Apprentissage et traitement adaptatif de l'information dans les réseaux de capteurs sans fil

Cédric Richard

Laboratoire H. Fizeau, UMR 6525, Observatoire de la Côte d'Azur
Université de Nice Sophia-Antipolis

Collaborations

This work was conducted in collaboration with

- Jose Bermudez (Univ. Santa Catarina, Brazil)
- Paul Honeine (Univ. Tech. Troyes)
- Hichem Snoussi (Univ. Tech. Troyes)

and a panel of PhD students

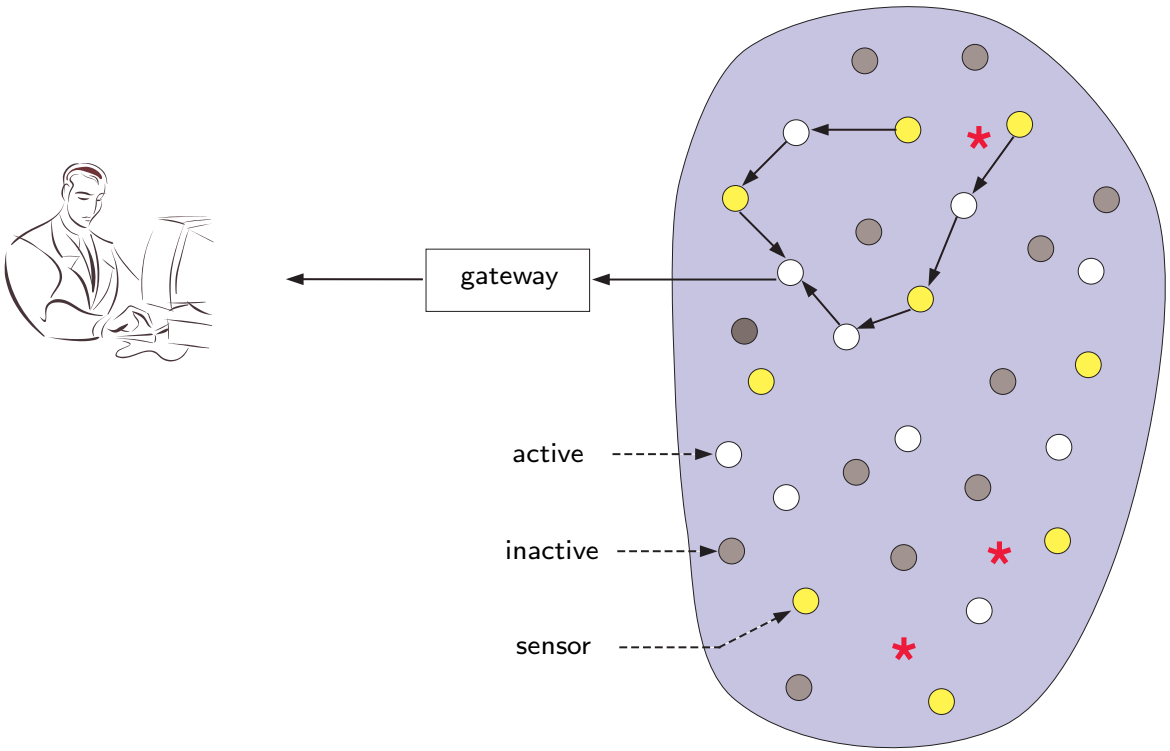
- Mehdi Essoloh (Univ. Tech. Troyes)
- Farah Mourad (Univ. Tech. Troyes)
- Wemerson D. Parreira (Univ. Santa Catarina, Brazil)
- Jing Tang (Univ. Tech. Troyes)
- ...

