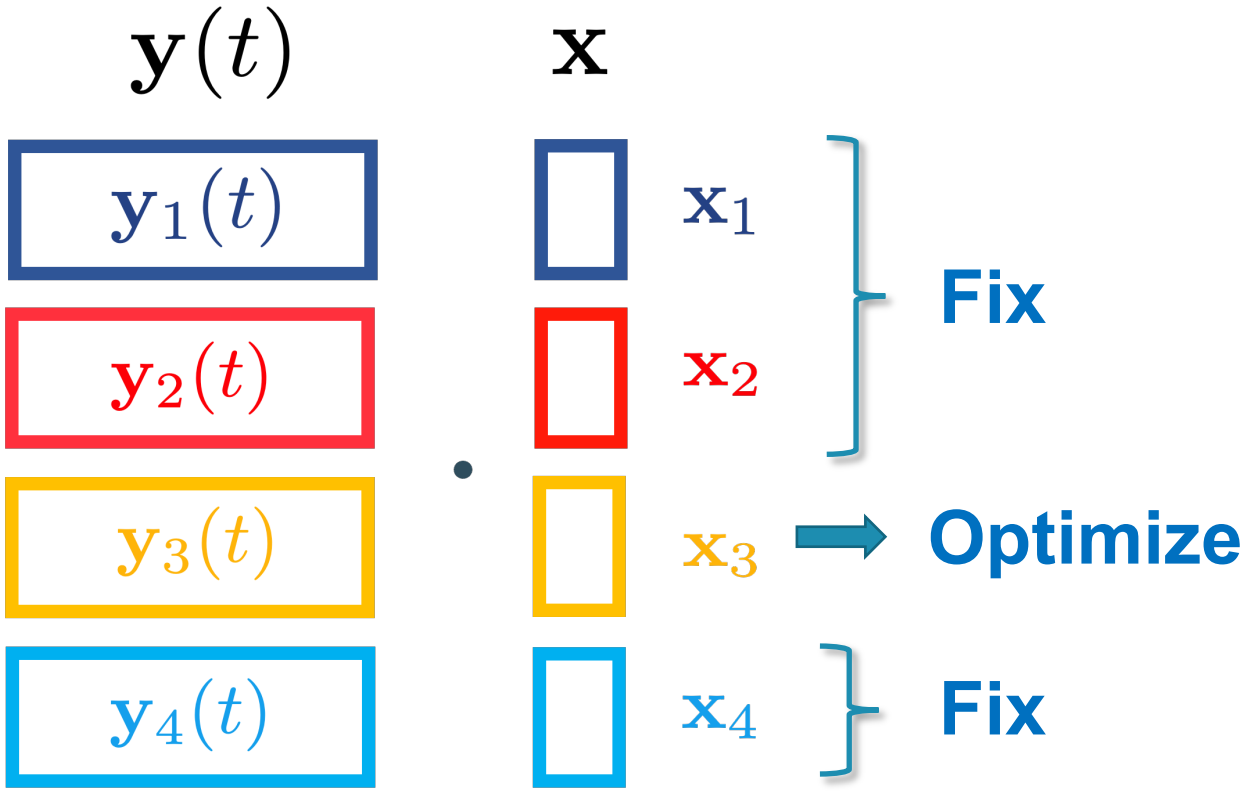


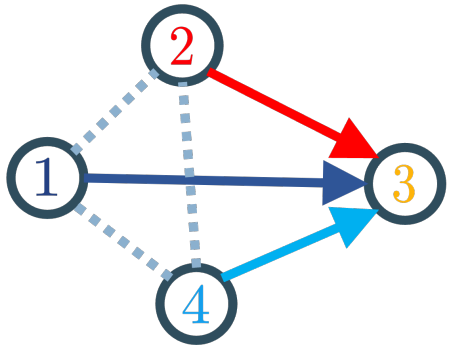


First attempt:
alternating optimization / block-coordinate descent / non-linear Gauss-Seidel

- Update in alternating fashion
- Spread iterations over different sample blocks

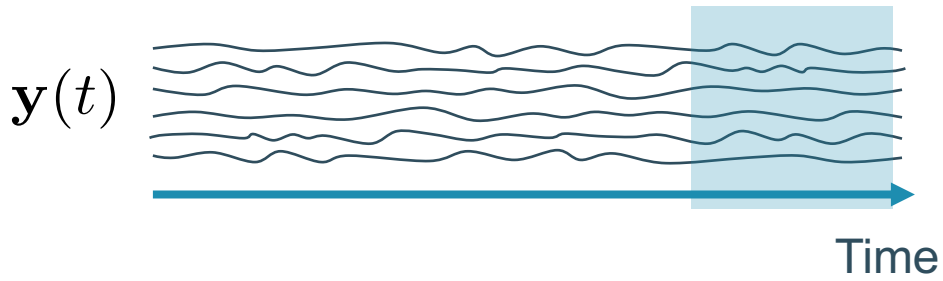


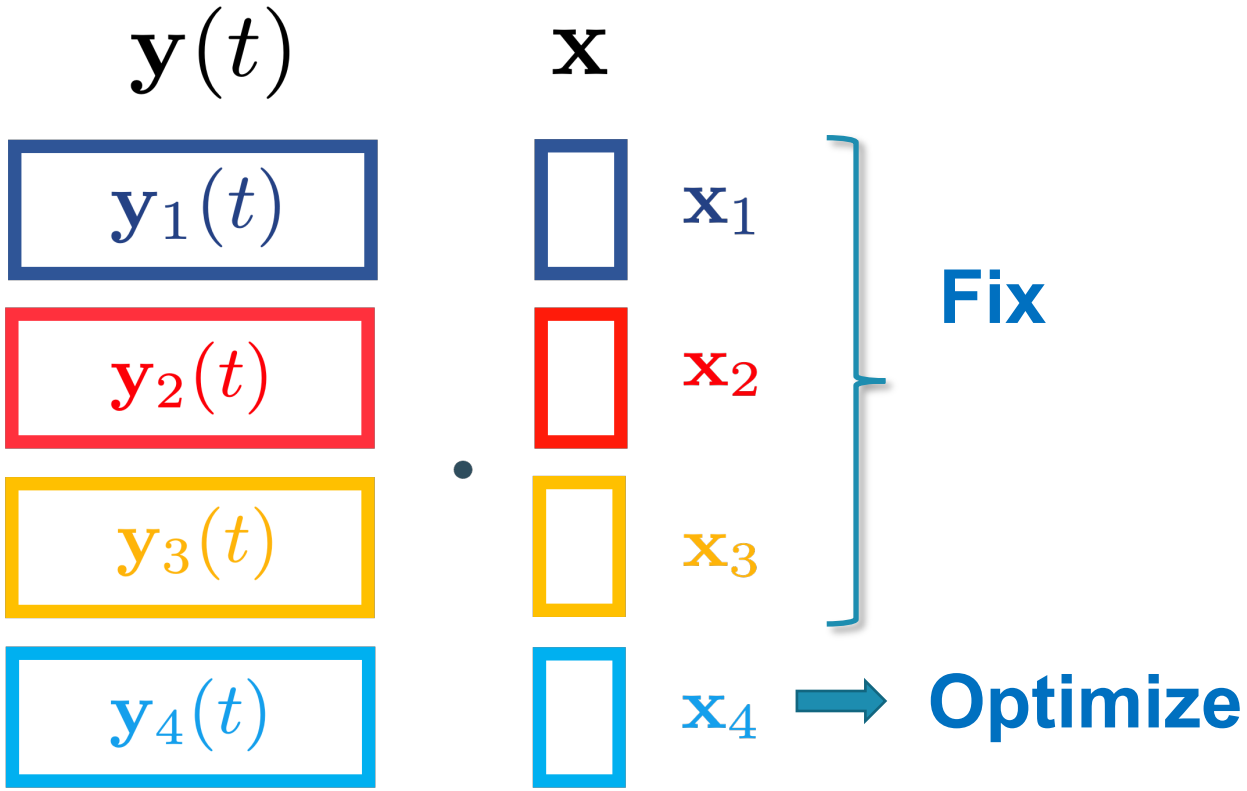


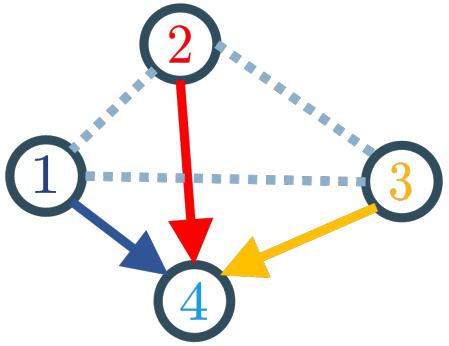


First attempt:
alternating optimization / block-coordinate descent / non-linear Gauss-Seidel

- Update in alternating fashion
- Spread iterations over different sample blocks







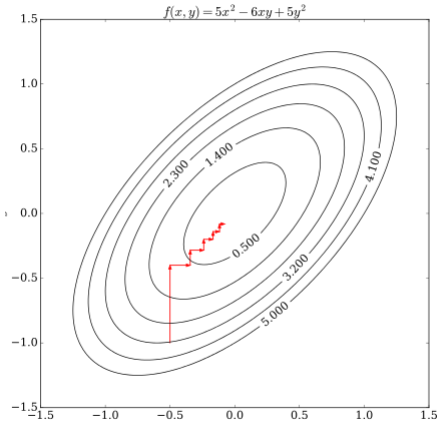
First attempt:
alternating optimization / block-coordinate descent / non-linear Gauss-Seidel

Two problems with this approach:

1) Fixing all x_k's except one

➔ **inefficient** in moving through optimization landscape

➔ effect worsens for large # nodes



2) Local sub-problems generally have a **different structure** than the original problem, and can be hard(er) to solve (see next slide)

First attempt:
alternating optimization / block-coordinate descent / non-linear Gauss-Seidel

PCA Example

$$\max_{\mathbf{x}} \mathbb{E}[|\mathbf{x}^T\mathbf{y}(t)|^2] \quad \text{subject to } \mathbf{x}^T\mathbf{x} = 1$$

Solution is an eigenvalue problem: $R_{\mathbf{yy}}\mathbf{x} = \lambda\mathbf{x}$

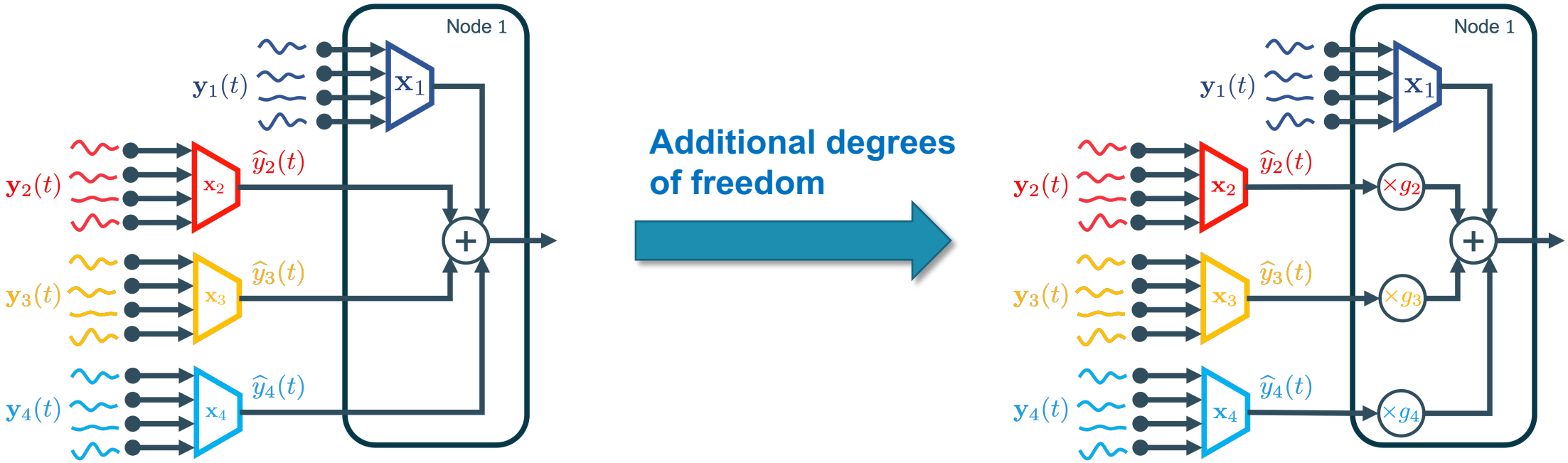
- All $\boldsymbol{x}_k$ are fixed except $\boldsymbol{x}_1$

$$\max_{\mathbf{x}_1} \mathbb{E}[|\mathbf{x}_1^T\mathbf{y}_1(t) + \sum_{k\neq 1}\hat{y}_k(t)|^2] \quad \text{subject to } \mathbf{x}_1^T\mathbf{x}_1 = 1 - \sum_{k\neq 1}\mathbf{x}_k^T\mathbf{x}_k$$

- Solution **not** an eigenvalue problem anymore

$$R_{\mathbf{y}_1\mathbf{y}_1}\mathbf{x}_1 = \lambda\mathbf{x}_1 - \mathbb{E}\left[\mathbf{y}_1(t)\sum_{k\neq 1}\widehat{y}_k(t)\right]$$

Second attempt…

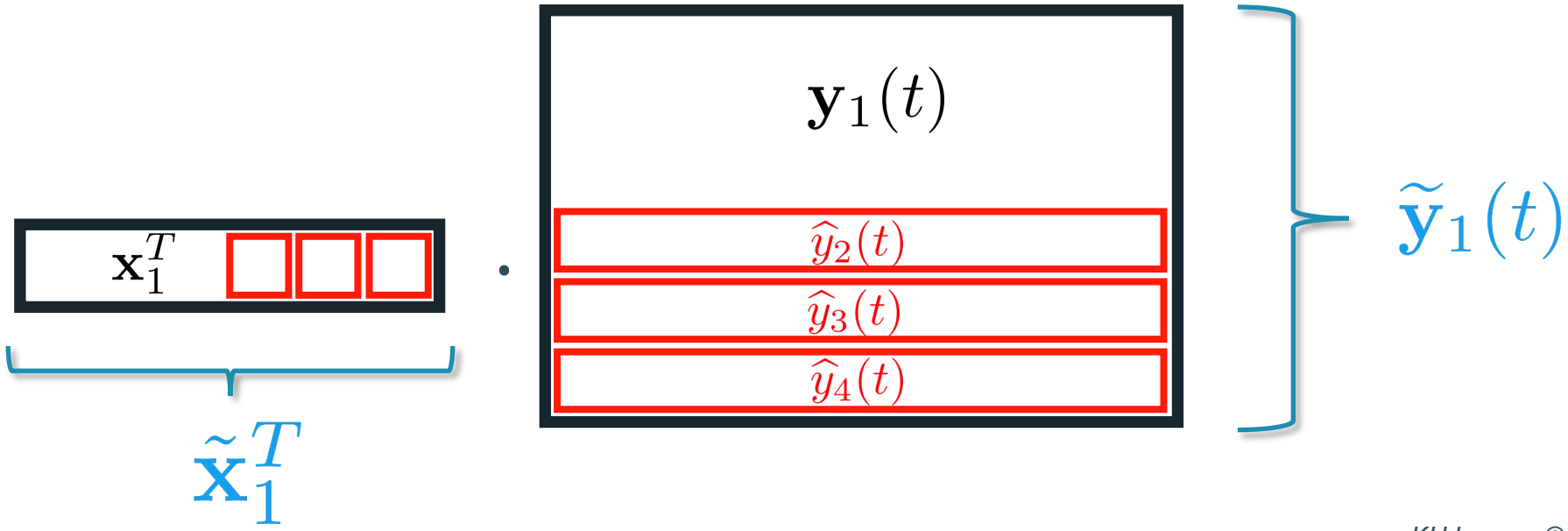


Remember this notation!

$\boldsymbol{\theta}_k$ = **original** (raw) data / variable

$\widehat{\boldsymbol{\theta}}_k$ = **fused** data / variable

$\widetilde{\boldsymbol{\theta}}_k$ = **stack** local $\boldsymbol{\theta}$ with $\widehat{\boldsymbol{\theta}}$ from neighboring node(s)



Revisiting the MMSE example

Local problem at node 1

$$\min_{\tilde{\mathbf{x}}_1} \mathbb{E}[|d(t) - \tilde{\mathbf{x}}_1^T \tilde{\mathbf{y}}_1(t)|^2]$$

$$\Rightarrow \tilde{\mathbf{x}}_1^* = R_{\tilde{\mathbf{y}}_1 \tilde{\mathbf{y}}_1}^{-1} \mathbb{E}\left[\tilde{\mathbf{y}}_1(t) d(t)\right]$$

Original (**centralized**) problem:

$$\min_{\mathbf{x}} \mathbb{E}\left[\left|d(t) - \mathbf{x}^T \mathbf{y}(t)\right|^2\right]$$

$$\Rightarrow \mathbf{x}^* = R_{\mathbf{yy}}^{-1} \mathbb{E}\left[\mathbf{y}(t) d(t)\right]$$

Local problem has same structure as original centralized problem, with smaller dimensions

Formalization:

II.B - The DASF(*) framework

(*) DASF= distributed adaptive signal fusion

(or if you want: distributed adaptive spatial filtering)

The scope of DASF

$$\underset{\mathbf{x}}{\text{minimize}}\ f(\mathbf{x}^T\mathbf{y}(t))$$

Example

LS / MMSE: Signal estimation

$$\underset{\mathbf{x}}{\text{minimize}}\ \mathbb{E}[||d(t) - \mathbf{x}^T\mathbf{y}(t)||]^2$$

The optimization variable x always appears as an inner product with a signal $y(t)$

The scope of DASF

MIMO spatial filters
(Q output channels)

$$\underset{X}{\text{minimize}}\ f(X^T\mathbf{y}(t))$$

$$X = \begin{bmatrix} | & & | \\ \mathbf{x}(1) & \dots & \mathbf{x}(Q) \\ | & & | \end{bmatrix}$$

Example

LS / MMSE: Signal estimation

$$\underset{X}{\text{minimize}}\ \mathbb{E}[||\mathbf{d}(t) - X^T\mathbf{y}(t)||]^2$$

The optimization variable X always appears as an inner product with a signal $\mathbf{y}(t)$

The scope of DASF

Constraints

$$\underset{X}{\text{minimize}}\ f(X^T\mathbf{y}(t))$$
$$\text{subject to}\ g_j(X^T\mathbf{y}(t)) = 0,$$
$$h_j(X^T\mathbf{y}(t)) \leq 0$$

The optimization variable X always appears as an inner product with a signal $\mathbf{y}(t)$

The scope of DASF

Multiple signal sets

$$\underset{X}{\text{minimize}}\ f(X^T\mathbf{y}(t), X^T\mathbf{v}(t))$$
$$\text{subject to}\ g_j(X^T\mathbf{y}(t), X^T\mathbf{v}(t)) = 0,$$
$$h_j(X^T\mathbf{y}(t), X^T\mathbf{v}(t)) \leq 0$$