Issues and challenges in wireless sensor networks

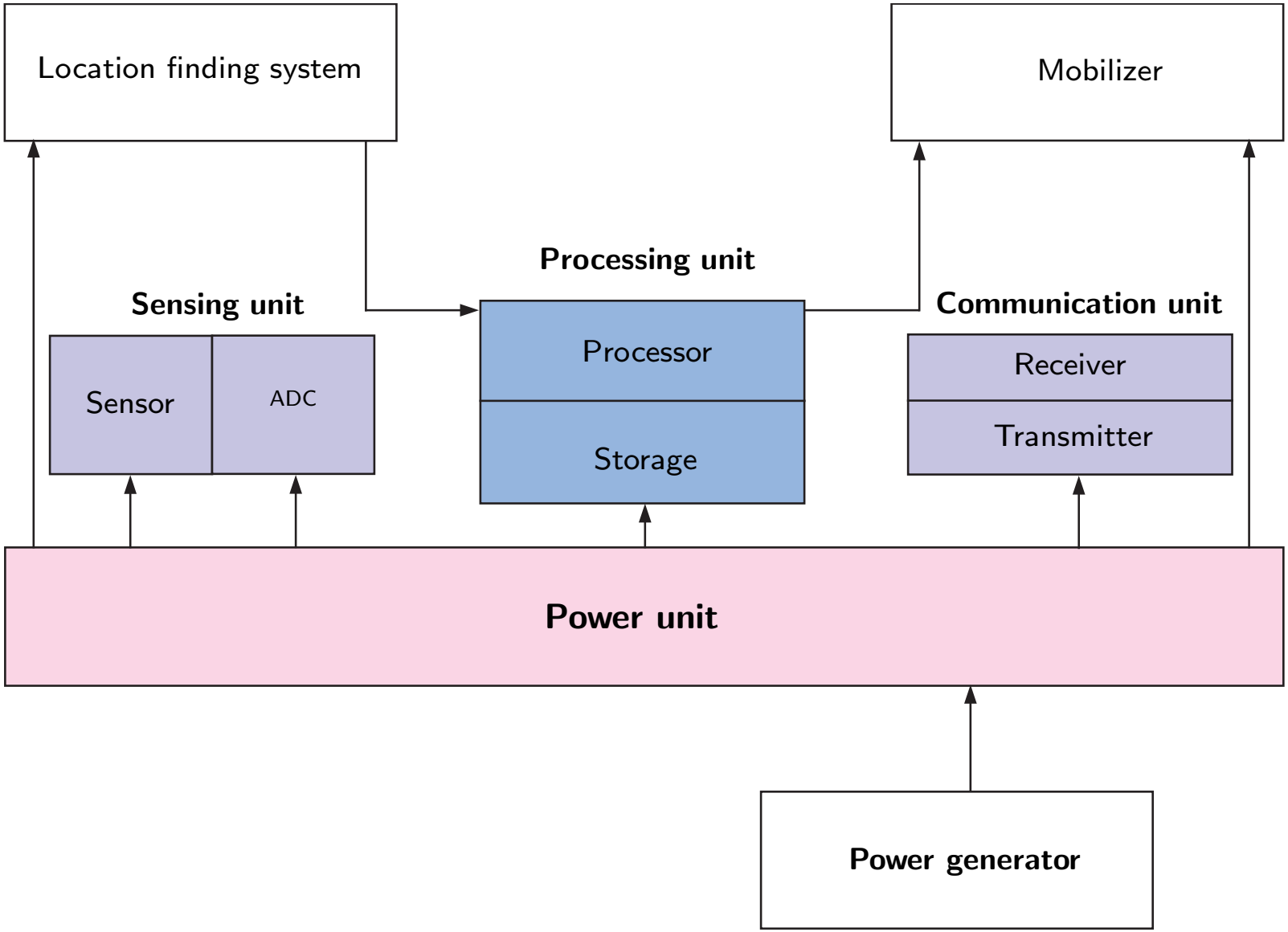
Definition

"A wireless sensor network is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or polluants, at different locations."

"One of the 10 technologies that will change the world", MIT Technology Review, 2003.



Issues and challenges in wireless sensor networks



Issues and challenges in wireless sensor networks

1. Limited CPU
   - Slow (8 MHz)
   - 512-point FFT takes 450 ms, ...
2. Limited memory
   - 10 KB of RAM, 60 KB of program ROM
   - Much of this taken up by system software
3. Potentially lots of storage
   - Some designs support up to 2 GB of MicroSD flash
   - But, expensive to access: 13 ms to read/write a 512-byte block; $\approx$ 25 mA
4. Low-power radio
   - 802.15.4 best case performance: 100 Kbps or so (single node transmitting, no interference, short range)
   - Approximatively 50 m range, and very unreliable

Issues and challenges in wireless sensor networks

LWIM III
(UCLA, 1996)

WeC
(Berkeley, 1999)

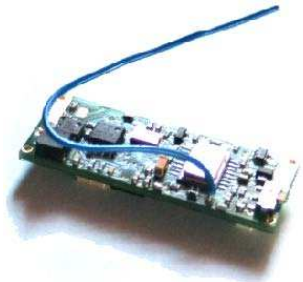
Rene
(Berkeley, 2000)

Dot
(Xbow, 2001)

Mica2
(Xbow, 2002)

Telos
(Moteiv, 2004)

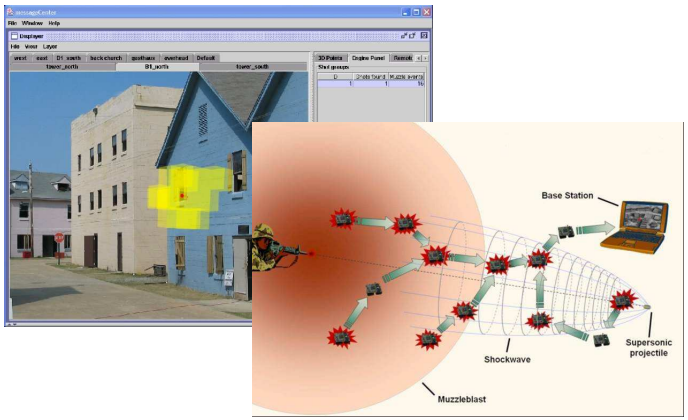
Tmote
(Moteiv, 2005)



pParticle
(Particle Comp., 2006)

Issues and challenges in wireless sensor networks

1. Military applications
   - monitoring forces, equipment and ammunition
   - battlefield surveillance, target tracking, NBC attack detection, ...



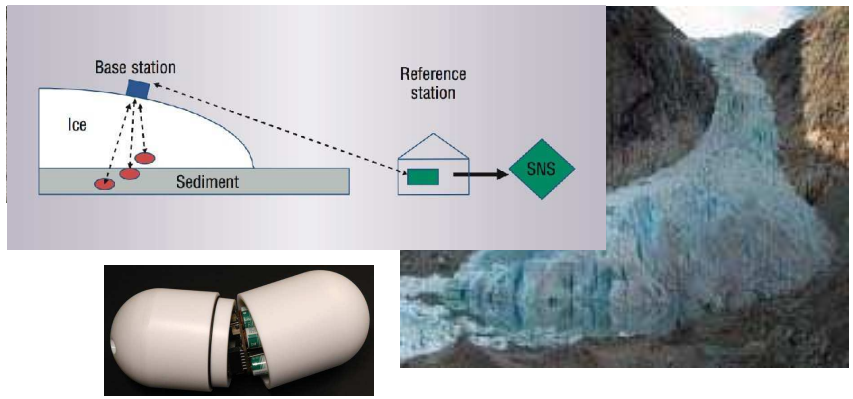
Gunshot detection

Vehicles tracking

Issues and challenges in wireless sensor networks

1. Military applications
   - monitoring forces, equipment and ammunition
   - battlefield surveillance, target tracking, NBC attack detection, ...
2. Environmental monitoring
   - precision agriculture, habitat modeling
   - flood and bushfire warning, ice caps and glaciers monitoring, ...

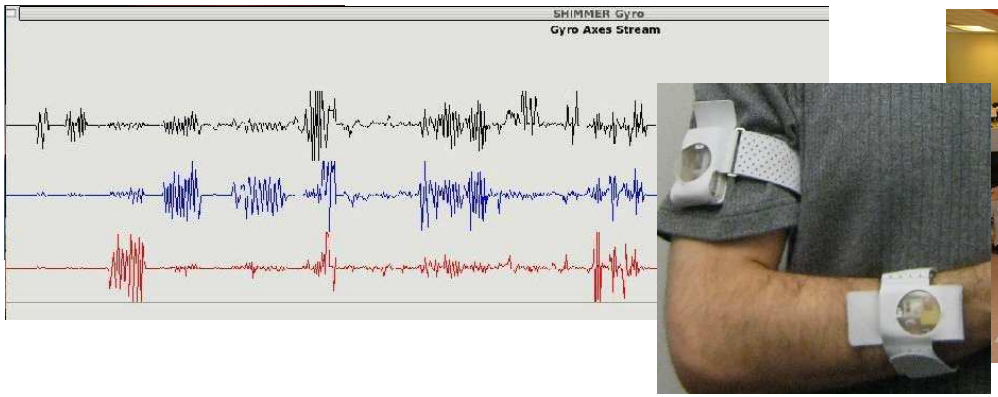
Forest fire detection



Glacier monitoring

Issues and challenges in wireless sensor networks

1. Military applications
   - monitoring forces, equipment and ammunition
   - battlefield surveillance, target tracking, NBC attack detection, ...
2. Environmental monitoring
   - precision agriculture, habitat modeling
   - flood and bushfire warning, ice caps and glaciers monitoring, ...
3. Health applications
   - physiological data monitoring, patients and doctors tracking
   - fall and unconsciousness detection, ...
4. And many others...



Neuromotor disease assessement

Emergency medical care

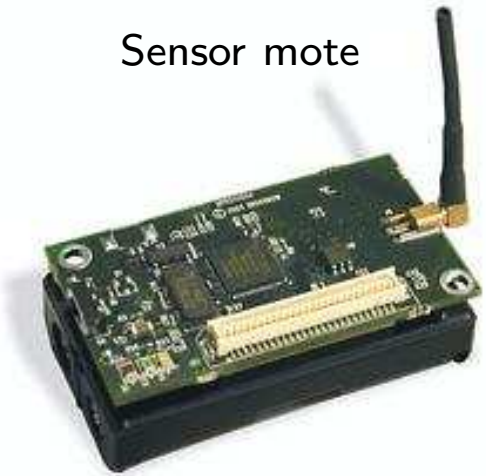
Issues and challenges in wireless sensor networks

Scientific top challenges include

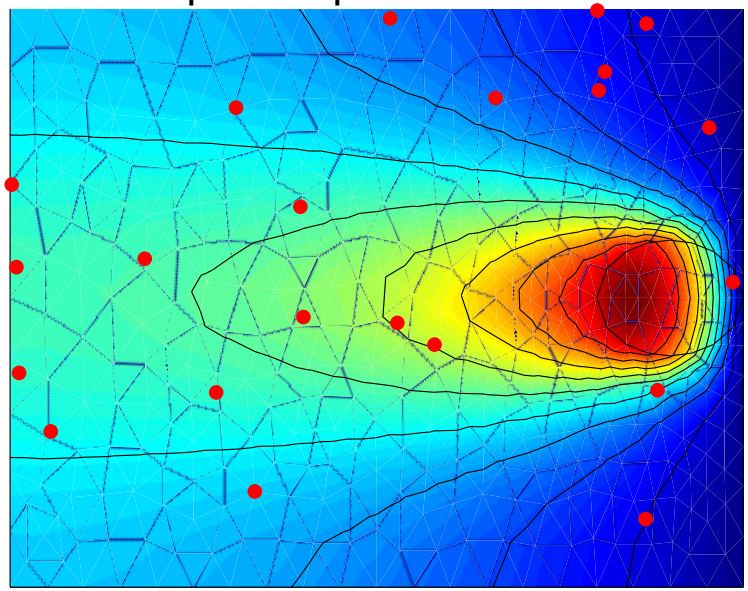
- Limited power they can harvest or store
- Dynamic network topology
- Network control and routing
- Self-organization and localization
- Heterogeneity of nodes
- Large scale of deployment
- Collaborative signal and information processing
- Ability to cope with node failures
- Ability to withstand harsh environmental conditions
- Miniaturization and low cost
- ...

Nonparametric data processing

Sensor mote



Anisotropic temperature distribution



One wish to estimate a function ψ of the location x that models the physical phenomenon, based on position-measurement data.

- Parametric approaches assume that sufficient **data** or **prior knowledge** is available to specify a statistical model
- These assumptions may not be satisfied in anticipated applications
- Nonparametric methods – machine learning – have been widely successful in centralized strategies

Our goal is to investigate robust nonparametric methods for **distributed inference**

Nonparametric data processing

The model we consider is as follows

- N sensors distributed over the plane, located at $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$
- Sensor i measures the field locally, and $d_i = \psi(\boldsymbol{x}_i) + \varepsilon_i$
- Sensor network topology specified by a graph
- Find a global estimate for field, i.e., at locations not yet occupied by a sensor

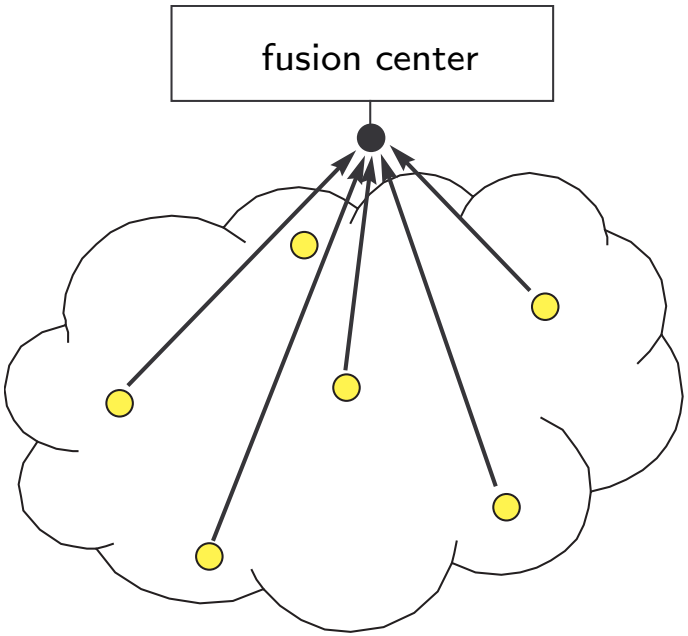
Nonparametric data processing

Let $d_{i,j}$ denote the j-th measurement taken at the i-th sensor. We would like to solve a problem of the form