Example

Max-SNR/GEVD: Dimensionality reduction

$$\underset{X}{\text{maximize}}\ \mathbb{E}[||X^T\mathbf{y}(t)||]^2$$
$$\text{subject to}\ \mathbb{E}[X^T\mathbf{n}(t)\mathbf{n}(t)^T X] = I$$

The optimization variable X always appears as an inner product with a signal $\boldsymbol{y}(t), \boldsymbol{v}(t), \ldots$

The scope of DASF

Multiple variables

$$\underset{(X,W)}{\text{minimize}}\ f(X^T\mathbf{y}(t), W^T\mathbf{v}(t))$$
$$\text{subject to}\ g_j(X^T\mathbf{y}(t), W^T\mathbf{v}(t)) = 0,$$
$$h_j(X^T\mathbf{y}(t), W^T\mathbf{v}(t)) \leq 0$$

Example

CCA: Correlation between data sets

$$\underset{(X,W)}{\text{maximize}}\ \mathbb{E}[\text{tr}(X^T\mathbf{y}(t)\mathbf{v}^T(t)W)]$$
$$\text{subject to}\ \mathbb{E}[X^T\mathbf{y}(t)\mathbf{y}(t)^TX] = I$$
$$\mathbb{E}[W^T\mathbf{v}(t)\mathbf{v}(t)^TW] = I$$

All optimization variables $X, W, \ldots$ always appear as an inner product with a signal $\boldsymbol{y}(t), \boldsymbol{v}(t), \ldots$

The scope of DASF

Deterministic matrices

$$\begin{aligned}\underset{X}{\text{minimize}}\ & f(X^T\mathbf{y}(t), X^TB, X^TC)\\ \text{subject to}\ & g_j(X^T\mathbf{y}(t), X^TB, X^TC) = 0,\\ & h_j(X^T\mathbf{y}(t), X^TB, X^TC) \leq 0\end{aligned}$$

Example

(Robust) minimum variance beamforming

$$\begin{aligned}\underset{X}{\text{minimize}}\ & \mathbb{E}[||X^T\mathbf{y}(t)||]^2\\ \text{subject to}\ & X^TB = H,\\ & \text{tr}(X^TX) \leq \alpha^2\end{aligned}$$

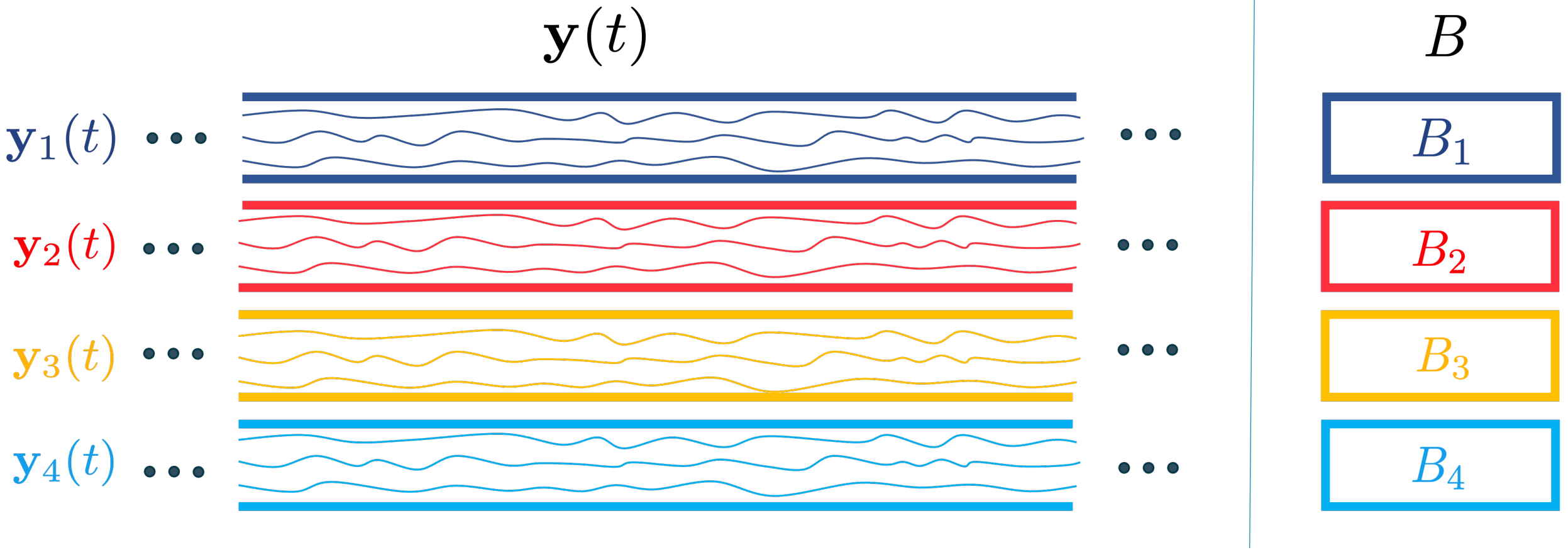
All optimization variables $X, W, \ldots$ always appear as an inner product with either:

a signal $\boldsymbol{y}(t), \boldsymbol{v}(t), \ldots$

… or a deterministic matrix $B, C, \ldots$

The scope of DASF

Deterministic matrices treated similarly as stochastic signal $\boldsymbol{y}$ but are fixed, i.e., not time-dependent



The scope of DASF

$$\underset{X^{(a)},\forall a}{\text{minimize}}\ f\left(X^{(a)T}\mathbf{y}^{(b)}(t), X^{(a)T}B^{(c)}, ...\right),\ \forall a,b,c$$

$$\text{subject to}\ h_j\left(X^{(a)T}\mathbf{y}^{(b)}(t), X^{(a)T}B^{(c)}, ...\right) \leq 0\ \ \forall j \in \mathcal{J}_I,$$

$$h_j\left(X^{(a)T}\mathbf{y}^{(b)}(t), X^{(a)T}B^{(c)}, ...\right) = 0\ \ \forall j \in \mathcal{J}_E.$$

$a \in \mathcal{A}$: Index set of all optimization variables

$b \in \mathcal{B}$: Index set of all signals

$c \in \mathcal{C}$: Index set of all deterministic matrices that interact as in inner product with $X^{(a)}s$

$\mathcal{J}_I$: Index set of all inequality constraints

$\mathcal{J}_E$: Index set of all equality constraints

All optimization variables $X^{(a)}$ always appear as an inner product with either:

a signal $y^{(b)}(t)$

… or a deterministic matrix $B^{(c)}$

… let's keep it simple today

$$\begin{aligned} \underset{X}{\text{minimize}}\ & f(X^T\mathbf{y}(t), X^T B) \\ \text{subject to}\ & g_j(X^T\mathbf{y}(t), X^T B) = 0, \qquad \forall j \in \mathcal{J}_I, \\ & h_j(X^T\mathbf{y}(t), X^T B) \leq 0 \qquad \forall j \in \mathcal{J}_E. \end{aligned}$$

X always appears as an inner product with :

signal $y(t)$

… or deterministic matrix B

A trick to remember

$$\begin{aligned}\underset{X}{\text{minimize}}\ & f(X^T\mathbf{y}(t), X^T B)\\ \text{subject to}\ & g_j(X^T\mathbf{y}(t), X^T B) = 0,\\ & h_j(X^T\mathbf{y}(t), X^T B) \leq 0\end{aligned}$$

If X appears alone or in an inner product with itself, e.g.,

$$\underset{X}{\text{minimize}}\ \|X\|_F$$

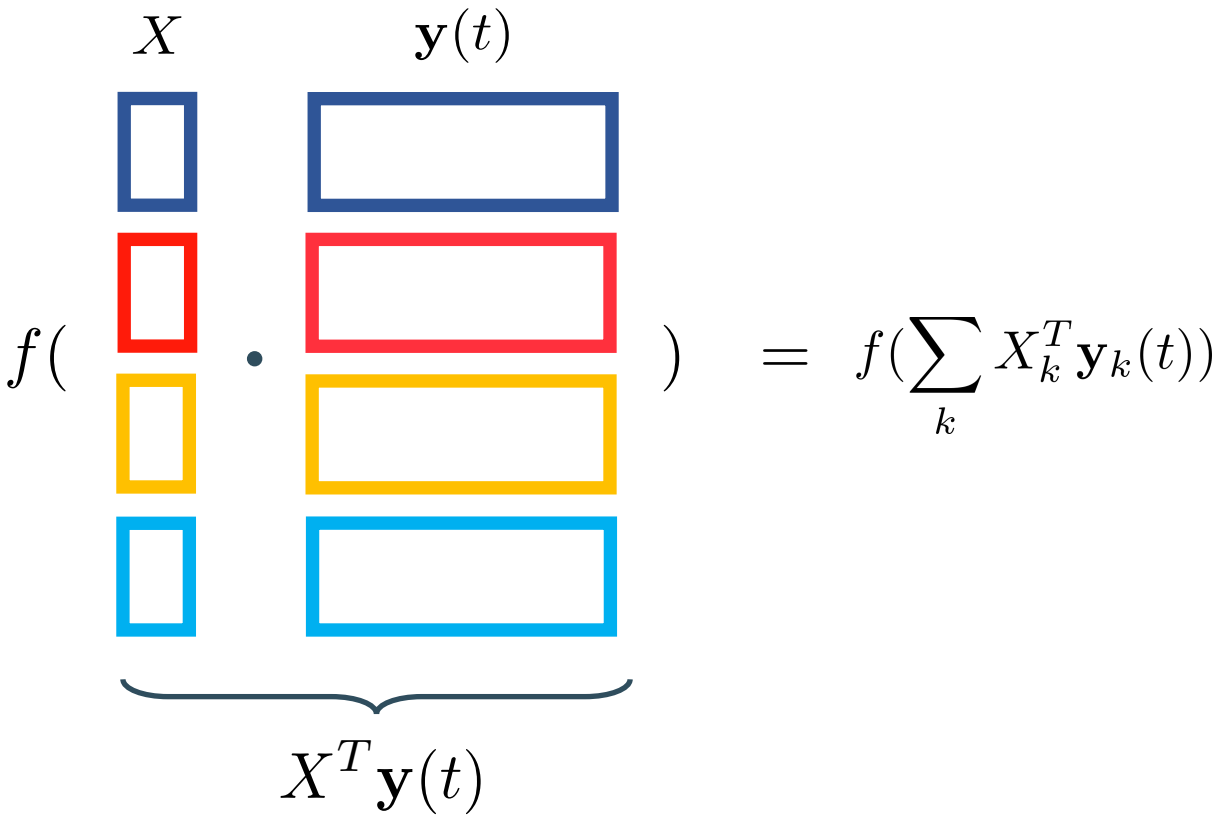
$$X^T X = (X^T \cdot I)(X^T \cdot I)^T = (X^T \cdot B)(X^T \cdot B)^T$$

(with $B = I$)

$$\underset{X}{\text{minimize}}\ \|X^T B\|_F$$

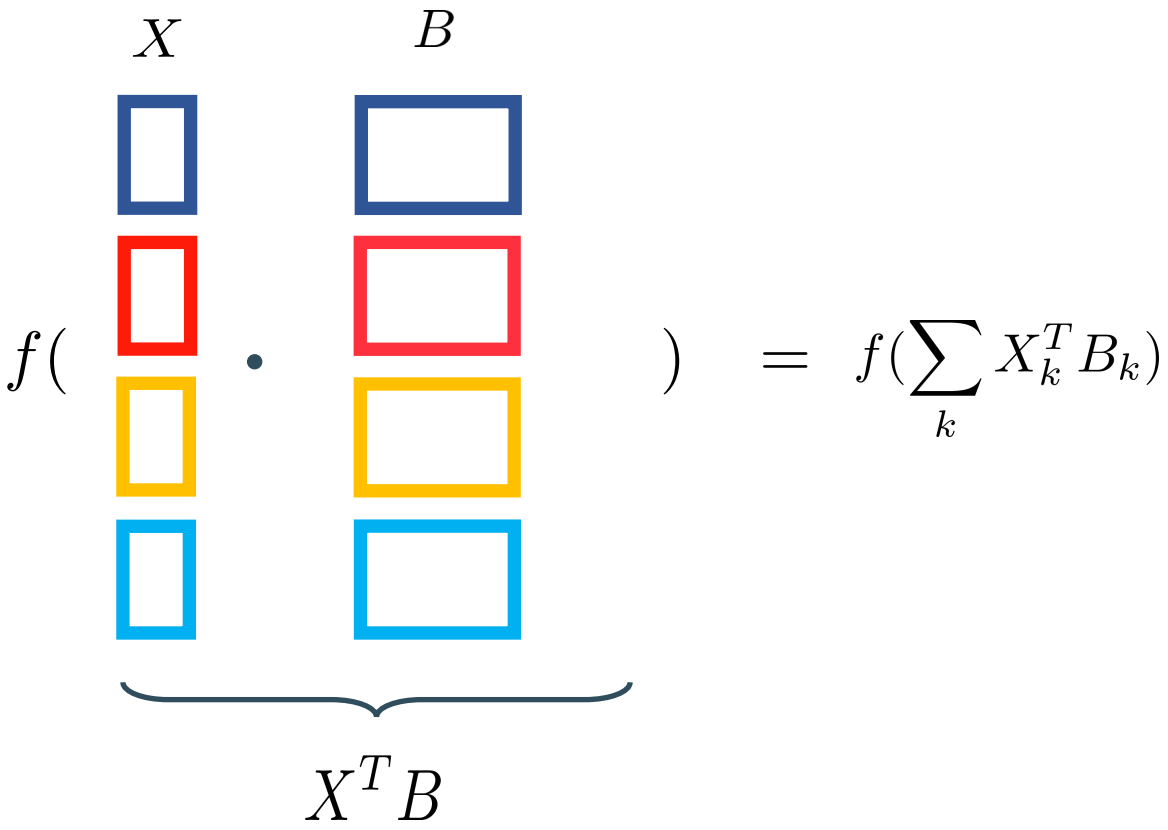
Functions of sums

Loss and constraints only depend on X through $X^T \boldsymbol{y}(t)$ and possibly on $X^T B$

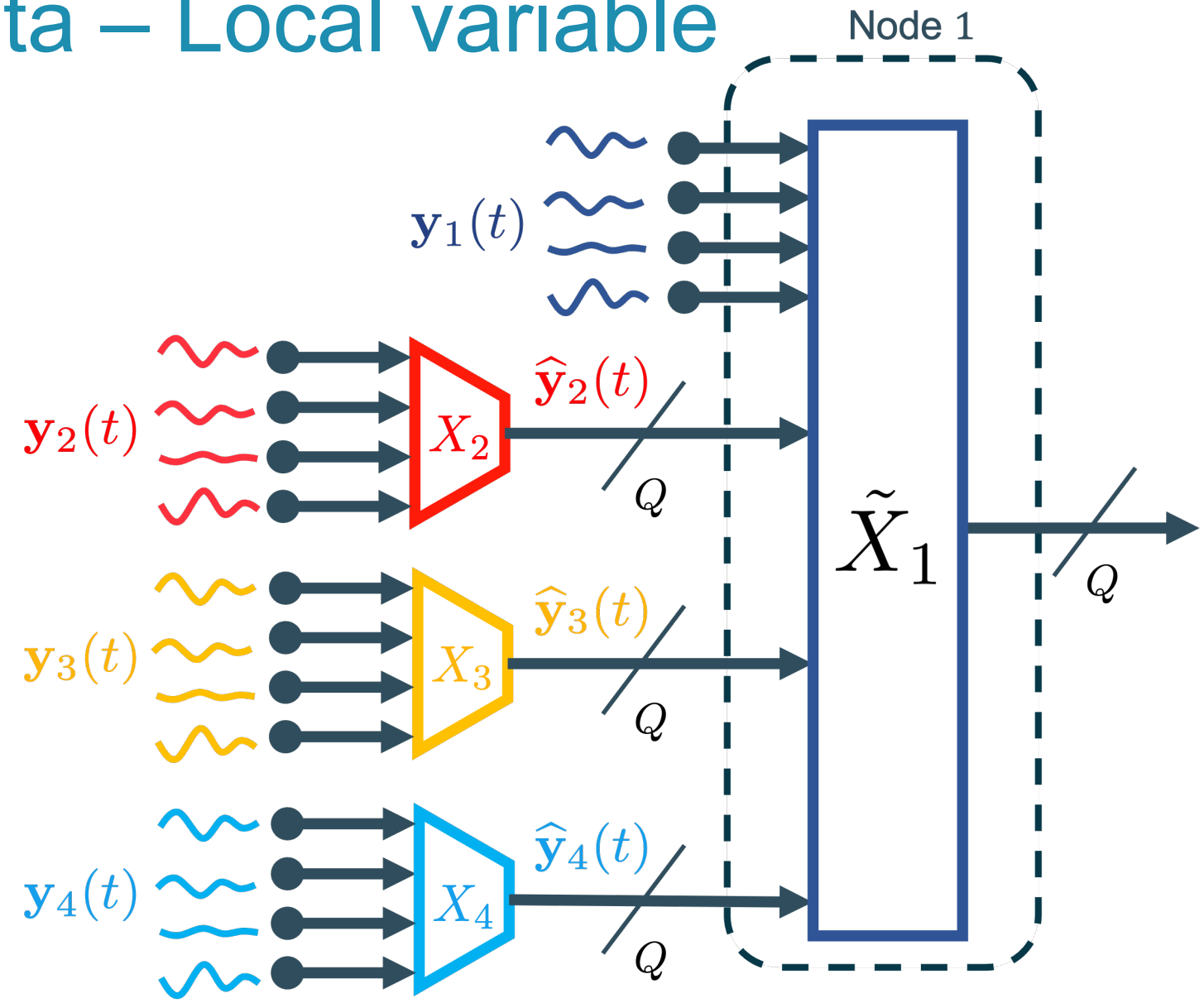


Functions of sums

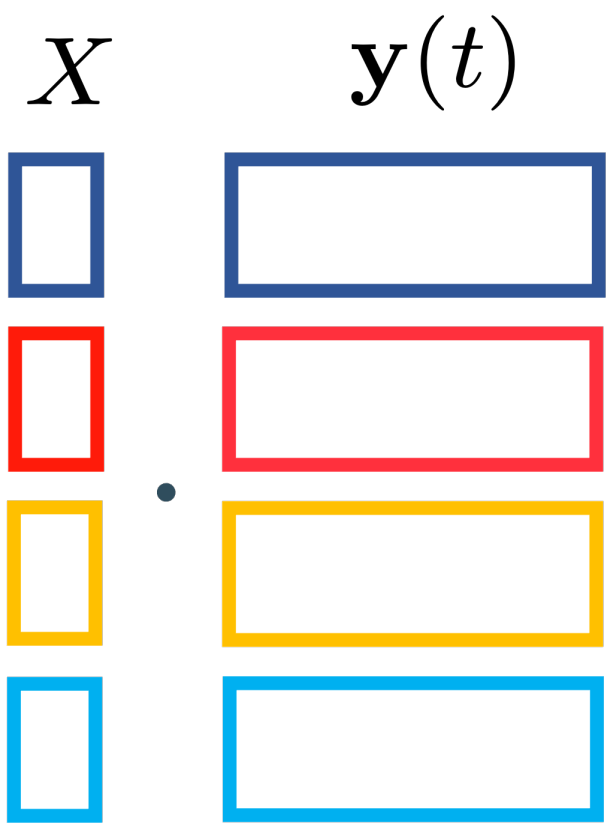
Loss and constraints only depend on X through $X^T \boldsymbol{y}(t)$ and possibly on $X^T B$