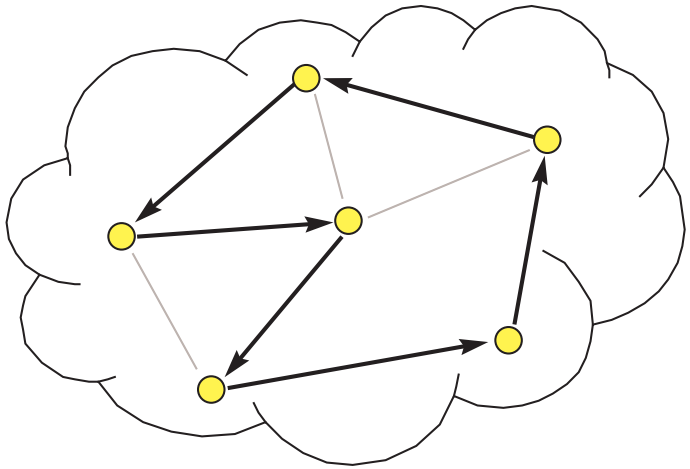
$$\psi^o = \arg\min_{\psi \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} J(\psi(\boldsymbol{x}_i), \{d_{i,j}\}_{j=1}^{M})$$

Two major learning strategies can be considered

- learning with a fusion center [Nguyen, 2005], ...
- distributed learning with in-network processing [Rabbat, 2004], [Predd, 2005], ...



$\mathcal{E} \propto MN^{1+1/d}$

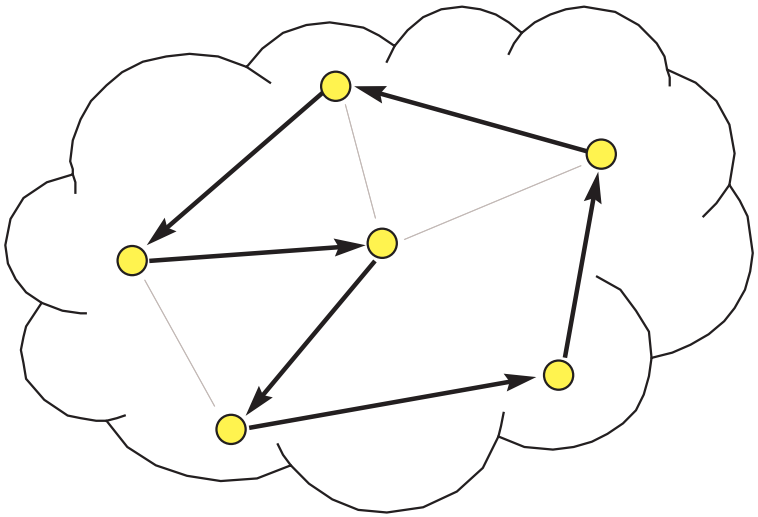


$\mathcal{E} \propto N\epsilon^{-2}$

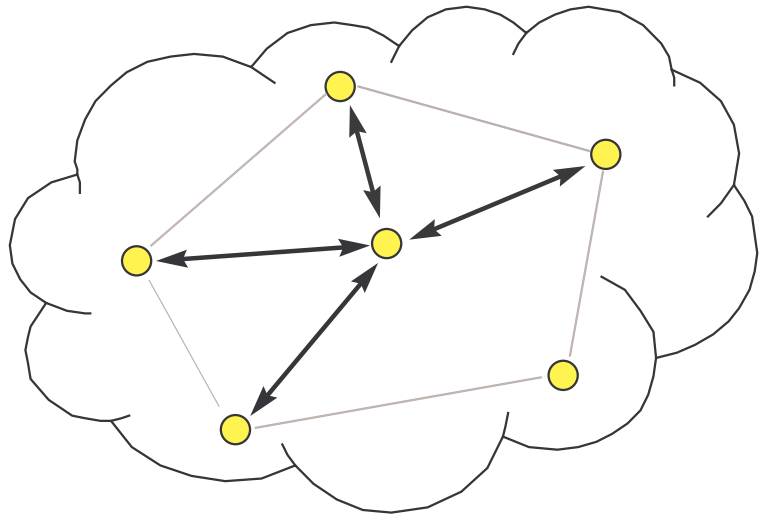
Nonparametric data processing

In-network processing suggests two major cooperation strategies

- incremental mode of cooperation [Rabbat, 2004], [Sayed, 2005], ...
- diffusion mode of cooperation [Predd, 2005], [Sayed, 2006], ...

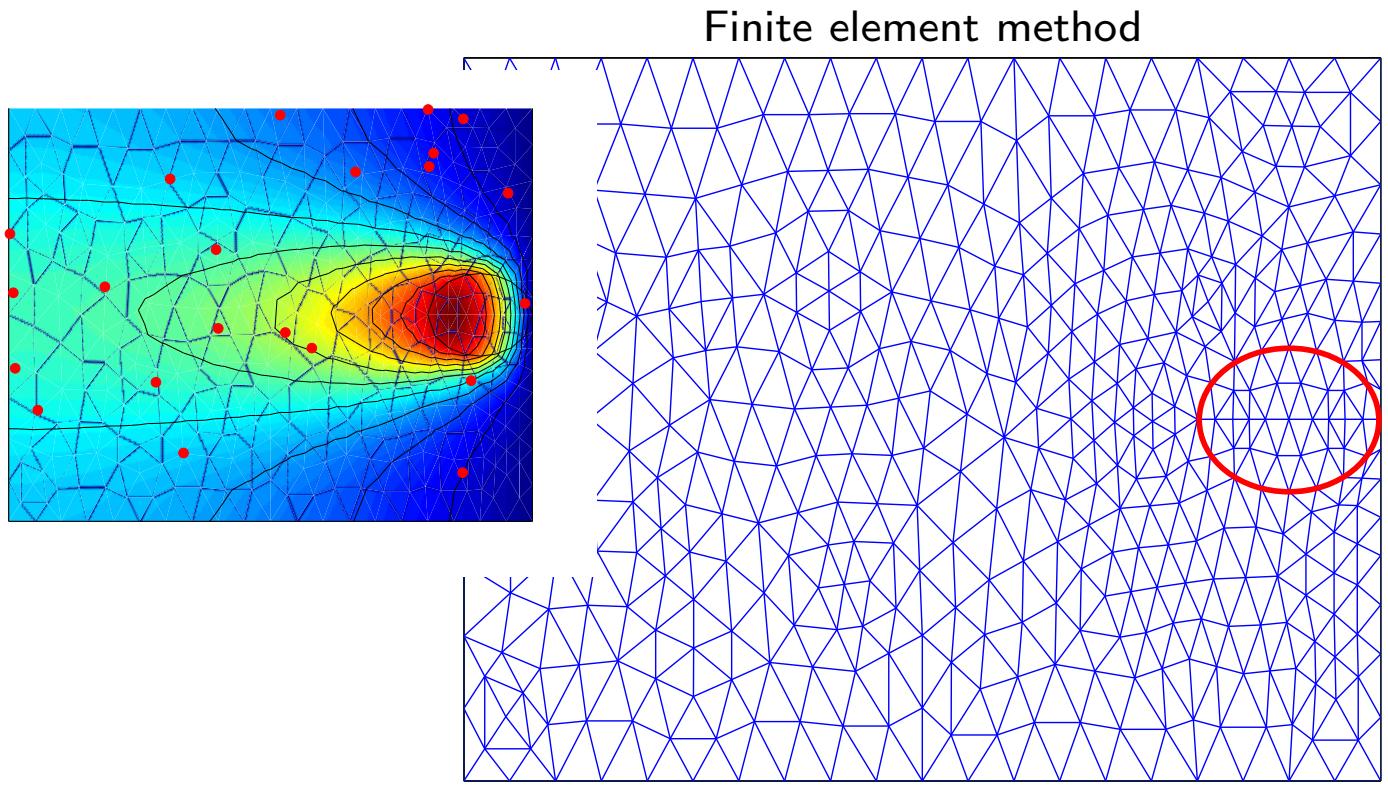


Incremental



Diffusion

Nonparametric data processing



Monitoring complex phenomena, e.g., diffusion, requires that the following questions be addressed:

1. online estimation of its parameters, sparseness of data
2. nonlinearity of the model
3. in-network processing strategy

The least-mean square problem

Let $\{(\boldsymbol{x}_n, d_n)\}_{n=1,\ldots,N}$ be a set of training samples drawn i.i.d. according to an unknown pdf, where d_n is a scalar measurement and $\boldsymbol{x}_n$ is a regression vector.

Problem setup

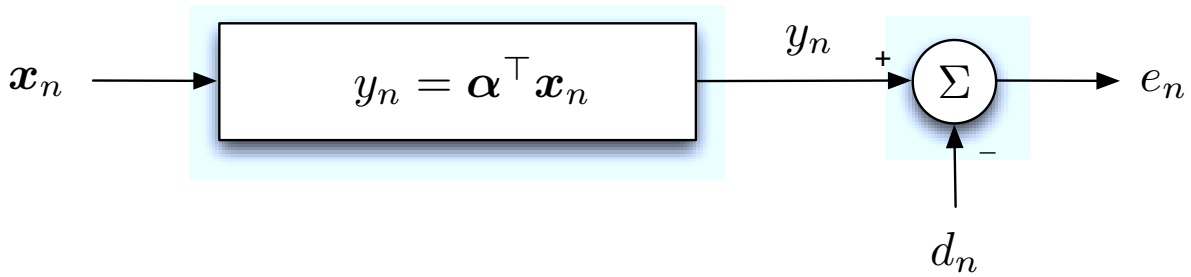
The goal is to solve the least-mean square problem

$$\boldsymbol{\alpha}^o = \arg\min_{\boldsymbol{\alpha}} E|\boldsymbol{\alpha}^\top \boldsymbol{x}_n - d_n|^2$$

The optimum $\boldsymbol{\alpha}^o$ is the solution of the *Wiener-Hopf equations*

$$\boldsymbol{R}_x \, \boldsymbol{\alpha}^o = \boldsymbol{r}_{dx}$$

where $\boldsymbol{R}_x = E\, \boldsymbol{x}_n \boldsymbol{x}_n^\top$, $\quad \boldsymbol{r}_{dx} = E\, \boldsymbol{x}_n d_n$