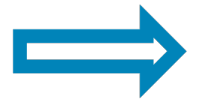


Local data – Local variable

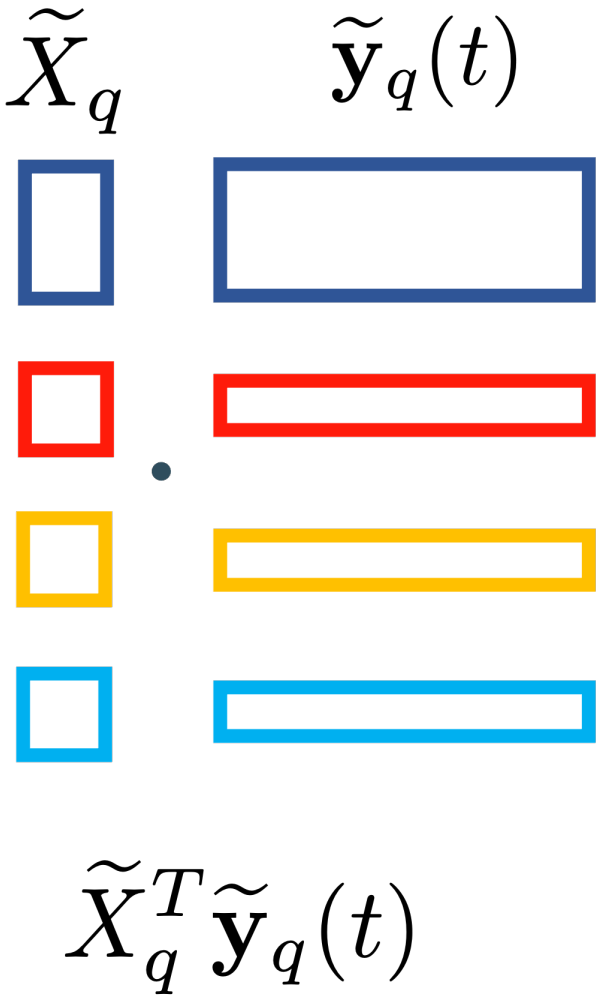


Local data – Local variable





At node q

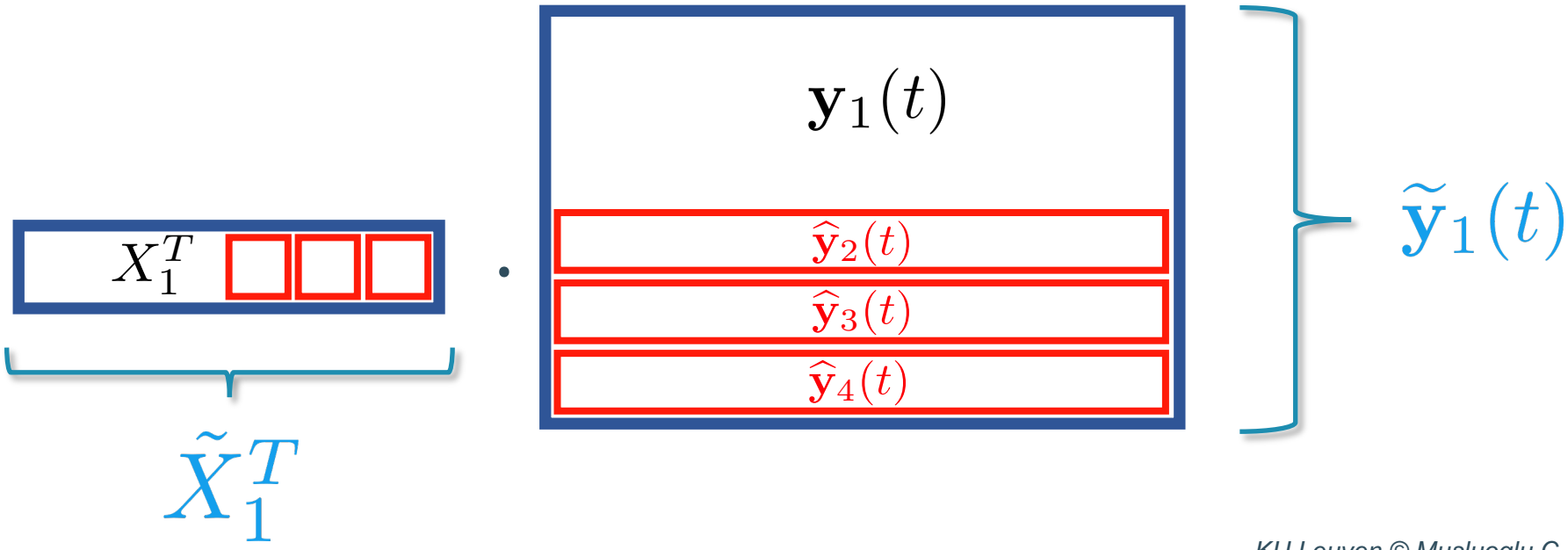


Remember this notation!

$\boldsymbol{\theta}$ = **original** (raw) data / variable

$\widehat{\boldsymbol{\theta}}_{\boldsymbol{k}}$ = **fused** data / variable

$\widetilde{\boldsymbol{\theta}}_{\boldsymbol{k}}$ = **stack** local $\boldsymbol{\theta}$ with $\widehat{\boldsymbol{\theta}}_{\boldsymbol{k}}$ from neighboring node(s)



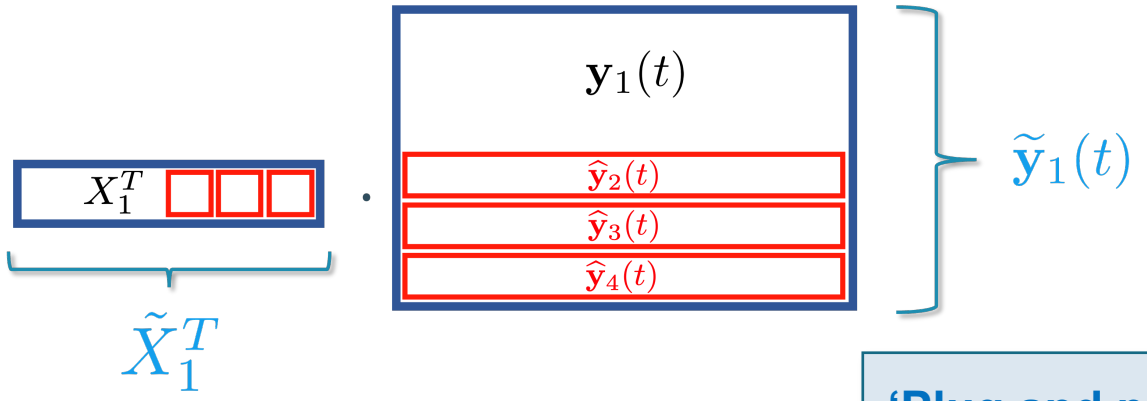
Local problems ~ Centralized problems

Original (**centralized**) problem:

$$\begin{aligned} \underset{X}{\text{minimize}}\ & f(X^T\mathbf{y}(t), X^T B) \\ \text{subject to}\ & g_j(X^T\mathbf{y}(t), X^T B) = 0, \\ & h_j(X^T\mathbf{y}(t), X^T B) \leq 0 \end{aligned}$$

Local problem at node *k*

$$\begin{aligned} \underset{\widetilde{X}_k}{\text{minimize}}\ & f(\widetilde{X}_k^T\tilde{\mathbf{y}}_k(t), \widetilde{X}_k^T\widetilde{B}_k) \\ \text{subject to}\ & g_j(\widetilde{X}_k^T\tilde{\mathbf{y}}_k(t), \widetilde{X}_k^T\widetilde{B}_k) = 0, \\ & h_j(\widetilde{X}_k^T\tilde{\mathbf{y}}_k(t), \widetilde{X}_k^T\widetilde{B}_k) \leq 0 \end{aligned}$$



'Plug and play': Can use the same solver as for original (centralized) problem

Local problems ~ Centralized problems

	Global	Local
MMSE:	$\underset{X}{\text{minimize}}\ \mathbb{E}[\|\|\mathbf{d} - X^T\mathbf{y}\|\|^2]$	$\underset{\widetilde{X}_k}{\text{minimize}}\ \mathbb{E}[\|\|\mathbf{d} - \widetilde{X}_k^T\widetilde{\mathbf{y}}_k\|\|^2]$
LCMV:	$\underset{X}{\text{minimize}}\ \mathbb{E}[\|\|X^T\mathbf{y}\|\|^2]$ subject to $X^TB = H$	$\underset{\widetilde{X}_k}{\text{minimize}}\ \mathbb{E}[\|\|\widetilde{X}_k^T\widetilde{\mathbf{y}}_k\|\|^2]$ subject to $\widetilde{X}_k^T\widetilde{B}_k = H$
CCA:	$\underset{X,W}{\text{maximize}}\ \mathbb{E}[\text{tr}(X^T\mathbf{y}\mathbf{v}^TW)]$ subject to $\mathbb{E}[X^T\mathbf{y}\mathbf{y}^TX] = I_Q$ $\mathbb{E}[W^T\mathbf{v}\mathbf{v}^TW] = I_Q$	$\underset{\widetilde{X}_k,\widetilde{W}_k}{\text{maximize}}\ \mathbb{E}[\text{tr}(\widetilde{X}_k^T\widetilde{\mathbf{y}}_k\widetilde{\mathbf{v}}_k^T\widetilde{W}_k)]$ subject to $\mathbb{E}[\widetilde{X}_k^T\widetilde{\mathbf{y}}_k\widetilde{\mathbf{y}}_k^T\widetilde{X}_k] = I_Q$ $\mathbb{E}[\widetilde{W}_k^T\widetilde{\mathbf{v}}_k\widetilde{\mathbf{v}}_k^T\widetilde{W}_k] = I_Q$

Local problems ~ Centralized problems

	Global	Local
PCA:	$\underset{\mathbf{x}}{\text{maximize}}\ \mathbb{E}[\lvert\mathbf{x}^T\mathbf{y}(t)\rvert^2]$ subject to $\mathbf{x}^T\mathbf{x} = 1$	$\underset{\tilde{\mathbf{x}}_k}{\text{maximize}}\ \mathbb{E}[\lvert\tilde{\mathbf{x}}_k^T\tilde{\mathbf{y}}_k(t)\rvert^2]$ subject to $\left(\tilde{\mathbf{x}}_k^T\widetilde{B}_k\right)\left(\widetilde{B}_k^T\tilde{\mathbf{x}}_k\right) = 1$

(with $B = I$) ⇩ Need to rewrite to fit DASF framework

$\underset{\mathbf{x}}{\text{maximize}}\ \mathbb{E}[\lvert\mathbf{x}^T\mathbf{y}(t)\rvert^2]$ subject to $\left(\mathbf{x}^T B\right)\left(B^T\mathbf{x}\right) = 1$

⇨ $R_{\mathbf{yy}}\mathbf{x}^* = \lambda(BB^T)\mathbf{x}^*$

generalized eigenvalue problem

⇨ $R_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}\tilde{\mathbf{x}}_k^* = \lambda(\widetilde{B}_k\widetilde{B}_k^T)\tilde{\mathbf{x}}_k^*$

generalized eigenvalue problem

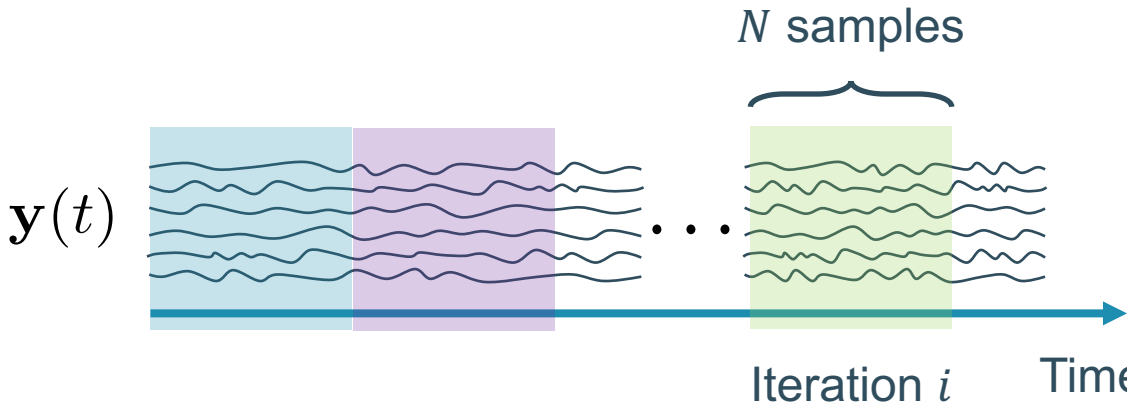
III - The fully-connected DASF algorithm (FC-DASF)

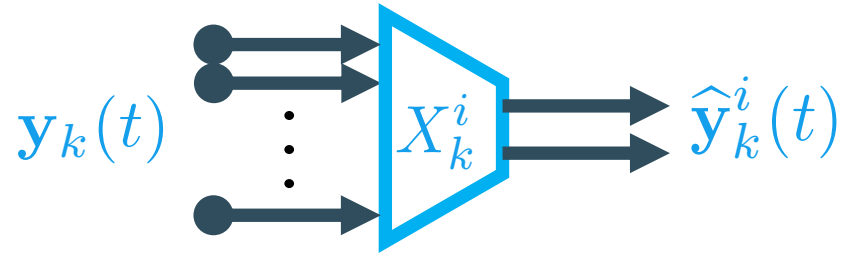
III.A - Algorithm derivation

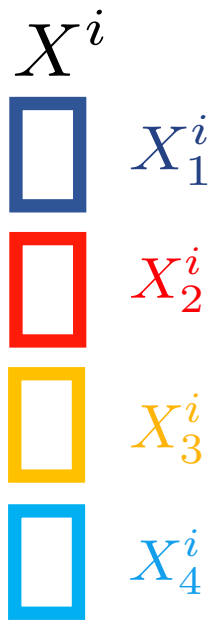
The FC-DASF algorithm

1) Sensor data collection & compression:

- Each node k collects N new samples of $\boldsymbol{y}_k$
- Each node k compresses its N samples : $\hat{\boldsymbol{y}}_k^i = X_k^{iT}\boldsymbol{y}_k$
- … and does the same for B_k (if applicable): $\hat{B}_k^i = X_k^{iT}B_k$







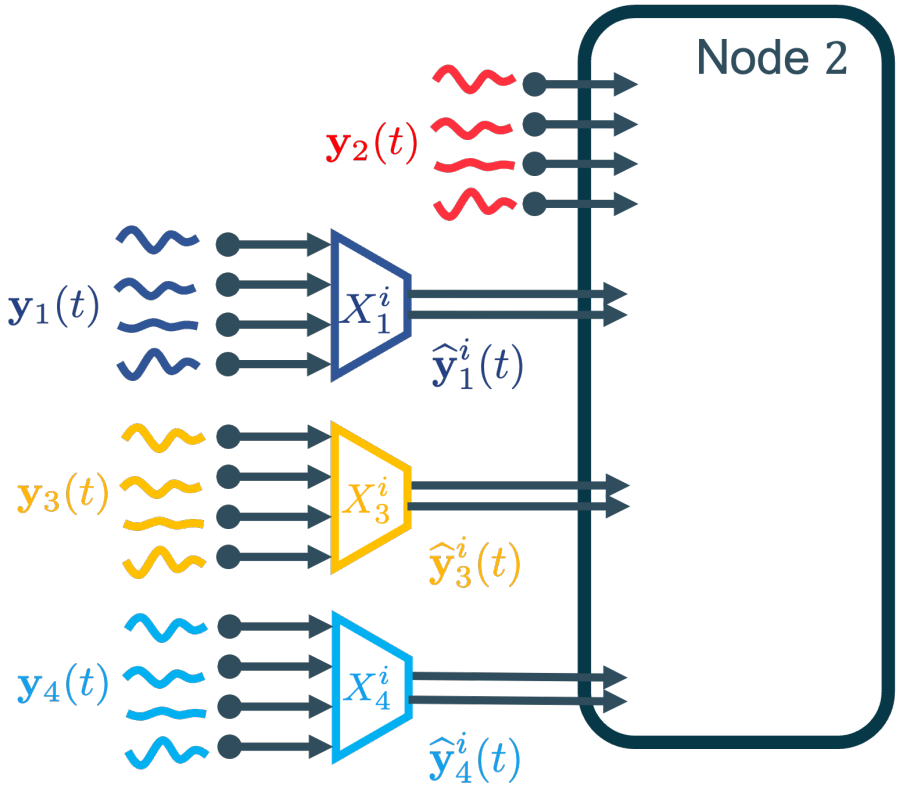
The FC-DASF algorithm

2) Transmit compressed data to updating node

- Select new updating node q (in this example $q = 2$)

- Nodes $k \neq q$ transmit N samples of $\hat{\mathbf{y}}_k^i$ to node q

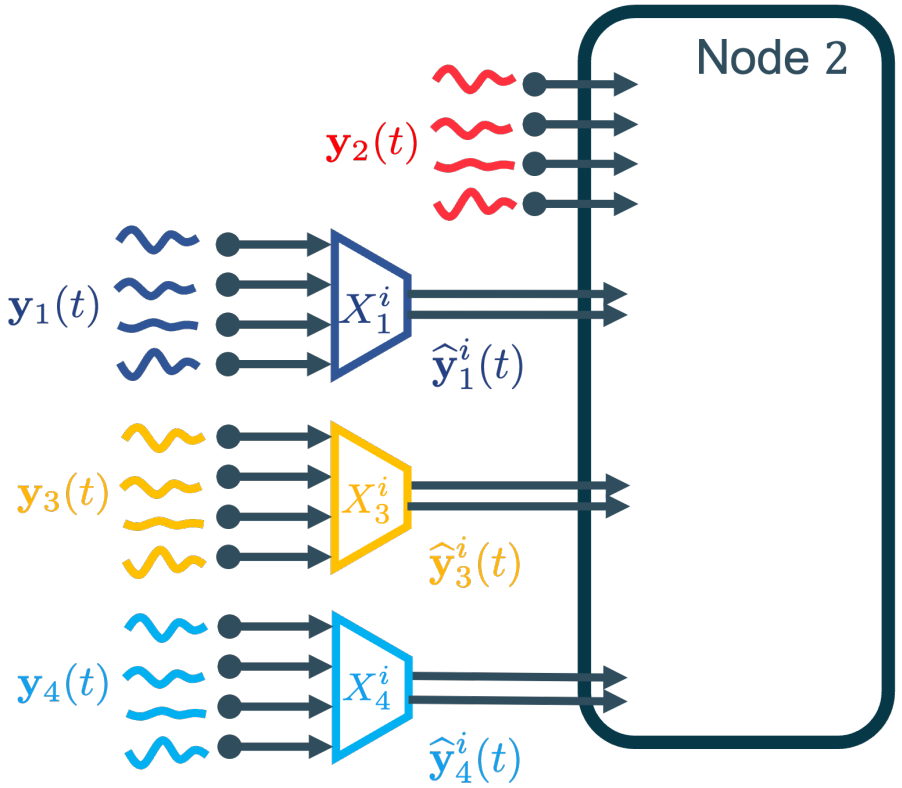
(+ $\hat{B}_k^i$ if applicable)



The FC-DASF algorithm

3) Solve local problem at updating node $q = 2$

- Node $q = 2$ has access to N samples $\{\tilde{y}_q^i(\tau)\}_{\tau=iN}^{iN+N-1}$ and to $\tilde{B}_q^i$



$$\{\tilde{\mathbf{y}}_2^i(\tau)\}_{\tau=iN}^{iN+N-1}$$

$$\widetilde{M}_2 \begin{cases} \{\mathbf{y}_2(\tau)\}_{\tau=iN}^{iN+N-1} \\ \{\hat{\mathbf{y}}_1^i(\tau)\}_{\tau=iN}^{iN+N-1} \\ \{\hat{\mathbf{y}}_3^i(\tau)\}_{\tau=iN}^{iN+N-1} \\ \{\hat{\mathbf{y}}_4^i(\tau)\}_{\tau=iN}^{iN+N-1} \end{cases}$$

$$\tilde{M}_2 = M_2 + Q(K-1)$$

The FC-DASF algorithm

4) Solve local problem at updating node $q = 2$

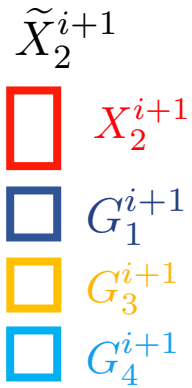
- Compute $\tilde{X}_q^{i+1}$ as the solution of

$$\widetilde{X}_q^{i+1} \triangleq \underset{\widetilde{X}_q}{\operatorname{argmin}} \; f(\widetilde{X}_q^T \widetilde{\mathbf{y}}_q^i(t), \widetilde{X}_q^T \widetilde{B}_q^i)$$

$$\text{subject to } g_j(\widetilde{X}_q^T \widetilde{\mathbf{y}}_q^i(t), \widetilde{X}_q^T \widetilde{B}_q^i) = 0,$$

$$h_j(\widetilde{X}_q^T \widetilde{\mathbf{y}}_q^i(t), \widetilde{X}_q^T \widetilde{B}_q^i) \leq 0$$

- Partition $\tilde{X}_2^{i+1}$



Iteration index i

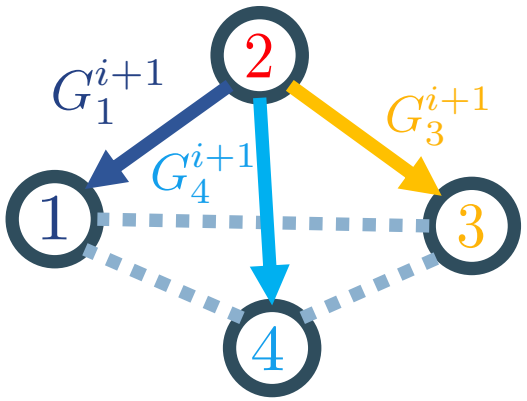
Updating node index q

'Plug and play': Can use the same solver as for original (centralized) problem

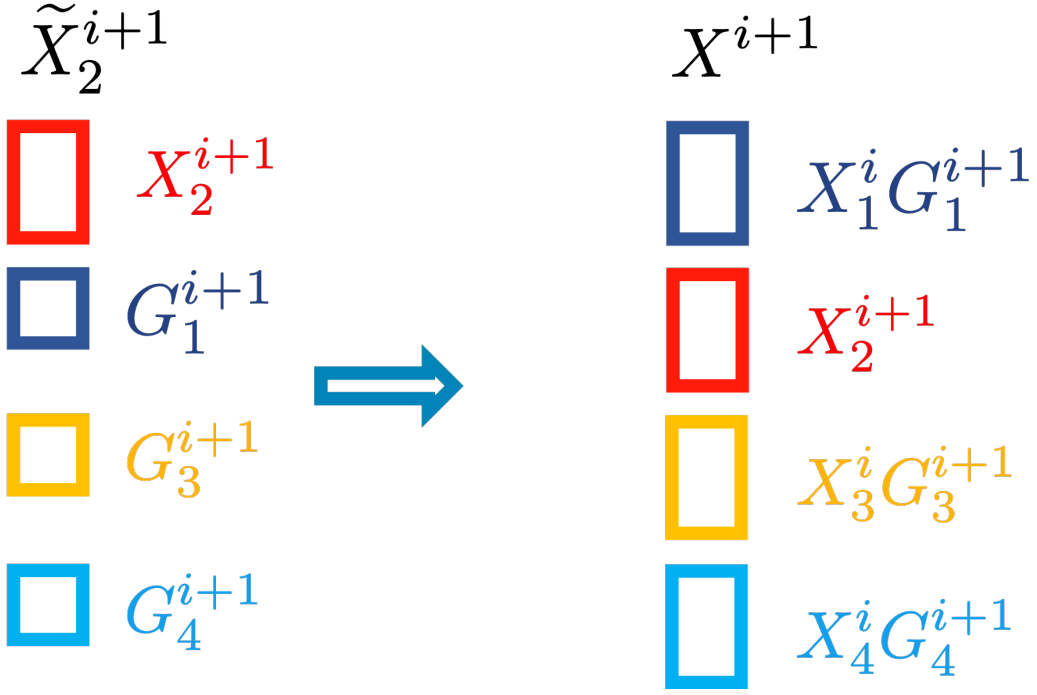
The FC-DASF algorithm

5) Update all nodes

- Node $q = 2$ transmits Q x Q matrices G_k^{i+1}



- X^{i+1} is the new estimate of X at iteration $i+1$



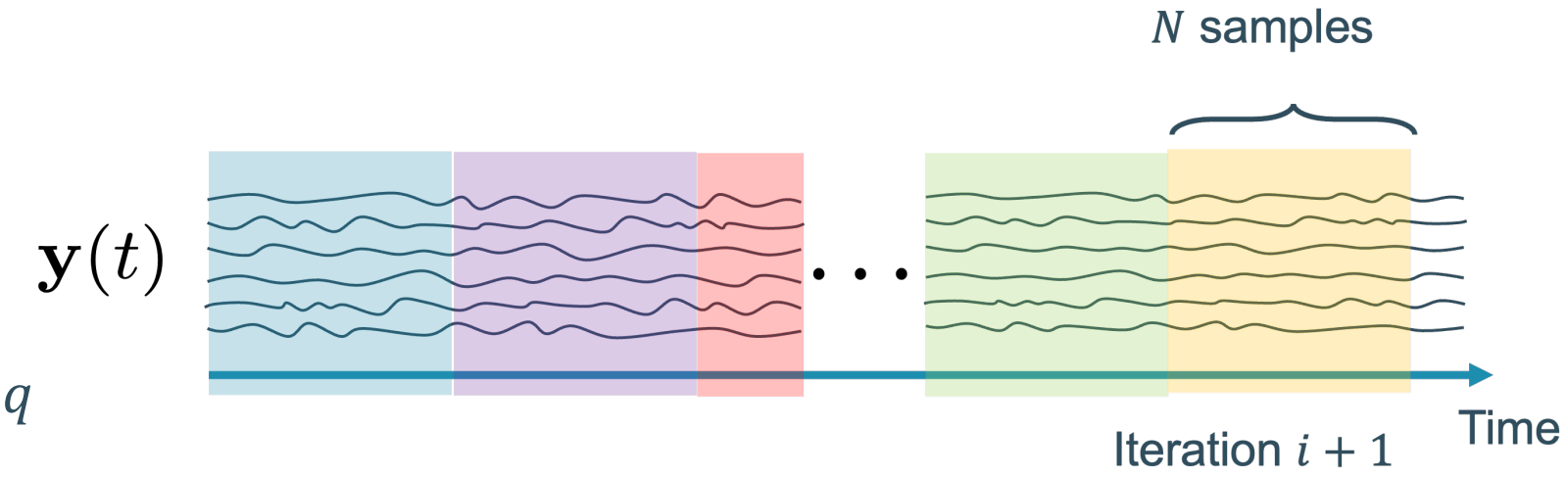
The FC-DASF algorithm

6) If desired: compute output of spatial filter for current block of N samples

- At updating node $q = 2$: $\left\{\boldsymbol{z}(\tau) = \left(\tilde{X}_q^{i+1}\right)^T \tilde{\boldsymbol{y}}_q^i(\tau)\right\}_{\tau=iN}^{iN+N-1}$
- If needed: transmit $\{\boldsymbol{z}(\tau)\}_{\tau=iN}^{iN+N-1}$ from node q to any node that is a data sink

The FC-DASF algorithm

- New samples
- Select new updating node q
- Repeat procedure
- Initialize X^0 randomly



Summary of FC-DASF

Initialization

Select updating node q

Collect N new samples, compress and transmit

Solve local problem at node q

Disseminate G's

Update X

Algorithm 1: Fully-Connected Distributed Adaptive Signal Fusion (FC-DASF) Algorithm

Initialize X^0, $i \leftarrow 0$.

repeat

Choose the updating node as $q \leftarrow (i \mod K) + 1$.

1) Every node k collects a new batch of N samples of $\mathbf{y}_k$, compresses these to N samples of $\widehat{\mathbf{y}}_k^i$ using $\widehat{\mathbf{y}}_k^i(t) = X_k^{iT}\mathbf{y}_k(t)$ and transmits them to node q.

$\widehat{B}_k^i$ is computed as $X_k^{iT}B_k$ and transmitted to node q.

at *Node* q **do**

2a) Compute $\widetilde{X}_q^{i+1}$ as the solution of the local problem. If the solution is not unique, select the solution which minimizes $||\widetilde{X}_q^{i+1} - \widetilde{X}_q^i||_F$ with $\widetilde{X}_q^i = [X_q^{iT}, I_Q, \ldots, I_Q]^T$.

2b) Partition X_q^{i+1} as $[X_q^T, G_1^T, \ldots, G_{q-1}^T, G_{q+1}^T, \ldots, G_K^T]^T$.

2c) Transmit G_k^{i+1} to node k for every $k \neq q$.

end

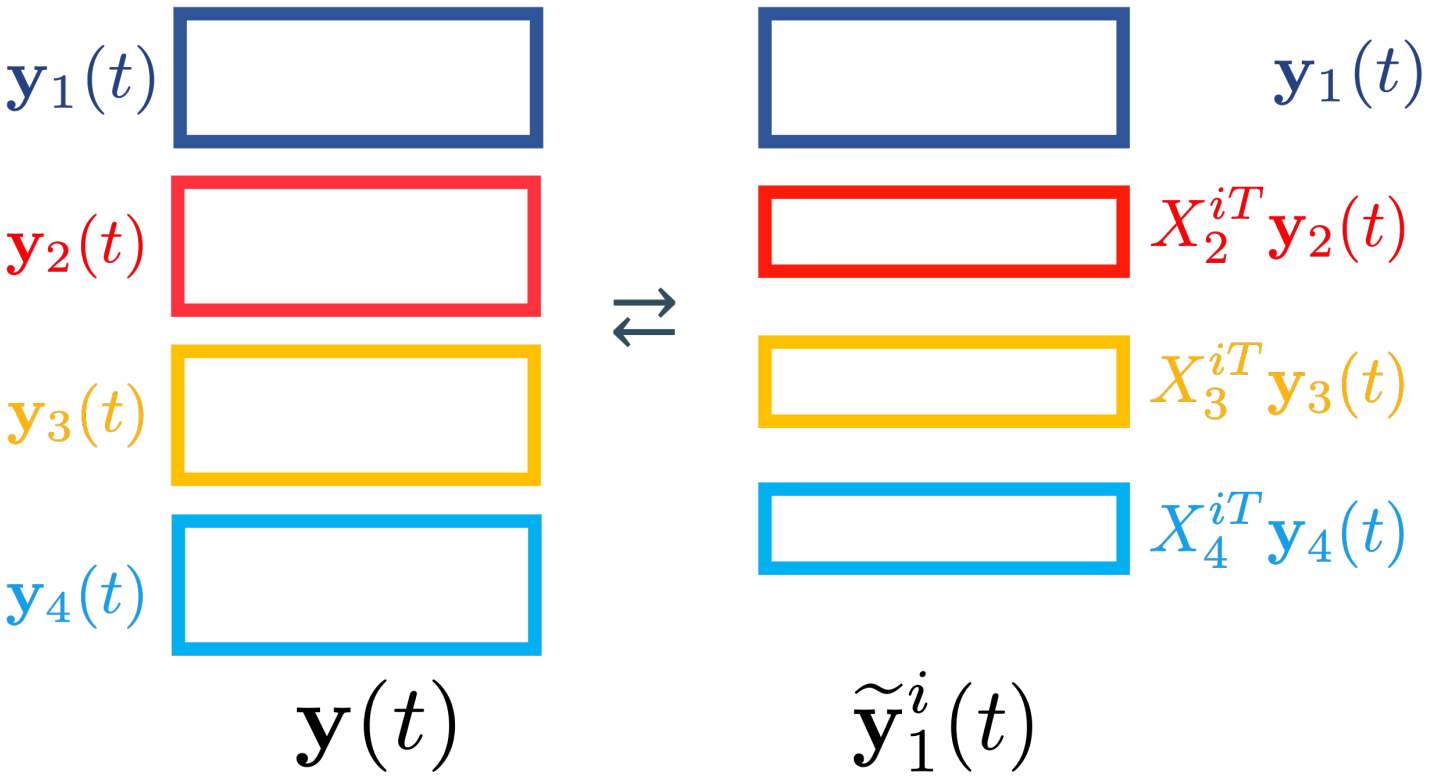
3) Every node updates $X_k^{i+1} = \begin{cases} X_q^{i+1}, & \text{if } q = k, \\ X_k^i G_k^{i+1}, & \text{if } q \neq k. \end{cases}$

$i \leftarrow i + 1$

III.B - Technical properties

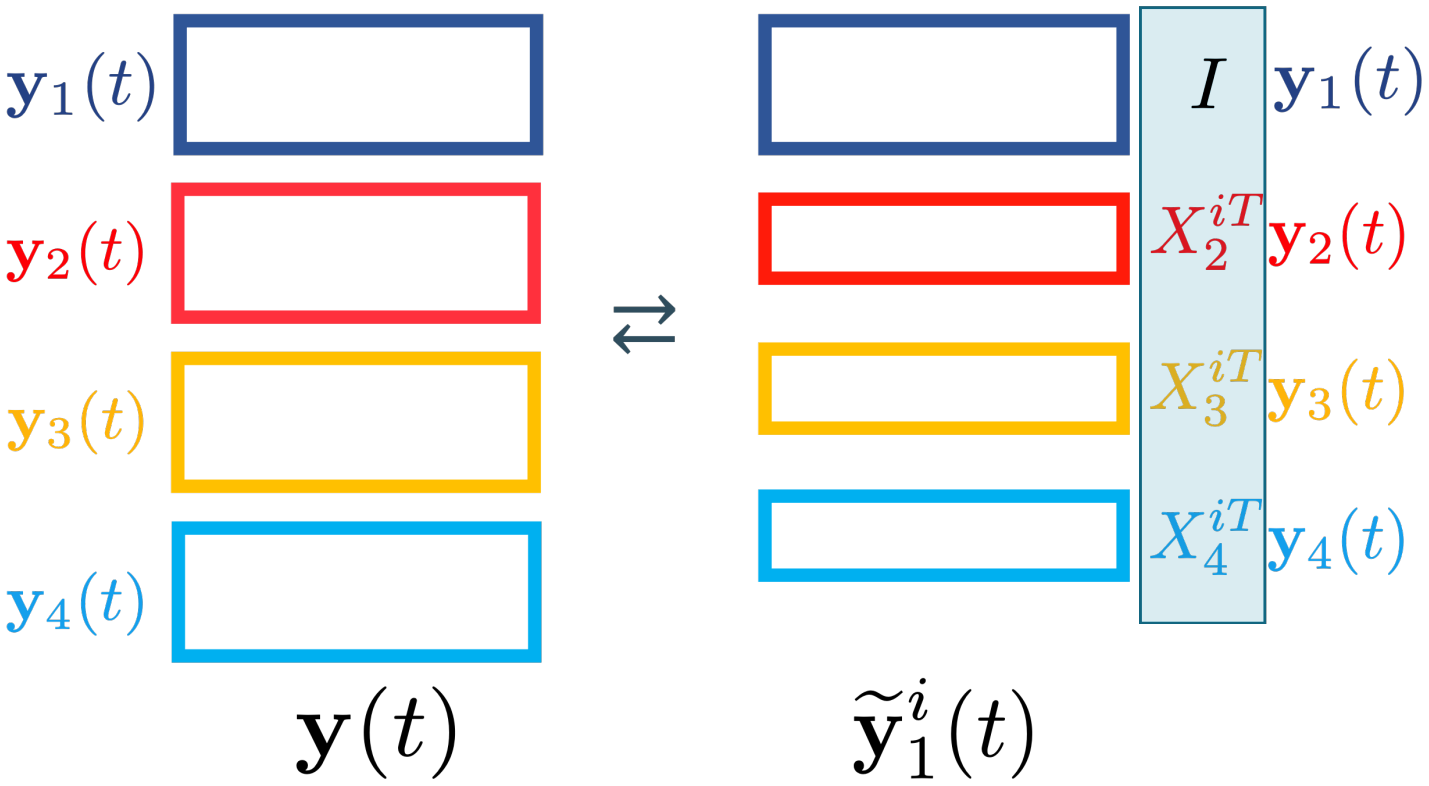
Linear relationship

$$\forall\, i, \forall\, q, \exists\, C_q^i \in R^{M \times \widetilde{M}_q} : \quad C_q^{iT}\mathbf{y}(t) = \widetilde{\mathbf{y}}_q^i(t)$$



Linear relationship

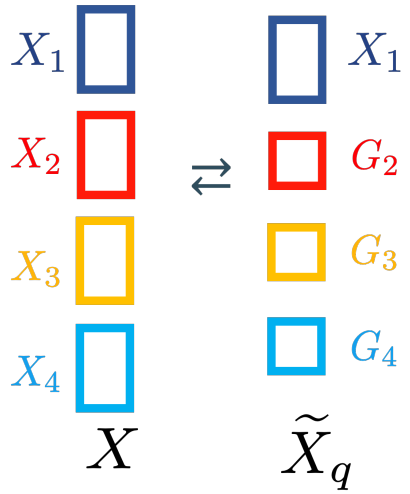
$$\forall\, i, \forall\, q, \exists\, C_q^i \in R^{M \times \widetilde{M}_q} : \quad C_q^{iT}\mathbf{y}(t) = \widetilde{\mathbf{y}}_q^i(t)$$



$$C_1^i = blkdiag(I, X_2^i, X_3^i, X_4^i)$$

Linear relationship

Parameterization of variable X via C_q^i : $X = C_q^i \widetilde{X}_q$



Local problem at node q :

$$\underset{\widetilde{X}_q}{\text{minimize}}\ f(\widetilde{X}_q^T \tilde{\mathbf{y}}_q(t), \widetilde{X}_q^T \widetilde{B}_q) \quad \text{subject to } \widetilde{X}_q \in \widetilde{\mathcal{S}}_q^i$$

$\rightleftarrows$

$$\underset{\widetilde{X}_q}{\text{minimize}}\ f(X^T \mathbf{y}(t), X^T B) \quad \text{subject to } X \in \mathcal{S},\ X = C_q^i \widetilde{X}_q$$

$\tilde{\mathcal{S}}_q^i$: Constraint set of the local problem

$\mathcal{S}$: Constraint set of the global problem

Feasible points

- It can be shown that:

$$\widetilde{X}_q \in \widetilde{\mathcal{S}}_q^i \iff X = C_q^i \widetilde{X}_q \in \mathcal{S}$$

- Since $\tilde{X}_q^{i+1} \in \tilde{\mathcal{S}}_q^i$ (by design):

$$X^i \in \mathcal{S}, \forall i > 0$$

i.e., all points generated by DASF are feasible points of the centralized problem

Monotonous decrease in objective

Introducing short notation: $F(X) = f(X^T \boldsymbol{y}(t), X^T B)$

$\left(F\left(X^i\right)\right)_i$ is a monotonously decreasing converging sequence

Assumptions

- Objective and constraint functions are **smooth**

  (we will briefly cover the non-smooth setting later)

- The problems are **well-posed**

  i.e., a small change in the parameters of the problems results in a small change in the optimal solution (~ no rank deficient covariance matrices etc.)