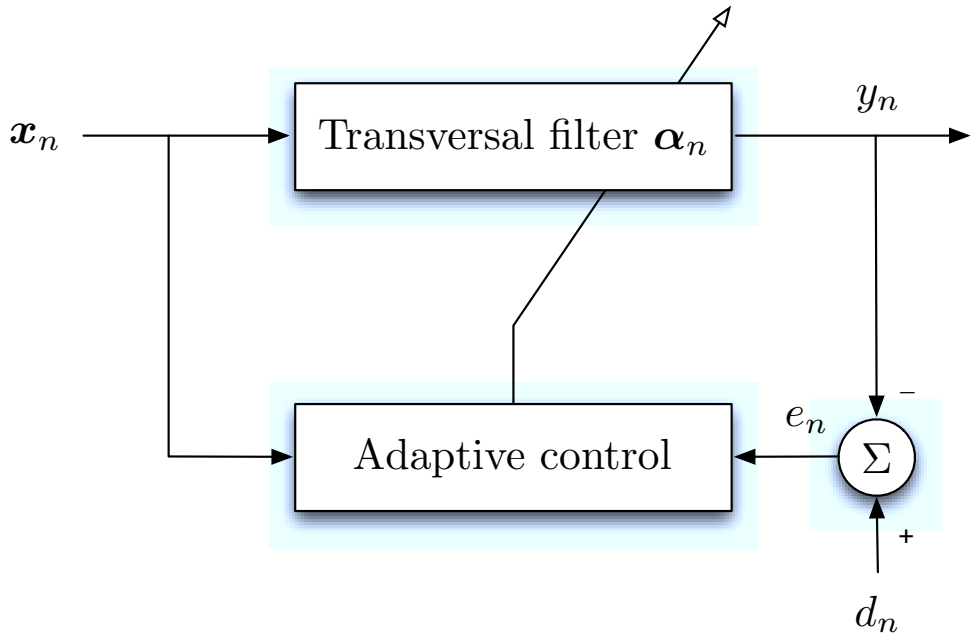


Designing adaptive filters

Adaptive filters are equipped with a built-in mechanism that enable them to adjust their free parameters automatically in response to statistical variations.

The traditional class of supervised adaptive filters rely on *error-correction learning* for their adaptive capability

$$\boldsymbol{\alpha}_n = \boldsymbol{\alpha}_{n-1} + e_n \boldsymbol{G}_n$$



Least-mean-square algorithm

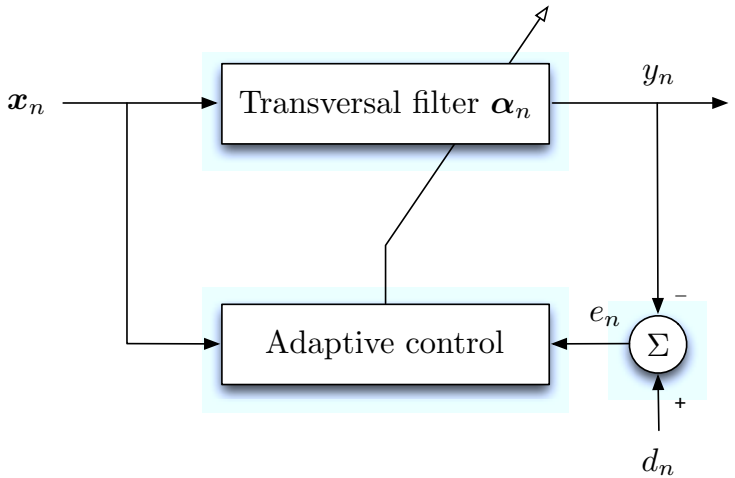
The most commonly used form of an adaptive filtering algorithm is the so-called LMS algorithm, which operates by minimizing the instantaneous cost function

$$\begin{aligned} J(n) &= \frac{1}{2} e_n^2 \\ &= \frac{1}{2} (d_n - \boldsymbol{\alpha}_{n-1}^\top \boldsymbol{x}_n)^2 \end{aligned}$$

The LMS algorithm

The instantaneous gradient vector is given by $\nabla J(n) = -e_n \, \boldsymbol{x}_n$. Following the instantaneous version of the method of gradient descent yields the LMS algorithm

$$\boldsymbol{\alpha}_n = \boldsymbol{\alpha}_{n-1} + \mu \, e_n \, \boldsymbol{x}_n$$



Recursive least-squares algorithm

The RLS algorithm follows a rationale similar to the LMS algorithm in that they are both examples of error-correction learning. The cost function is now defined by

$$J(n) = \sum_{i=1}^{n} \lambda^{n-i} (d_i - \boldsymbol{\alpha}^\top \boldsymbol{x}_i)^2$$

This means that, at each time instant n, the estimate $\boldsymbol{\alpha}_n$ provides a summary of all the data processed from $i = 0$ up to and including time $i = n$.

The RLS algorithm

Accordingly, the updated estimate of the actual parameter vector $\boldsymbol{\alpha}_n$ in the RLS algorithm is defined by

$$\boldsymbol{\alpha}_n = \boldsymbol{\alpha}_{n-1} + e_n \, \boldsymbol{R}_n^{-1} \, \boldsymbol{x}_n$$

where

$$\boldsymbol{R}_n = \sum_{i=1}^{n} \lambda^{n-i} \, \boldsymbol{x}_i \, \boldsymbol{x}_i^\top$$

In order to generate the coefficient vector we are interested in the inverse of $\boldsymbol{R}_n$. For that task, the Woodbury matrix identity comes in handy

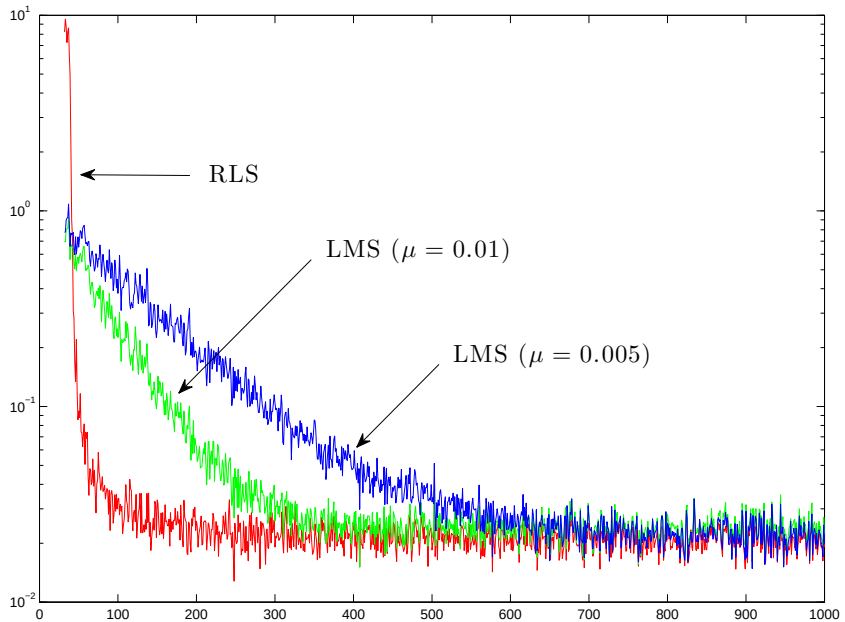
$$\begin{aligned} \boldsymbol{R}_n^{-1} &= \lambda^{-1} \boldsymbol{R}_{n-1}^{-1} - \boldsymbol{g}_n \boldsymbol{x}_n^\top \lambda^{-1} \boldsymbol{R}_{n-1}^{-1} \\ \boldsymbol{g}_n &= \boldsymbol{R}_{n-1}^{-1} \boldsymbol{x}_n \left\{ \lambda + \boldsymbol{x}_n^\top \boldsymbol{R}_{n-1}^{-1} \boldsymbol{x}_n \right\}^{-1} \end{aligned}$$