- Solutions of the centralized problem are **KKT points**,

  i.e. so-called linearly independent constraint qualifications (**LICQ**) are satisfied

- $\mathcal{S} \cap \{X : F(X) \leq F(X^0)\}$ is **compact**

Fixed points of DASF

- Under some technical conditions (akin to LICQ in optimization literature):

> Any fixed point $\bar{X}$ of DASF is a stationary point of the centralized problem

fixed point = invariant under DASF update, i.e., $X^{i+1} = X^i = \bar{X}$ for any updating node q

- These conditions can be shown to be satisfied ('with high probability') if the number of constraints J is upper bounded as

$$J \leq KQ^2$$

K: Number of nodes in the network
Q: Number of columns of X

Examples

Stiefel manifold constraints: $X^T X = I_Q$

$$X \; is \; M \times Q \qquad \rightarrow J = \frac{Q(Q+1)}{2}$$

$J \leq KQ^2$ always satisfied

Linear constraints: $X^T B = H$

$$B \; is \; M \times L \qquad \rightarrow J = QL$$

$J \leq KQ^2 \Leftrightarrow L \leq KQ$

Example: $Q = 2, K = 5 \Rightarrow L \leq 10$

Convergence of FC-DASF

Under previous assumptions and conditions:
$(X^i)_i$ converges to a stationary point $\bar{X}$ if the number of 'achievable' solutions is finite

- # 'achievable' solutions here depends on problem and solver:
  - **MMSE** is strongly convex, so only 1 achievable solution by definition
  - **PCA**: $\min_X \operatorname{tr}(X^T R_{\mathbf{yy}} X)$ s. t. $X^T X = I$
    - → Infinitely many solutions (if $\bar{X}$ is a solution, then also $\bar{X}U$ with U orth.)
    - → Eigenvalue solver extracts the $\bar{X}$ containing the eigenvectors (unique up to sign ambiguities), so # 'achievable' solutions by the solver is finite

- In case of infinite solution set → add a deterministic selection mechanism that leads to a finite set of achievable solutions (e.g. minimum norm, greedy selection, etc.)

Convergence of FC-DASF

Minima are the only **stable** fixed points of DASF

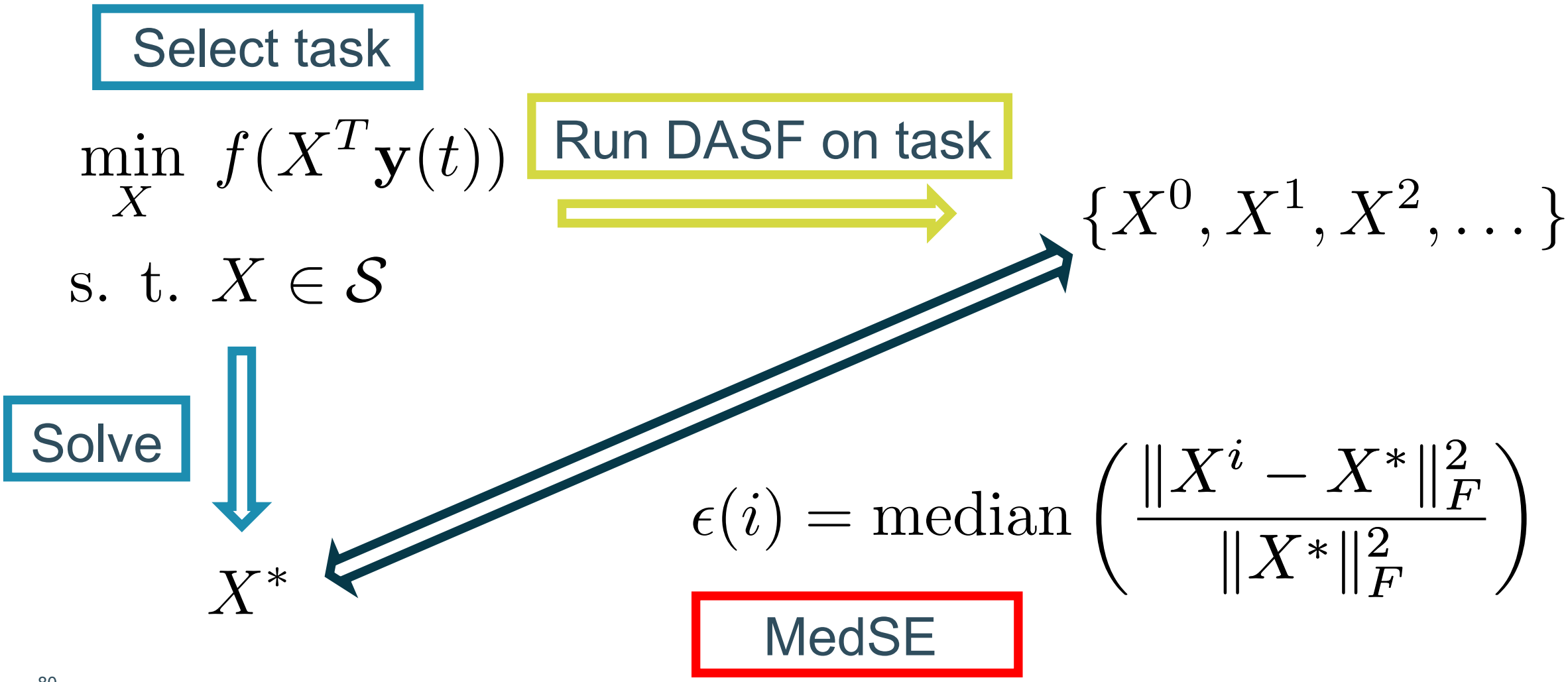
→ Follows from monotonic decrease of the objective

If all minima are global minima:

- **convergence to global solution** in practice (due to instability of all other stationary points)

- holds for many spatial filtering problems (e.g. PCA, CCA, generalized Rayleigh, convex problems, etc.)

Measuring performance of DASF

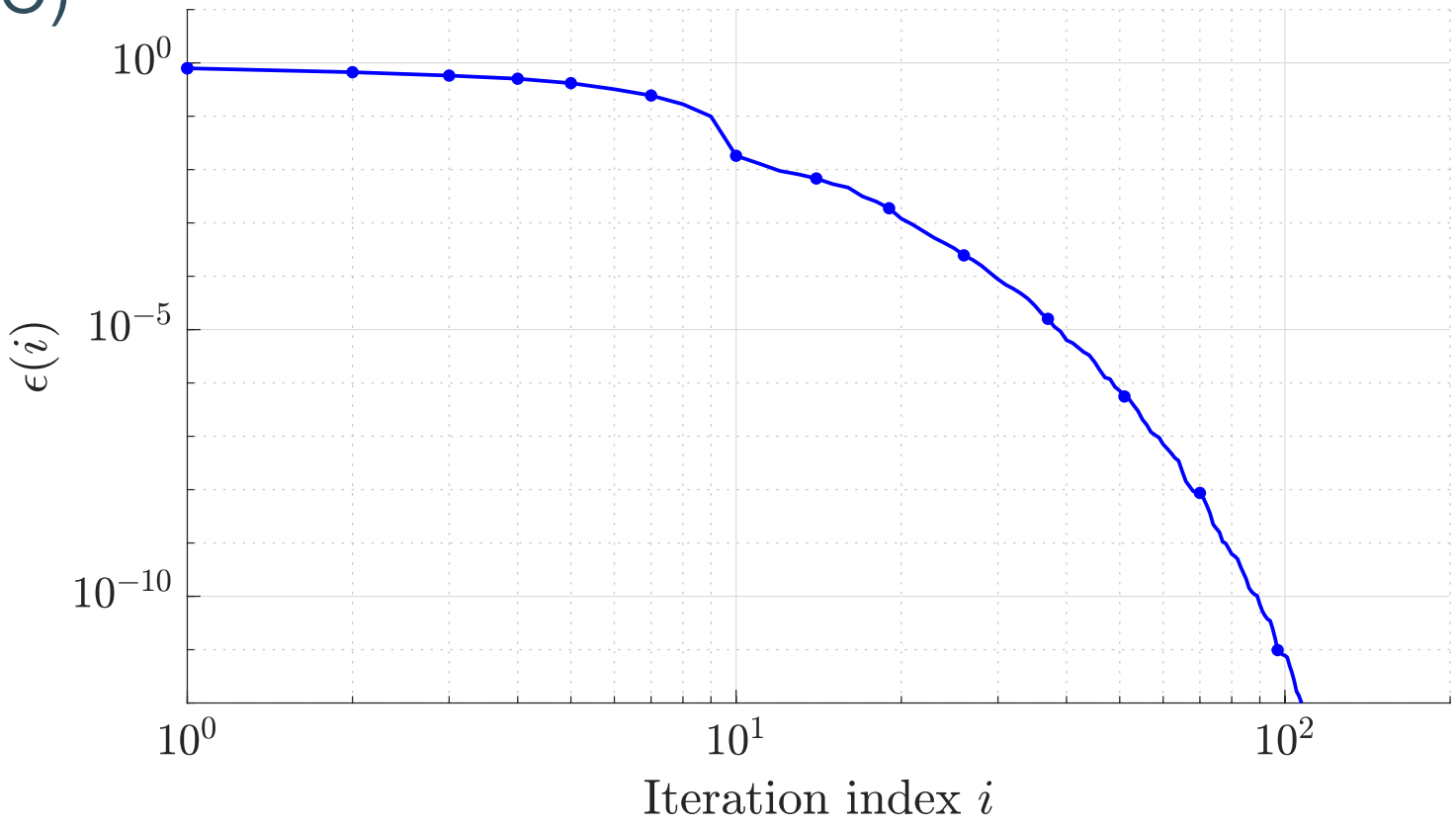


Simulations

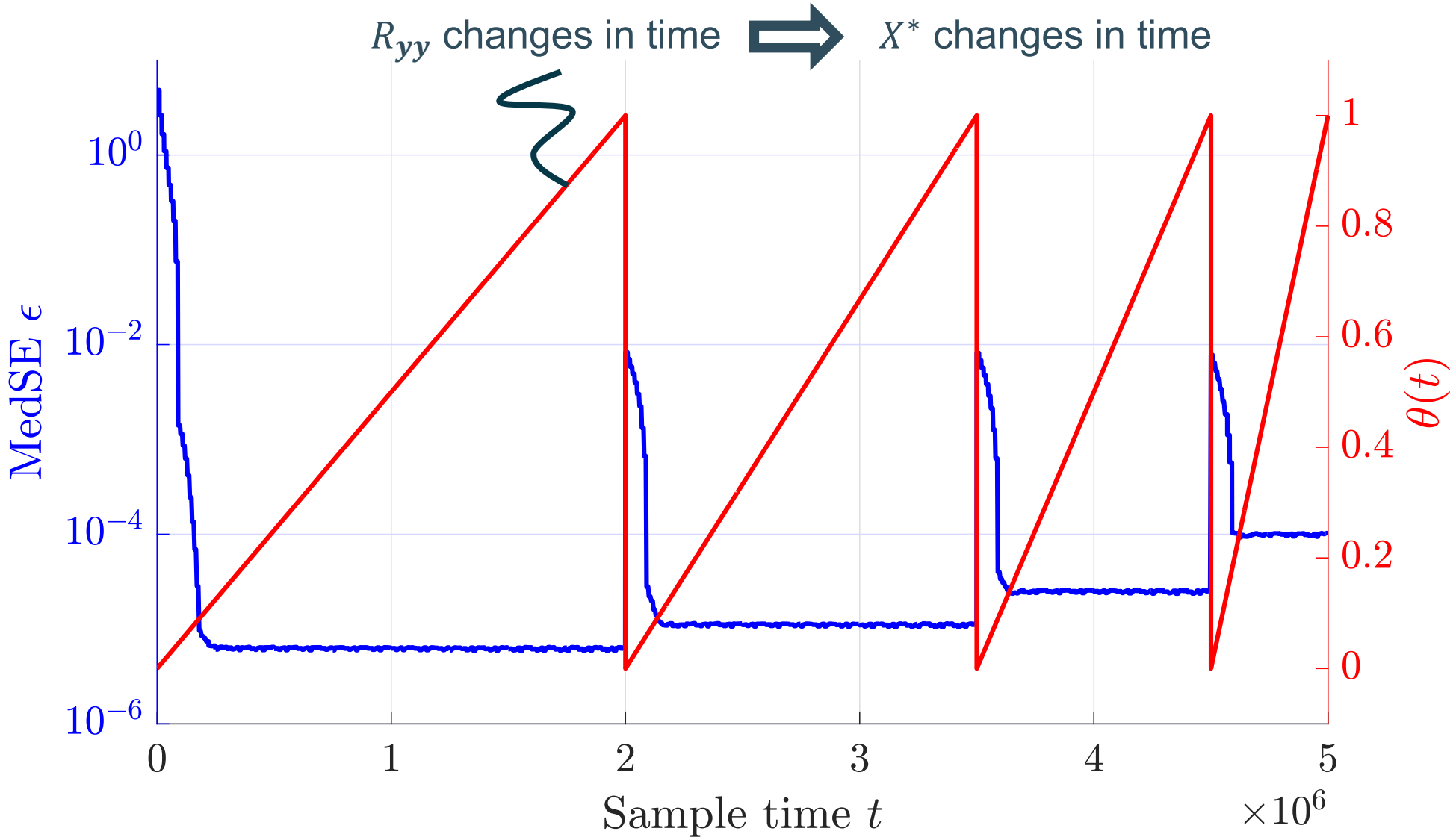
Trace ratio optimization (TRO)

(stationary setting)

$$\underset{X}{\text{maximize}} \ \frac{\mathbb{E}[||X^T\mathbf{y}(t)||^2]}{\mathbb{E}[||X^T\mathbf{v}(t)||^2]}$$
$$\text{subject to } X^TX = I_Q$$



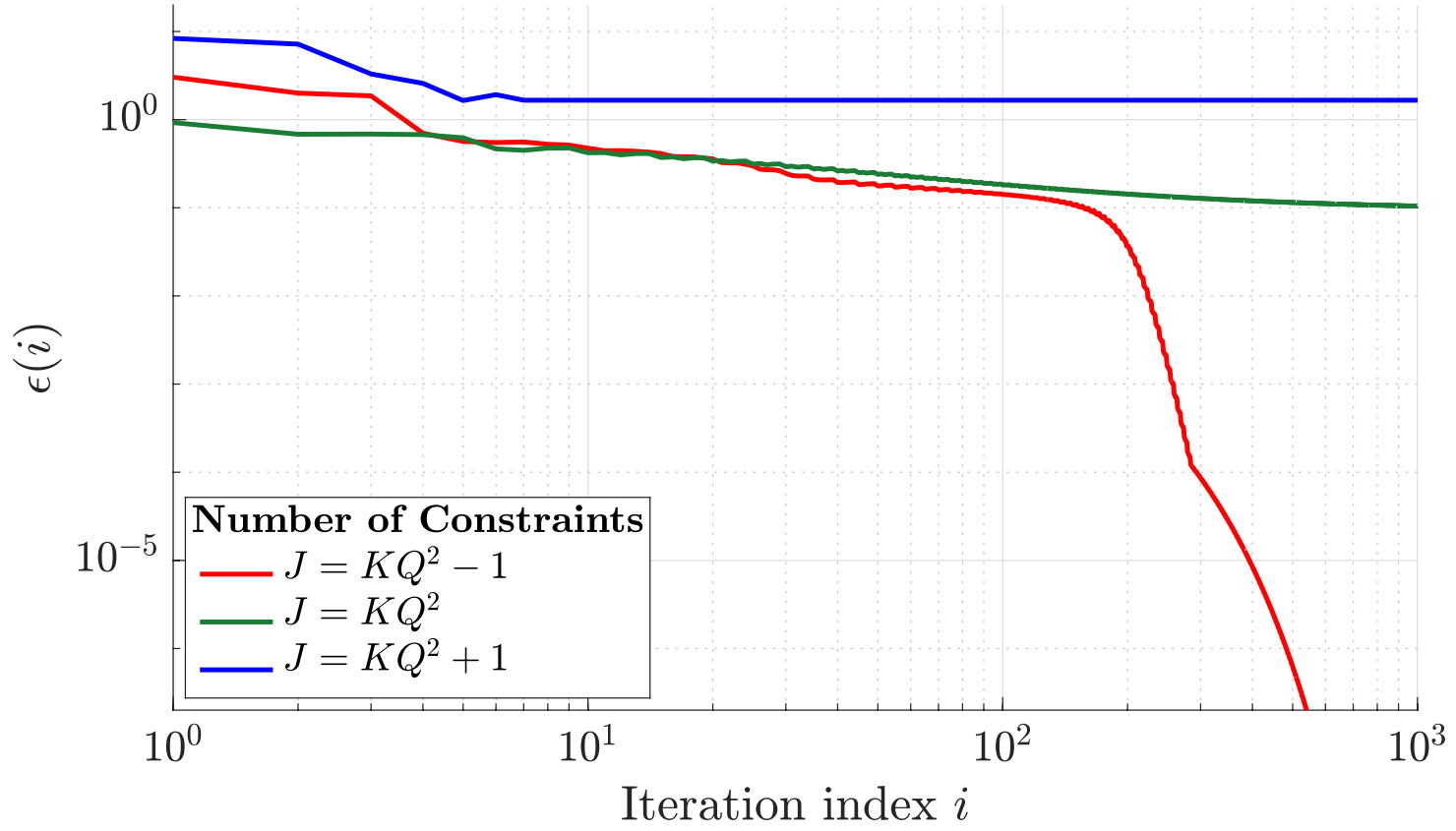
Adaptive setting



Convergence vs number of constraints

LCMV problem (stationary setting)

$$\underset{X}{\text{minimize}}\ \mathbb{E}[||X^T\mathbf{y}(t)||^2]$$
$$\text{subject to } X^T B = H$$