Experimental comparison

In closing this brief discussion of the LMS and RLS algorithm, we may compare their characteristics as follows:

- Complexity of the LMS algorithm scales linearly with $\dim(\alpha)$, whereas the complexity of the RLS algorithm follows a square law.
- Being model independent, the LMS algorithm is more robust.
- The convergence rate of the RLS algorithm is faster than the LMS algorithm.

Example: identification of a 32-order FIR filter with 16 dB additive gaussian noise.



Functional estimation with reproducing kernels

Consider a set $\{(\boldsymbol{x}_n, d_n)\}_{n=1,\ldots,N}$ of training samples drawn i.i.d. according to an unknown pdf, where d_n is a scalar measurement and $\boldsymbol{x}_n$ is a regression vector.

Problem formulation

$$\psi^o = \arg\min_{\psi \in \mathcal{H}} E|\psi(\boldsymbol{x}_n) - d_n|^2$$

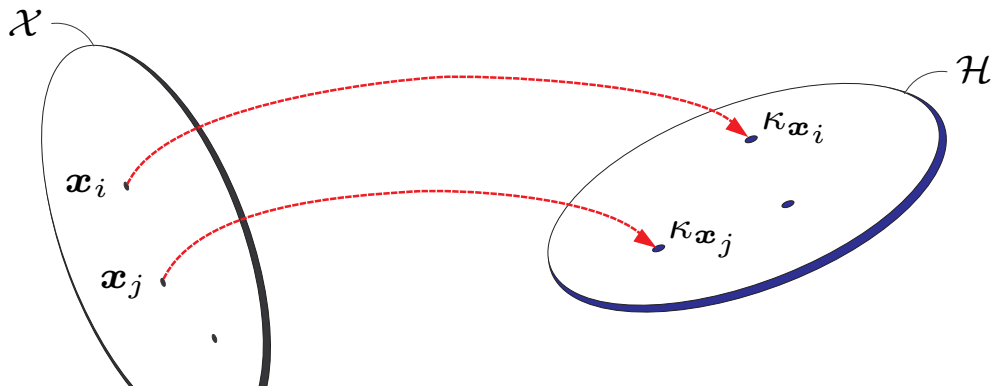
In a RKHS

$$\psi^o = \arg\min_{\psi \in \mathcal{H}} E|\langle \psi, \kappa_{\boldsymbol{x}_n} \rangle_{\mathcal{H}} - d_n|^2$$

Definition (Reproducing kernels and RKHS)

Let $(\mathcal{H}, \langle \cdot\ , \cdot \rangle_{\mathcal{H}})$ denote a Hilbert space of real-valued functions defined on $\mathcal{X}$. The function $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ from $\mathcal{X} \times \mathcal{X}$ to $\mathbb{R}$ is the reproducing kernel of $\mathcal{H}$ if, and only if,

- the function $\kappa(\boldsymbol{x}_i, \cdot)$, denoted hereafter by $\kappa_{\boldsymbol{x}_i}$, belongs to $\mathcal{H}$ for all $\boldsymbol{x}_i \in \mathcal{X}$
- $\psi(\boldsymbol{x}_n) = \langle \kappa_{\boldsymbol{x}_n}, \psi \rangle_{\mathcal{H}}$ for all $\boldsymbol{x}_n \in \mathcal{X}$ and $\psi \in \mathcal{H}$



Functional estimation with reproducing kernels

Remark 1 Replacing ψ by $\kappa_{\boldsymbol{x}_j}$ in $\psi(\boldsymbol{x}_i) = \langle \kappa_{\boldsymbol{x}_i}, \psi \rangle_{\mathcal{H}}$ leads us to

$$\langle \kappa_{\boldsymbol{x}_i}, \kappa_{\boldsymbol{x}_j} \rangle_{\mathcal{H}} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

this being the origin of the term *reproducing kernel*.

Remark 2 The Moore-Aronszajn theorem (1950) states that for every positive definite function $\kappa(\cdot\,,\cdot)$ on $\mathcal{X} \times \mathcal{X}$, namely,

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0, \qquad \forall \boldsymbol{x}_i \in \mathcal{X}, \quad \forall \alpha_i \in \mathbb{R}$$

there exists a unique RKHS and vice versa.

Examples

Polynomial	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle_{\mathcal{H}} + \beta)^p$	$\beta \geq 0,\ q \in \mathbb{N}$
Laplace	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\gamma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|\right)$	$\gamma > 0$
Gaussian	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 / 2\sigma_0^2\right)$	$\sigma_0 > 0$

Functional estimation with reproducing kernels

Consider the set $\{(\boldsymbol{x}_n, d_n)\}_{n=1,\ldots,N}$ of i.i.d. training samples.