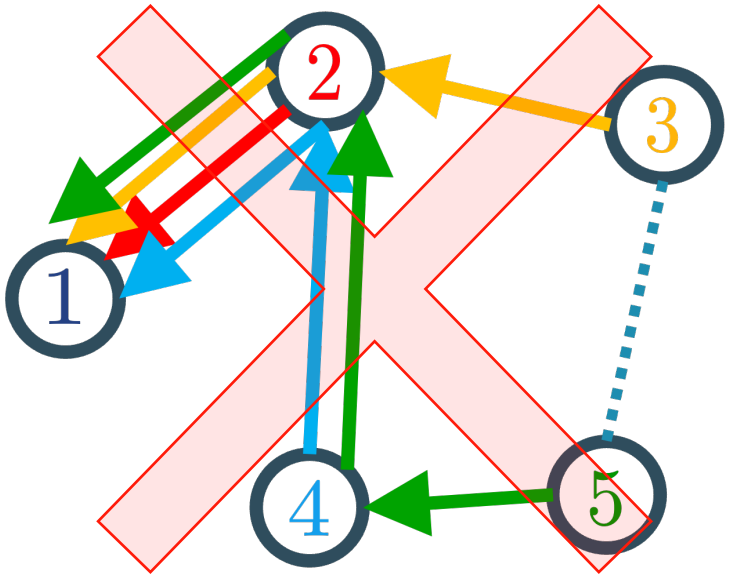


IV - Topology-independent DASF (TI-DASF)

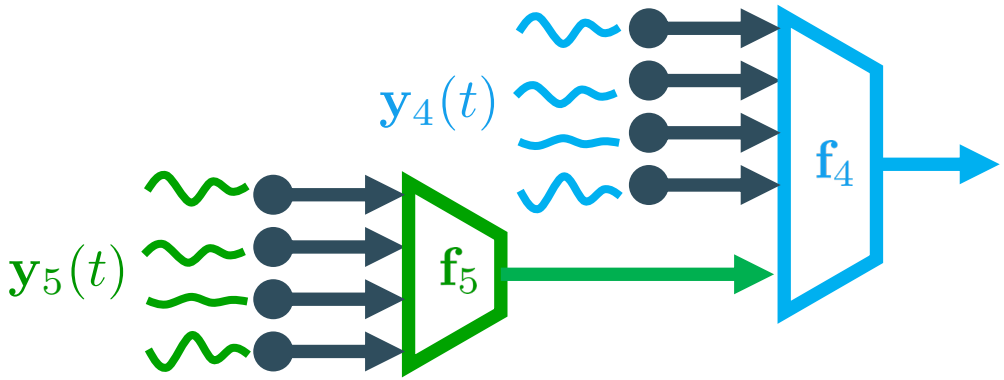
Extending DASF to arbitrary topologies
Naïve approach: Data relaying

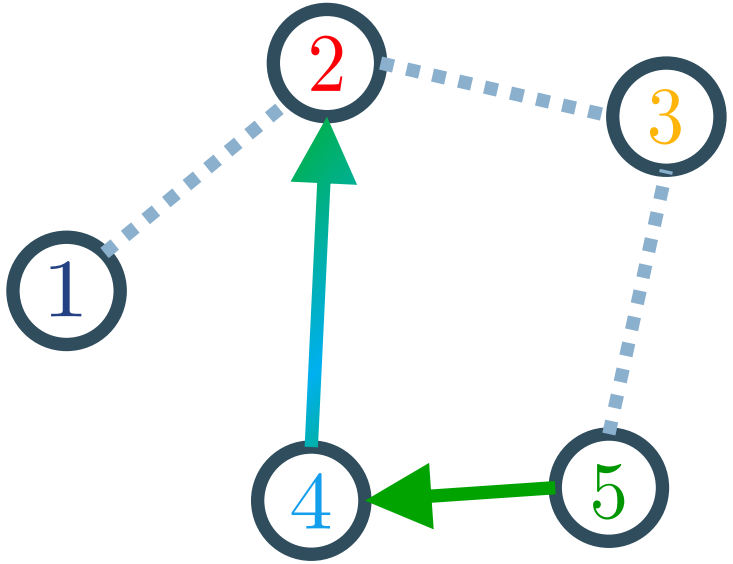
What if we emulate an FC topology by relaying the data?

- Requires **an additional networking layer**
- Worst-case bandwidth (Line topology): $\mathcal{O}(K)$
  - Scales **poorly** with the number of nodes
  - Should ideally be **independent** of the network's size



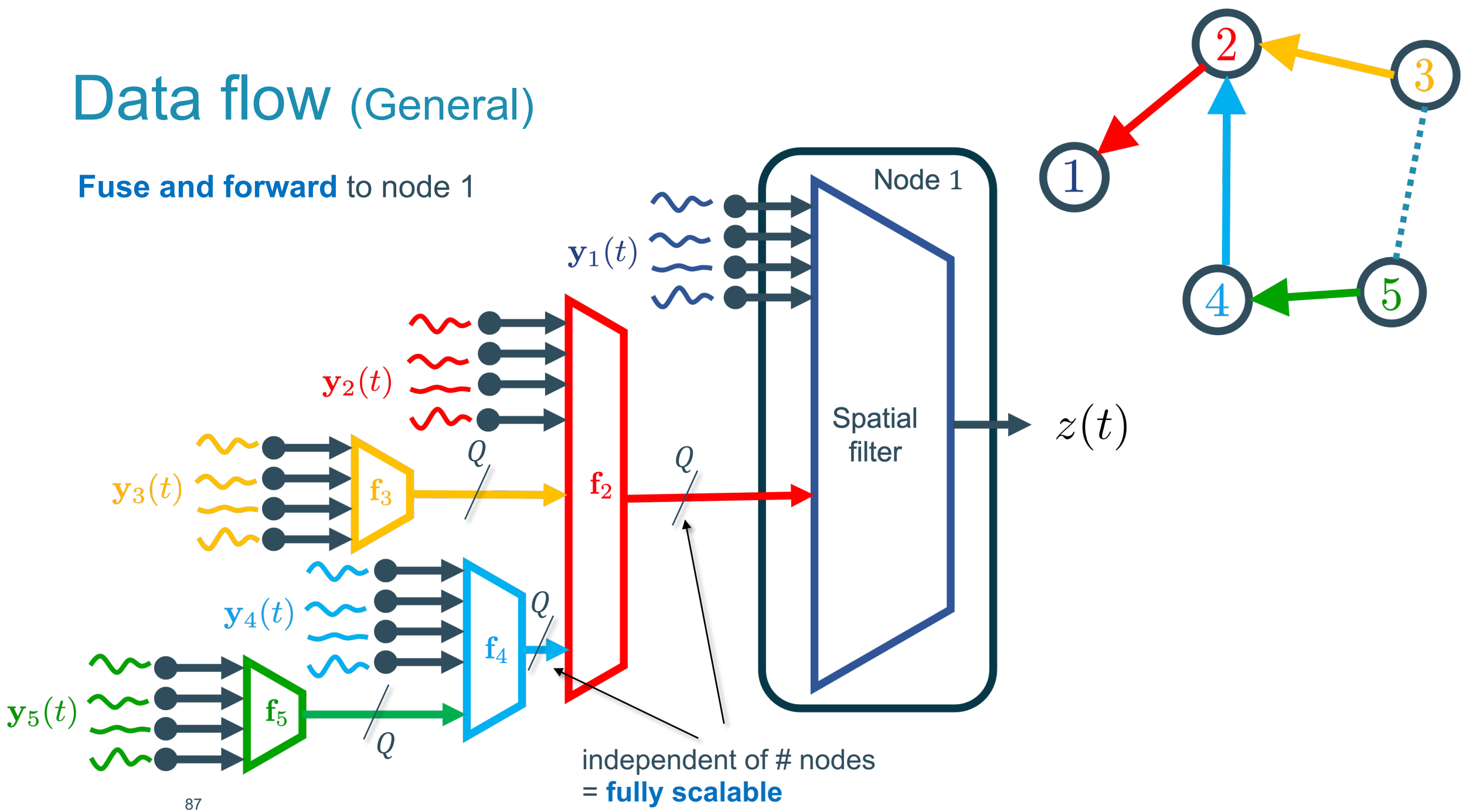
Data flow (General)





Data flow (General)

Fuse and forward to node 1

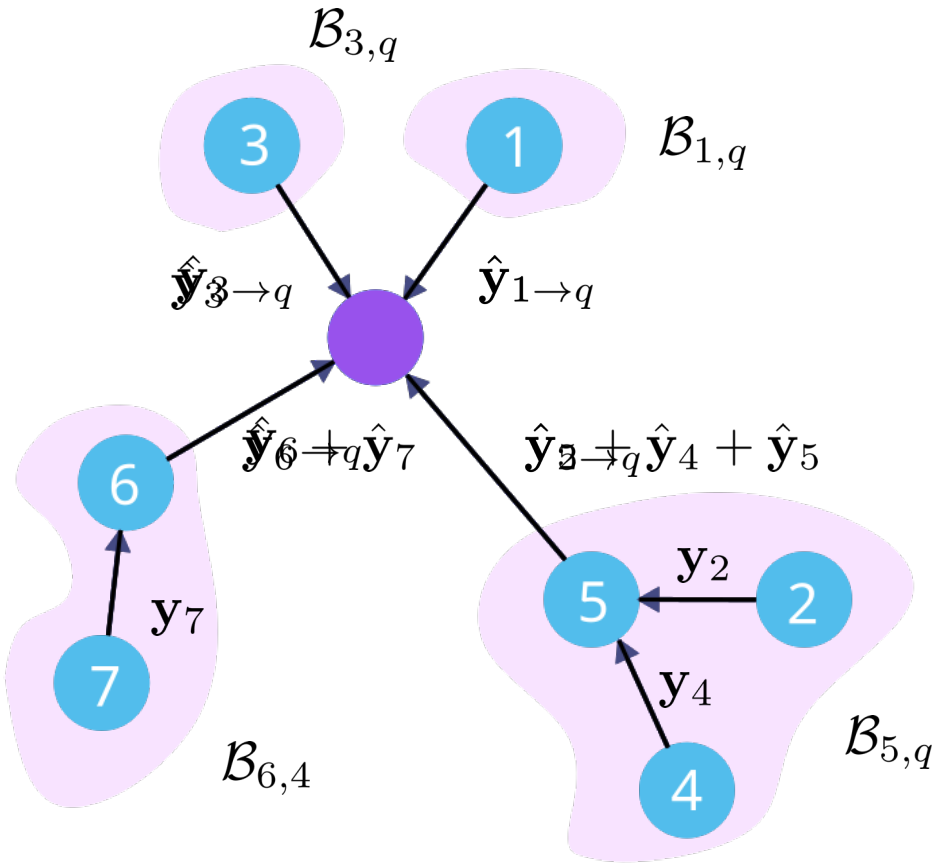


Tree topologies: fuse and forward

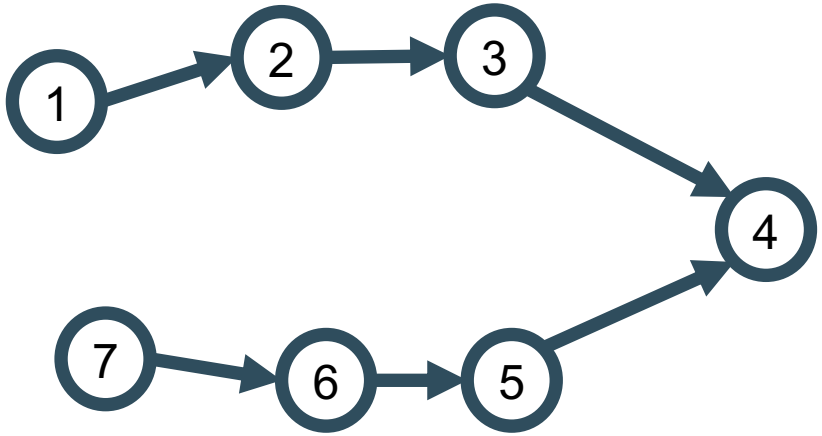
1. **Group** the nodes according to the branch $\mathcal{B}_{n,q}$ they belong to
2. **Recursively sum** (a new batch of) the data within each branch

$$\hat{\mathbf{y}}_{n\to q} = \sum_{k\in\mathcal{B}_{nq}} \hat{\mathbf{y}}_k$$

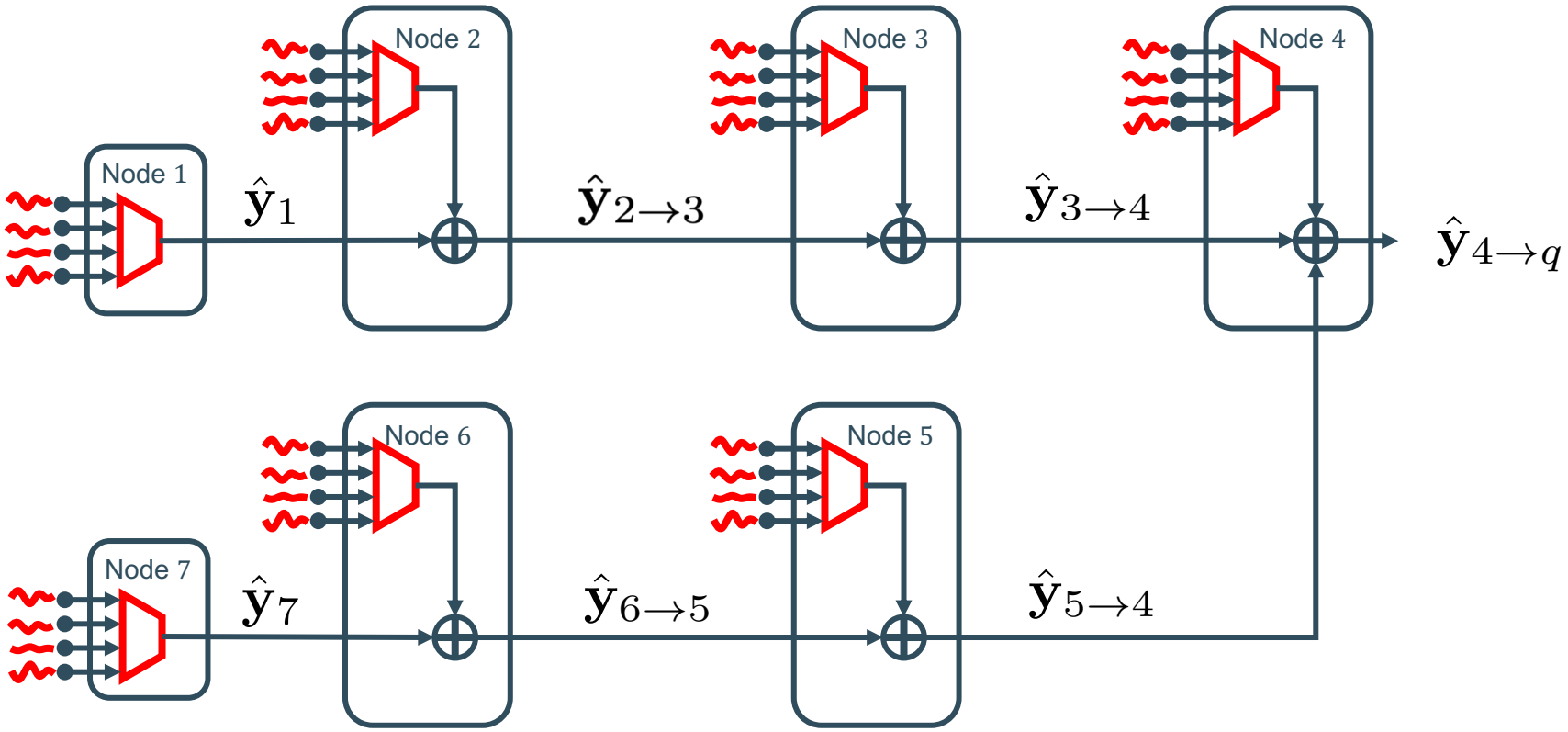
3. **Collect the sums** at the updating node



Tree topologies: fuse and forward



Fuse and forward towards updating node q



It is as if the branch consisted of a single node with filter

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \end{bmatrix}$$

and data

$$\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \\ \mathbf{y}_4 \\ \mathbf{y}_5 \\ \mathbf{y}_6 \\ \mathbf{y}_7 \end{bmatrix}$$

With compressed data

$$\hat{\mathbf{y}}_{4\to q}$$

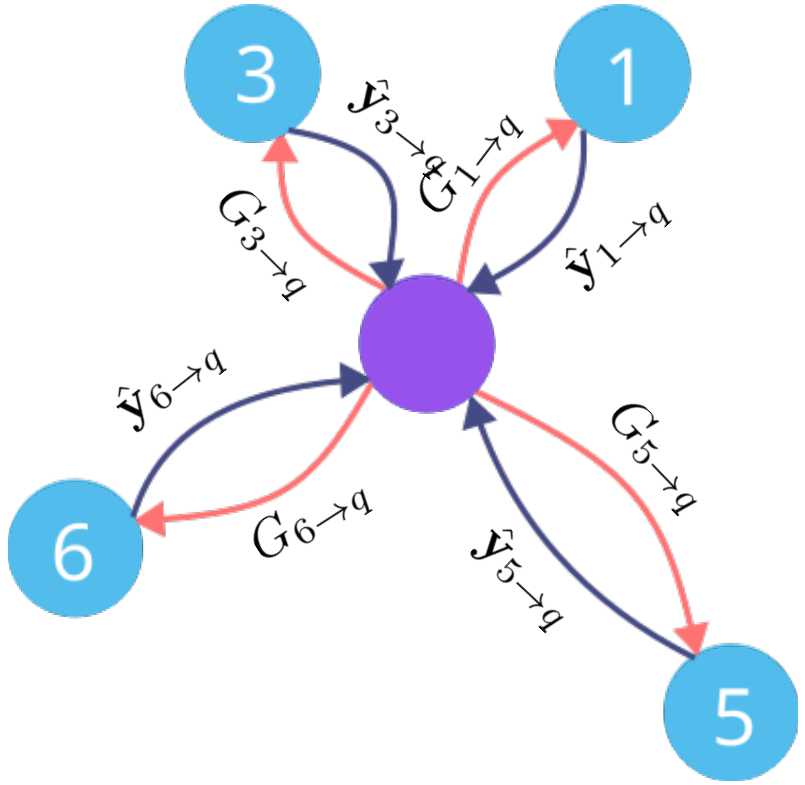
Tree topologies: Updating node

- The updating node has 4 neighbors, sending it 4 signals.
- As if this was a FC network, it constructs

$$\tilde{\mathbf{y}}_q = \begin{bmatrix} \mathbf{y}_q \\ \hat{\mathbf{y}}_{1\to q} \\ \hat{\mathbf{y}}_{3\to q} \\ \hat{\mathbf{y}}_{5\to q} \\ \hat{\mathbf{y}}_{6\to q} \end{bmatrix}$$

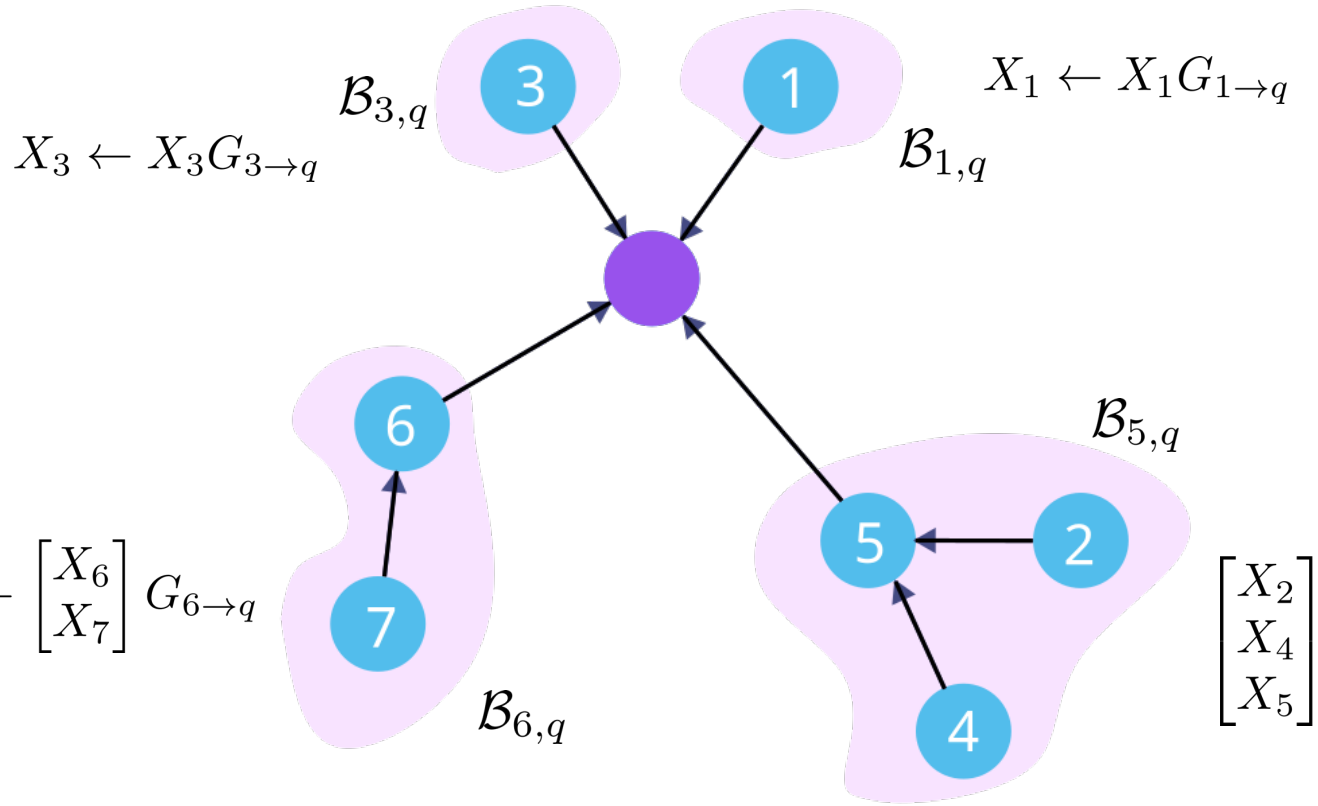
- And solve the usual local problem, obtaining

$$\widetilde{X}_q = \begin{bmatrix} X_q \\ G_{1\to q} \\ G_{3\to q} \\ G_{5\to 4} \\ G_{6\to 4} \end{bmatrix}$$



Tree topologies: Updating node

- The update is performed in each branch, as if it consisted of a single node i.e.



$$X_k^{i+1} \leftarrow X_k^i G_{n\to q} \quad \forall k \in \mathcal{B}_{n,q}$$

$$\begin{bmatrix} X_6 \\ X_7 \end{bmatrix} \leftarrow \begin{bmatrix} X_6 \\ X_7 \end{bmatrix} G_{6\to q}$$

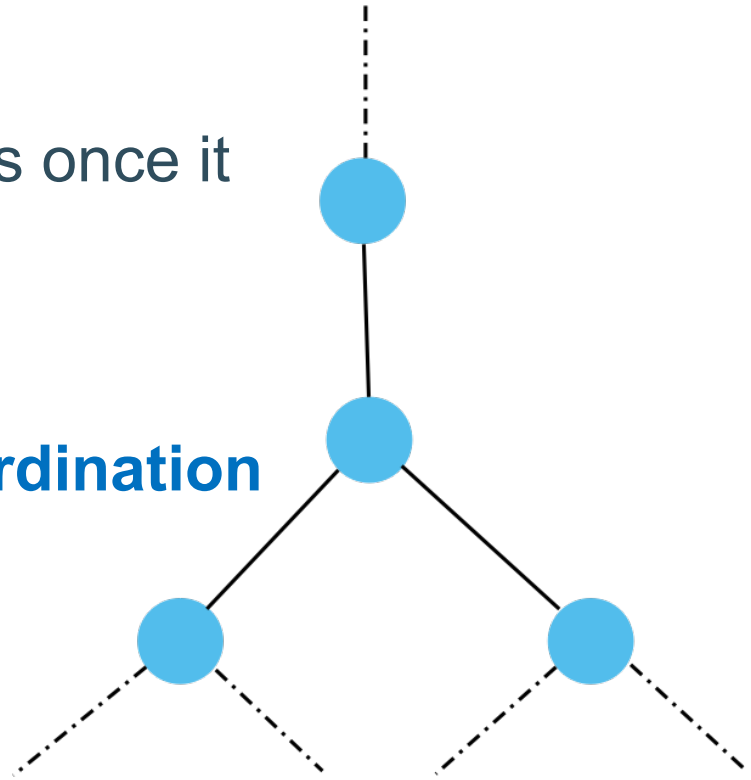
$$\begin{bmatrix} X_2 \\ X_4 \\ X_5 \end{bmatrix} \leftarrow \begin{bmatrix} X_2 \\ X_4 \\ X_5 \end{bmatrix} G_{5\to q}$$

Tree topologies: Non-updating nodes

- Any node k fuses and sends data to remaining neighbors once it has received data from $|\mathcal{N}_k| - 1$ neighbors

  → **data flow arises naturally without central coordination**

- Updates filter as soon as an update matrix is received
- Forwards the update to other neighbors



Tree topologies: fuse and forward

A node is unaware of the existence of other nodes in the network (except its neighbors)

- Each node transmits Q-channel signals (*once* per block of N samples)
- Communication load at each node only depends on # neighbors

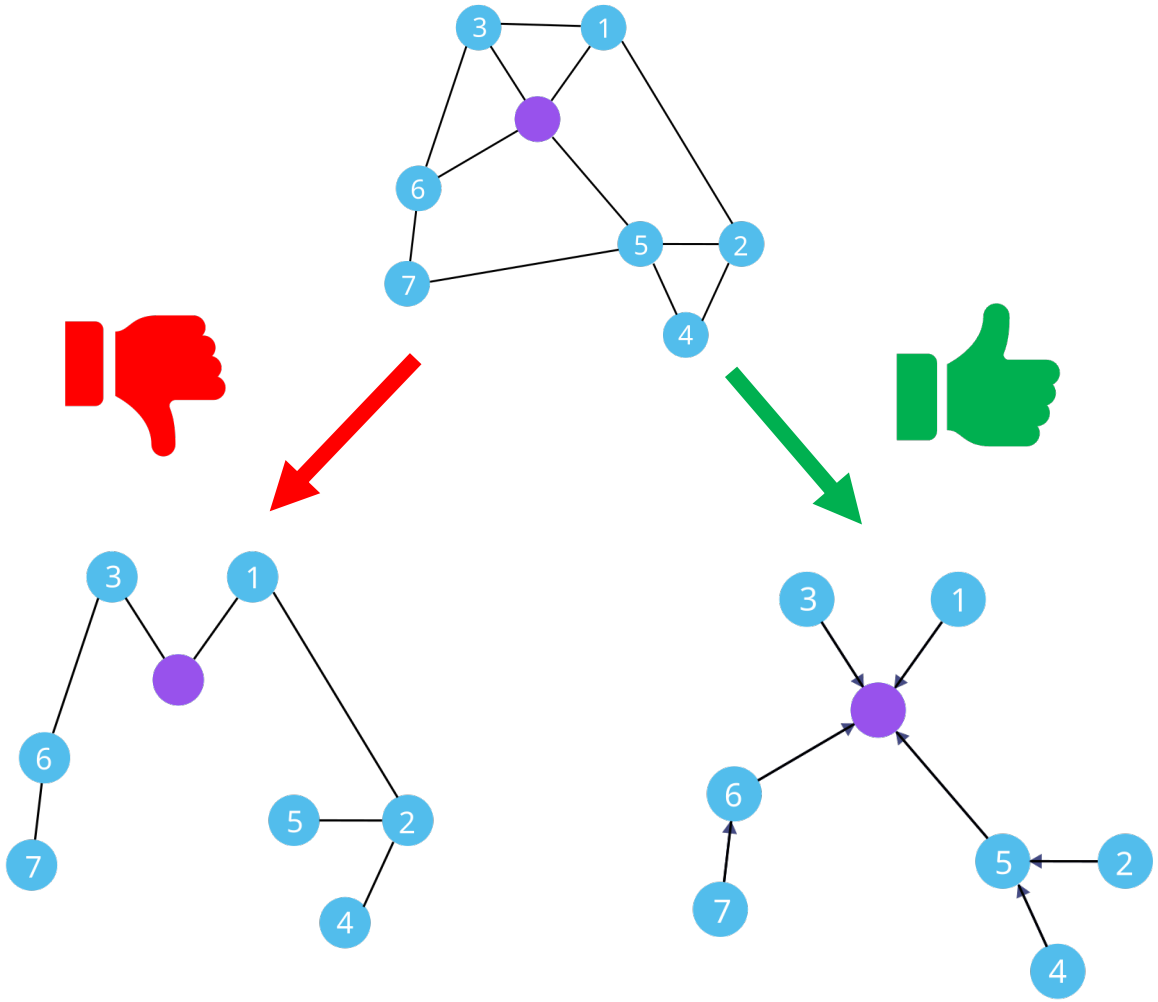
  → **Independent of network size**

  → **Fully scalable**

What about arbitrary topologies?

- Prune to spanning tree at each iteration
- **Keep as many neighbors of the updating node as possible**

  → maximize the degrees of freedom in $\tilde{X}_q$
- Pruning can be reused across cycles



Convergence

- Same convergence properties under LICQ-like conditions
- These LICQ-like conditions result in a new bound on the maximum # constraints

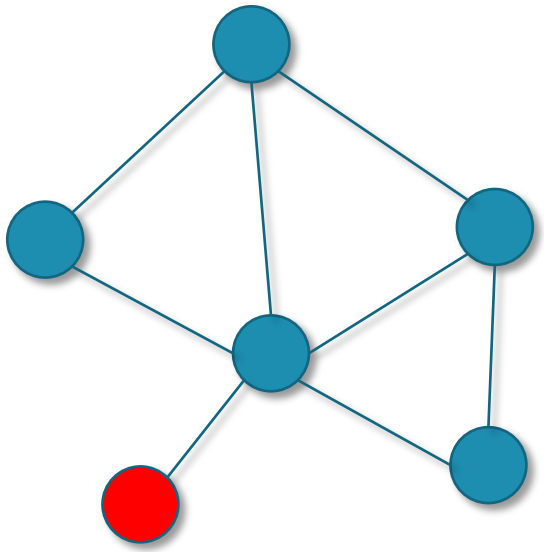
$$J \leq \min\left(\frac{Q^2}{K-1}\sum_{k\in\mathcal{K}}|\mathcal{N}_k|,\ (1+\min_{k\in\mathcal{K}}|\mathcal{N}_k|)Q^2\right)$$

K : Number of nodes in the network
Q : Number of columns of X
$\mathcal{N}_k$: Neighbors of node k (excl. node k itself)

Convergence

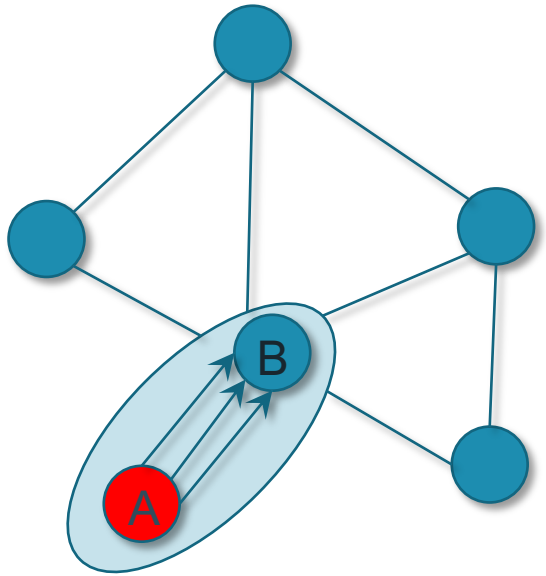
One single node with few neighbors can have a large impact

$$J \leq \min\left(\frac{Q^2}{K-1}\sum_{k\in\mathcal{K}}|\mathcal{N}_k|,\ (1+\min_{k\in\mathcal{K}}|\mathcal{N}_k|)Q^2\right)$$



If more constraints needed
→ merge nodes

i.e., A sends raw data to B, who treats this as own sensor data.



Examples

Stiefel manifold constraints: $X^T X = I_Q$

$$X \; is \; M \times Q \qquad \rightarrow J = \frac{Q(Q+1)}{2}$$

always satisfied

Linear constraints: $X^T B = H$

$$B \; is \; M \times L \qquad \rightarrow J = QL$$

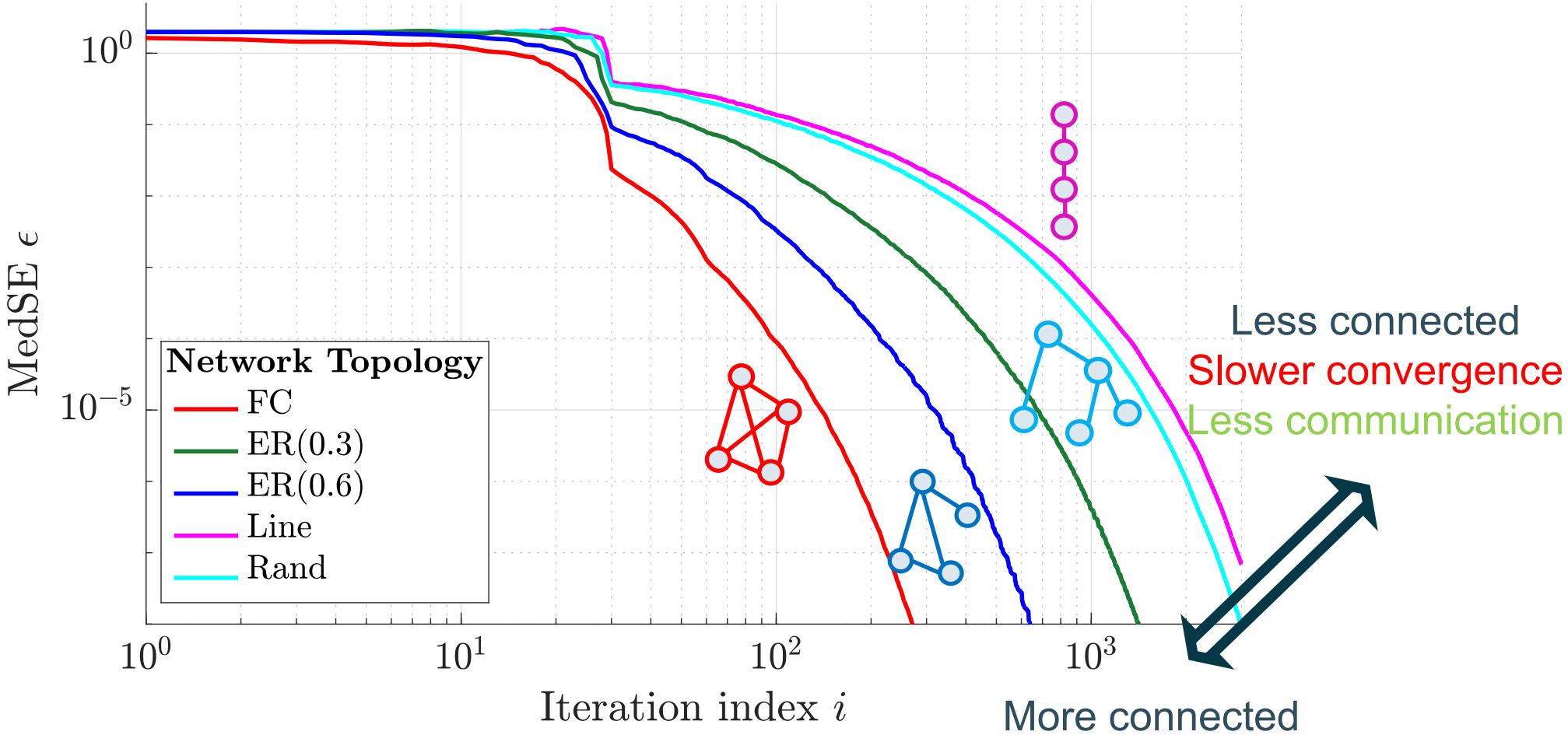
$$L \leq (1 + \min |\mathcal{N}_q|)Q$$

$$L \leq \frac{Q}{K-1} \sum |\mathcal{N}_q|$$

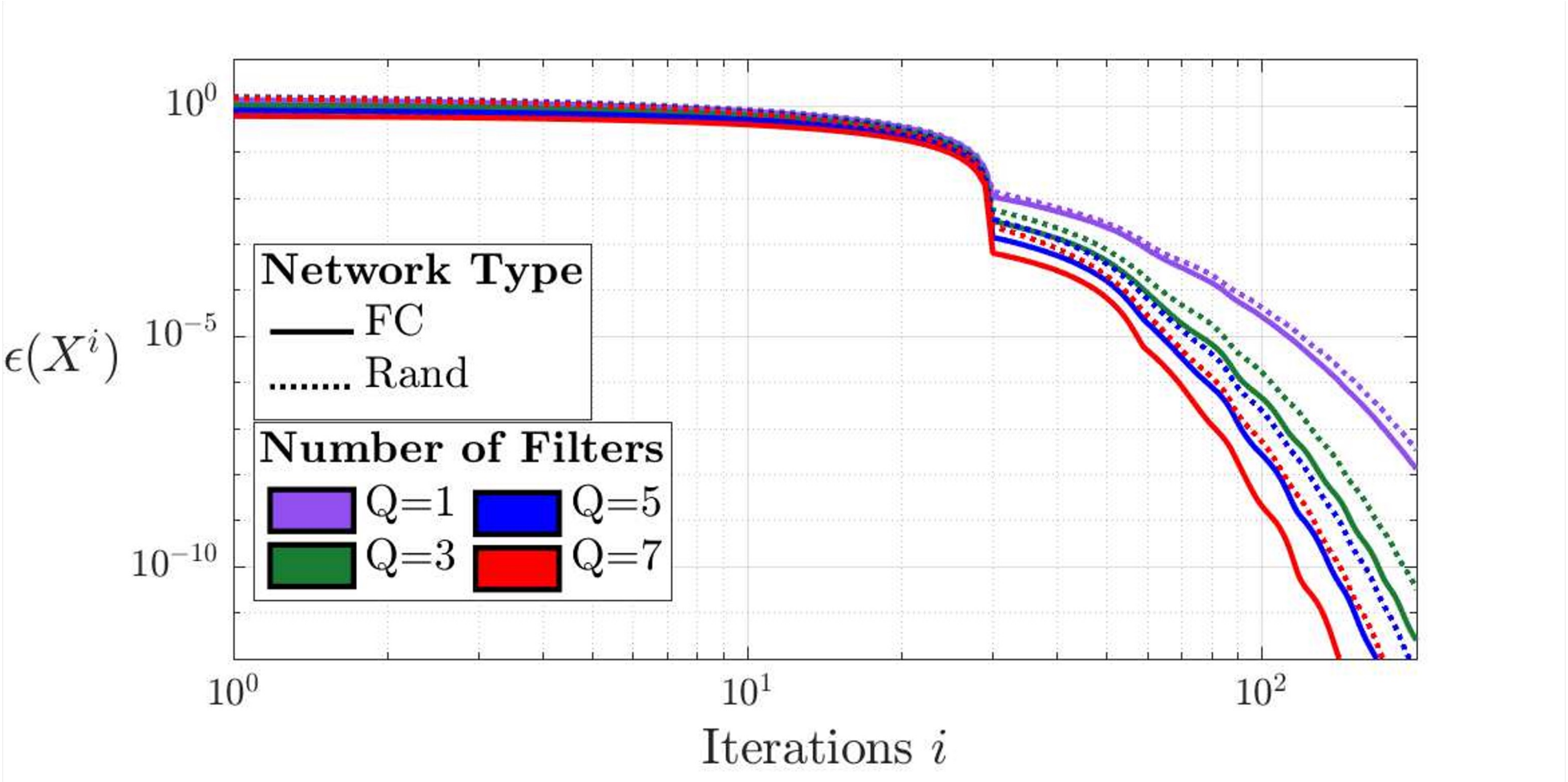
Stationary setting

Trace ratio optimization (TRO)

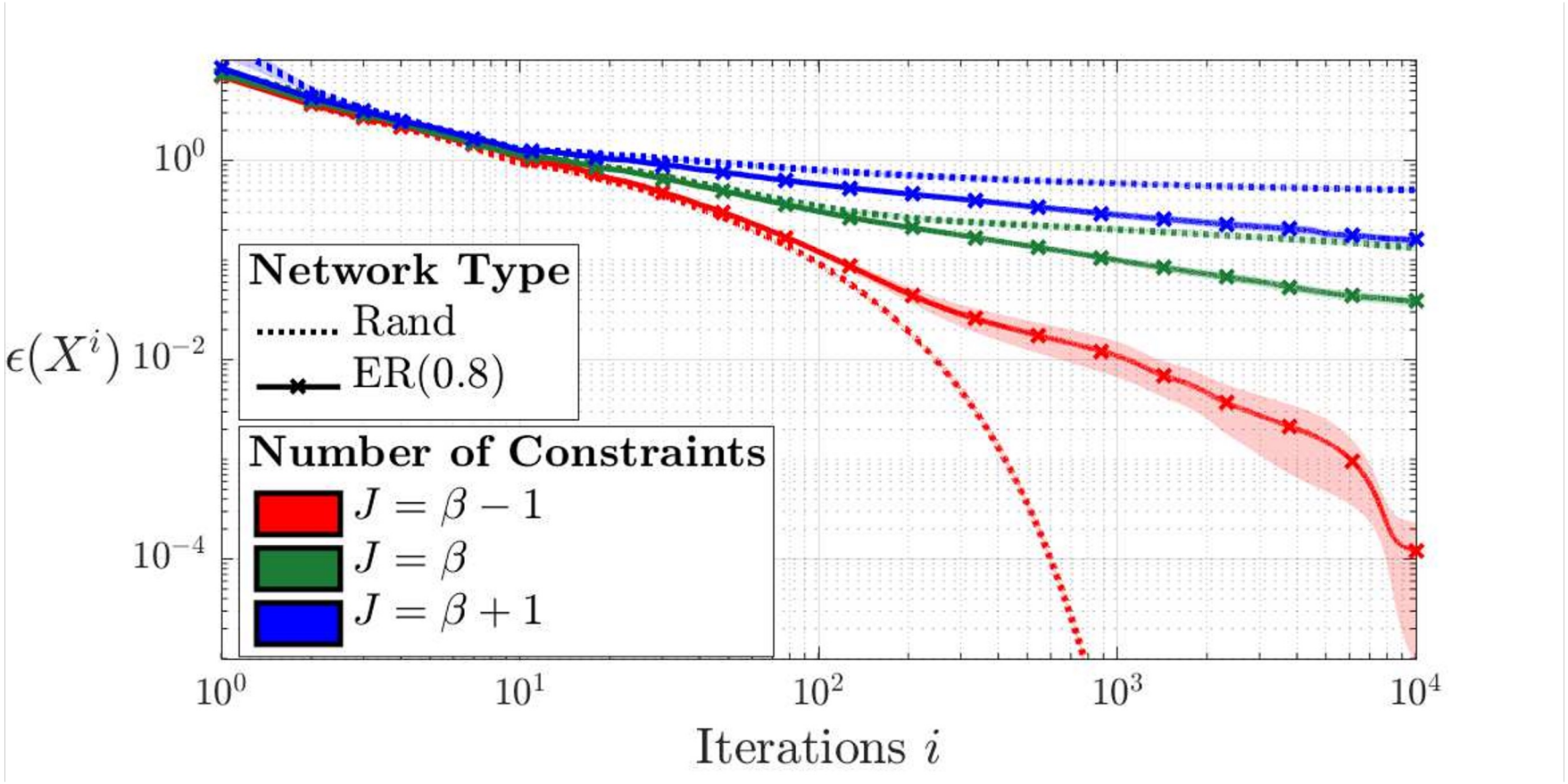
$$\underset{X}{\text{maximize}} \; \frac{\mathbb{E}[||X^T \mathbf{y}(t)||^2]}{\mathbb{E}[||X^T \mathbf{v}(t)||^2]} \quad \text{subject to } X^T X = I_Q$$



Convergence: Impact of bandwidth



Convergence: Impact of constraint bounds



V - The DASF toolbox

DASF toolbox

Purpose:

- Experiment with the DASF algorithm on your favorite spatial filter

Entry-Point:

A function expecting the following inputs:

- Function containing your **solver** for the centralized spatial filter design (i.e., function)
- Input signals
- Network topology
- Sequence of updating nodes
- Number of channels and output filters

(*)

(*) can also be automatically generated

Several examples available:

- PCA/GEVD
- MMSE
- TRO
- QCQP
- SCQP
- LCMV
- CCA
- **Easily add your own!**

Available for both MATLAB and Python

http://tinyurl.com/DASFToolbox

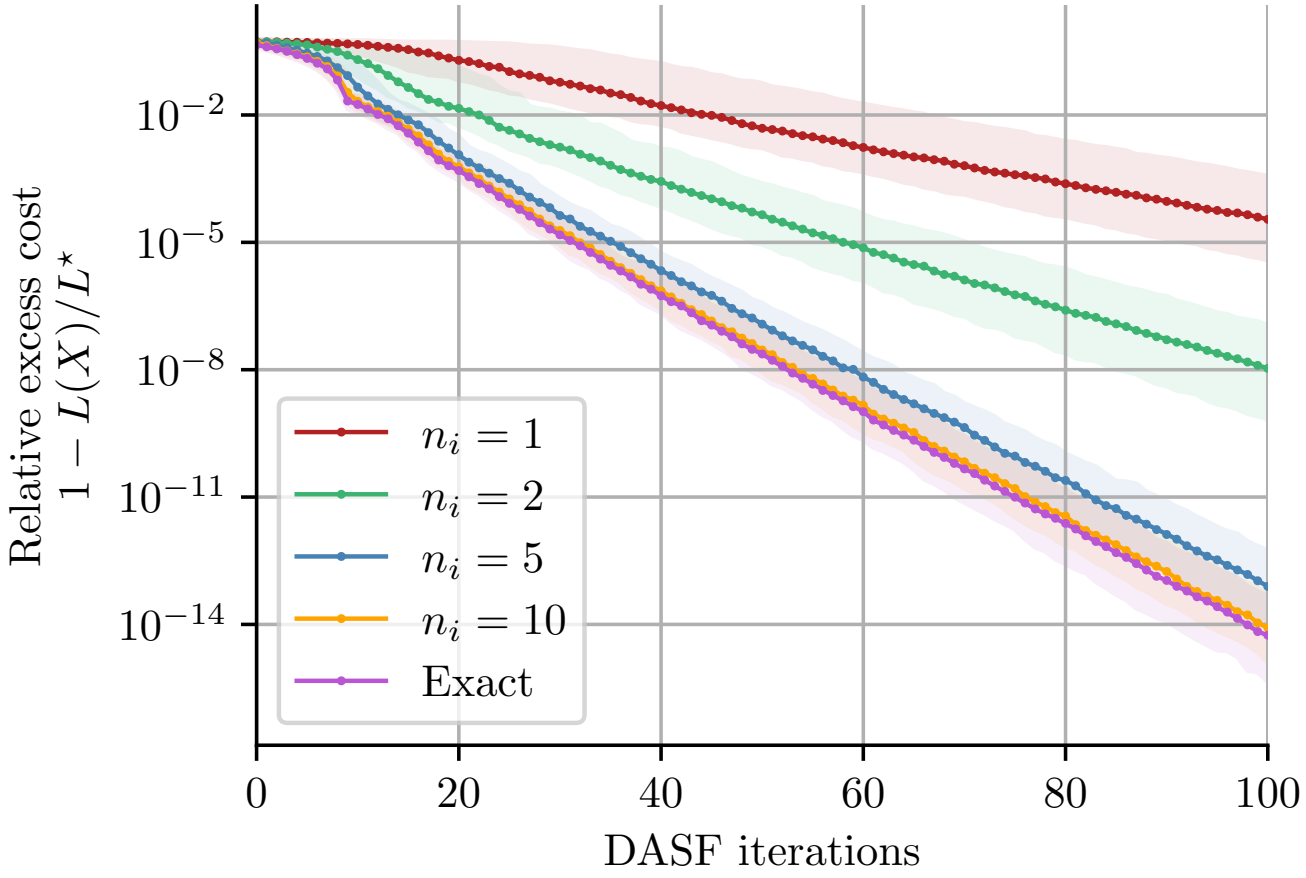
Scan me

VI - Extensions

Inexact solvers

- Only solve the local problems partially (e.g. apply only a few gradient descent steps, only a few power iterations, etc.)
- Convergence proof for a family of solvers that contains many commonly used descent methods
- Reduces computational cost

Inexact solvers



Non-smooth problems

- Convergence and optimality has also been shown for **non-smooth problems**

$$\begin{aligned} \underset{X}{\text{minimize}}\ & f(X^T\mathbf{y}(t)) + g(X^T\mathbf{\Gamma}) \\ \text{subject to}\ & l_j(X_k^T\mathbf{y}_k(t)) = 0, \\ & h_j(X_k^T\mathbf{y}_k(t)) \leq 0 \end{aligned}$$

- **Non-smooth term** must be **node-separable**, i.e. $g(X^T\mathbf{\Gamma}) = \sum_k g(X_k^T\mathbf{\Gamma}_k)$
- Limitations:
  - **Constraints** must also be **node-separable**
  - In arbitrary networks, constraints can depend on B **only**, not $\mathbf{y}(t)$

(satisfied for l_1, l_2, $l_{1,2}$, Frobenius norm, …)

Non-smooth problems
Examples

- Block-sparse generalized CCA (SUMCORR variant):

$$\min_X \mathbb{E}[\mathrm{tr}(X^T \mathbf{y}(t)\mathbf{y}(t)^T X)] + \sum_k ||X_k||_{2,1}$$
$$\text{s.t.} \quad \mathbb{E}[X_k^T \mathbf{y}_k(t)\mathbf{y}_k(t)^T X_k] = I_Q \quad \forall k \in \mathcal{K}$$

- Block-sparse regularized MMSE / MWF

$$\min_X \mathbb{E}[||X^T \mathbf{y}(t) - \mathbf{d}(t)||_F^2] + \sum_k ||X_k||_F$$
$$\text{s.t.} \quad \mathbb{E}[||X_k^T \mathbf{y}_k(t)||_F^2] \leq P_k \quad \forall k \in \mathcal{K}$$

Other extensions

- Tracking improvements
  - Efficiently re-use data from previous iterations for improved accuracy
- Parallel updates
  - Nodes (or subsets of nodes) can update simultaneously
- Node-specific problems
  - Each node wishes to solve a **node-specific** problem
- Fractional programs
  - Special version of DASF for fractional programs with improved computational efficiency

VII - Conclusion

Conclusions

- Generic algorithmic framework for **distributed** & **adaptive** spatial filtering
- Covers large number of optimization problems including PCA, GEVD, CCA, MMSE, LCMV, TRO, MAXVAR, SUMCORR, sparsity norms, etc.
- **Convergence guarantees** under mild assumptions generally satisfied in practice
- Various extensions

Authors

Cem Musluoglu

Charles Hovine

Alexander Bertrand

Bibliography: key papers

[Musluoglu et al. 2023a] C. A. Musluoglu and A. Bertrand, "A Unified Algorithmic Framework for Distributed Adaptive Signal and Feature Fusion Problems—Part I: Algorithm Derivation," in *IEEE Transactions on Signal Processing*, vol. 71, pp. 1863-1878, 2023

[Musluoglu et al. 2023b] C. A. Musluoglu, C. Hovine and A. Bertrand, "A Unified Algorithmic Framework for Distributed Adaptive Signal and Feature Fusion Problems — Part II: Convergence Properties," in *IEEE Transactions on Signal Processing*, vol. 71, pp. 1879-1894, 2023

Bibliography: DASF extensions

[Hovine et al. 2024] C. Hovine and A. Bertrand, "Distributed Adaptive Spatial Filtering with Inexact Local Solvers," *arXiv preprint arXiv:2405.03277*, 2024.

[Hovine et al. 2023] C. Hovine and A. Bertrand, "A Distributed Adaptive Algorithm for Non-Smooth Spatial Filtering Problems," *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Rhodes Island, Greece, 2023, pp. 1-5.

[Musluoglu et al. 2023a] C. A. Musluoglu and A. Bertrand, "A Distributed Adaptive Algorithm for Node-Specific Signal Fusion Problems in Wireless Sensor Networks," *2023 31st European Signal Processing Conference (EUSIPCO)*, Helsinki, Finland, 2023, pp. 1654-1658.

[Musluoglu et al. 2023b] C. A. Musluoglu and A. Bertrand, "A Computationally Efficient Algorithm for Distributed Adaptive Signal Fusion Based on Fractional Programs," *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Rhodes Island, Greece, 2023, pp. 1-5.

[Musluoglu et al. 2022] C. A. Musluoglu, M. Moonen and A. Bertrand, "Improved Tracking for Distributed Signal Fusion Optimization in a Fully-Connected Wireless Sensor Network," *2022 30th European Signal Processing Conference (EUSIPCO)*, Belgrade, Serbia, 2022, pp. 1836-1840

Bibliography (I)

[Doclo et al. 2009] S. Doclo, M. Moonen, T. Van den Bogaert and J. Wouters, "Reduced-Bandwidth and Distributed MWF-Based Noise Reduction Algorithms for Binaural Hearing Aids," in IEEE Trans. Audio, Speech, and Language Processing, vol. 17, no. 1, pp. 38-51, 2009.

[Musluoglu et al. 2023a] C. A. Musluoglu and A. Bertrand, "A Unified Algorithmic Framework for Distributed Adaptive Signal and Feature Fusion Problems—Part I: Algorithm Derivation," in *IEEE Trans. Signal Processing*, vol. 71, pp. 1863-1878, 2023.

[Musluoglu et al. 2023b] C. A. Musluoglu, C. Hovine and A. Bertrand, "A Unified Algorithmic Framework for Distributed Adaptive Signal and Feature Fusion Problems — Part II: Convergence Properties," in *IEEE Trans. Signal Processing*, vol. 71, pp. 1879-1894, 2023.

[Bertrand et al. 2011] A. Bertrand, and M. Moonen, "Distributed LCMV beamforming in wireless sensor networks with node-specific desired signals." *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2011.

[Zeng et al. 2013] Y. Zeng and Richard C. Hendriks. "Distributed delay and sum beamformer for speech enhancement via randomized gossip." *IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 22, no. 1* 22.1, pp. 260-273, 2013.

Bibliography (II)

[Li et al. 2011] L. Li, A. Scaglione, and J. H. Manton. "Distributed principal subspace estimation in wireless sensor networks." IEEE Journal of Selected Topics in Signal Processing vol. 5, no.4, pp725-738, 2011.

[Mota et al. 2011] J.F.C Mota, et al. "Distributed basis pursuit." IEEE Transactions on Signal Processing vol. 60, pp. 1942-1956, 2011.

[Markovich et al. 2015] S. Markovich-Golan, et al. "Optimal distributed minimum-variance beamforming approaches for speech enhancement in wireless acoustic sensor networks." Signal Processing, vol.107, pp. 4-20, 2015.

[Fu et al. 2016] X. Fu, et al. "Efficient and distributed algorithms for large-scale generalized canonical correlations analysis." 2016 IEEE 16th international conference on data mining (ICDM). IEEE, 2016.

[Bertrand et al. 2018] A. Bertrand, and M. Moonen. "Distributed canonical correlation analysis in wireless sensor networks with application to distributed blind source separation." IEEE Transactions on Signal Processing vol. 63, no.18, pp. 4800-4813, 2018.

[Zhang et al. 2019] J. Zhang, et al. "Distributed rate-constrained LCMV beamforming." IEEE Signal Processing Letters, vol. 26, no. 5, pp. 675-679, 2019.

[Musluoglu et al. 2021] C.A. Musluoglu, and A. Bertrand. "Distributed adaptive trace ratio optimization in wireless sensor networks." IEEE Transactions on Signal Processing, vol. 69 pp. 3653-3670, 2021.