Least-mean square problem

$$\psi^o = \arg\min_{\psi \in \mathcal{H}} E|\langle \psi, \kappa_{\boldsymbol{x}_n} \rangle_{\mathcal{H}} - d_n|^2$$

Representer theorem

$$\psi^o = \sum_{i=1}^{N} \alpha_i^o \, \kappa_{\boldsymbol{x}_i}$$

Dual problem

The optimum $\boldsymbol{\alpha}^o = [\alpha_1^o \ldots \alpha_N^o]^\top$ is the solution of the *normal equations*

$$\boldsymbol{R}_x \, \boldsymbol{\alpha}^o = \boldsymbol{r}_{dx}$$

where $\boldsymbol{R}_x = E\, \boldsymbol{k}_n \boldsymbol{k}_n^\top, \quad \boldsymbol{r}_{dx} = E\, \boldsymbol{k}_n d_n$
$\boldsymbol{k}_n = [\kappa(\boldsymbol{x}_n, \boldsymbol{x}_1) \ \ldots \ \kappa(\boldsymbol{x}_n, \boldsymbol{x}_N)]^\top$

Remark: we shall suppose that $E\, \boldsymbol{k}_n = 0$ and $E\, d_n = 0$

The order N of the kernel expansion model is the number of available data ...

Nonlinear adaptive filters

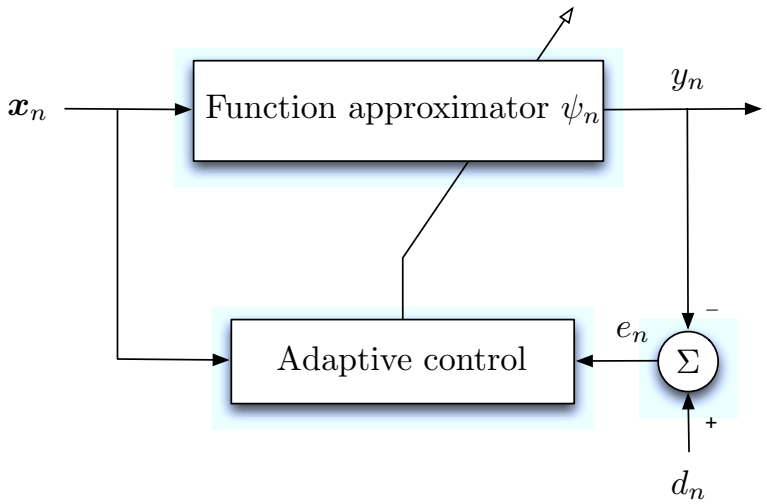
The limited computational power of linear learning machines was first highlighted in [Minsky, 1969] in a famous work on perceptrons.

The problem of designing nonlinear adaptive filters has been addressed by

- cascading a static nonlinearity with a linear filter such as in Hammerstein and Wiener models [Wiener, 1958], [Billings, 1982]
- using Volterra series [Gabor, 1968]
- designing multilayer perceptrons, etc. [Lang, 1988], [Principe, 1992]

We shall now consider adaptive filters that possesses the ability of modeling nonlinear mappings $y = \psi(\boldsymbol{x})$ and obeys the following sequential learning rule

$$\psi_n = \psi_{n-1} + e_n \boldsymbol{G}_n$$



Basic kernel LMS algorithm

To overcome the limitation of linearity of the LMS algorithm, we shall ow formulate a similar algorithm that can learn arbitrary nonlinear mappings

The LMS algorithm

$$\boldsymbol{\alpha}^o = \arg\min_{\boldsymbol{\alpha}} E|\boldsymbol{\alpha}^\top \boldsymbol{x}_n - d_n|^2$$

Initialization
$\boldsymbol{\alpha}_0 = [0 \ldots, 0]$

While $\{\boldsymbol{x}_n, d_n\}$ available
$e_n = d_n - \boldsymbol{\alpha}_{n-1}^\top \boldsymbol{x}_n$
$\boldsymbol{\alpha}_n = \boldsymbol{\alpha}_{n-1} + \mu\, e_n\, \boldsymbol{x}_n$

In a RKHS

$$\psi^o = \arg\min_{\psi \in \mathcal{H}} E|\langle \psi, \kappa_{\boldsymbol{x}_n} \rangle_{\mathcal{H}} - d_n|^2$$

Initialization
$\psi_0 = 0$

While $\{\boldsymbol{x}_n, d_n\}$ available
$e_n = d_n - \langle \psi_{n-1}, \kappa_{\boldsymbol{x}_n} \rangle_{\mathcal{H}}$
$\psi_n = \psi_{n-1} + \mu\, e_n\, \kappa_{\boldsymbol{x}_n}$

The KLMS algorithm is a simple method. However, we need to pay attention to how to cope with the growing memory and computation requirement for online operation.

Two variations on the basic kernel LMS algorithm

Normalized kernel LMS algorithm It is easy to derive the normalized KNLMS based on the NLMS algorithm [Richard, 2008]

$$\psi_n = \psi_{n-1} + \frac{\mu}{\varepsilon + \|\kappa_{\boldsymbol{x}_n}\|^2} e_n \, \kappa_{\boldsymbol{x}_n}.$$

Leaky kernel LMS algorithm In [Kivinen, 2004], the authors consider the following regularized problem

$$\psi^o = \arg \min_{\psi \in \mathcal{H}} E|\langle \psi, \kappa_{\boldsymbol{x}_n} \rangle_{\mathcal{H}} - d_n|^2 + \lambda \|\psi\|^2.$$

This leads us to the following update rule

$$\psi_n = (1 - \lambda\mu) \, \psi_{n-1} + \mu \, e_n \, \kappa_{\boldsymbol{x}_n}.$$

The regularization introduces a bias in the solution that is well known in leaky LMS, even for small λ [Sayed, 2003].

Basic kernel RLS algorithm

We formulate the recursive least-squares algorithm on $\{(\psi(\boldsymbol{x}_1), d_1), (\psi(\boldsymbol{x}_2), d_2), \ldots\}$. At each iteration, the minimizer $\psi_n \in \mathcal{H}$ of $J(n)$ needs to be determined recursively.

$$J(n) = \sum_{i=1}^{n} \lambda^{n-i} (\langle \psi, \kappa_{\boldsymbol{x}_i} \rangle_{\mathcal{H}} - d_i)^2 + R(\|\psi\|)$$

We obtain the following sequential learning rule for the KRLS algorithm

$$\psi_n = \psi_{n-1} + r_n^{-1} \left[\kappa_{\boldsymbol{x}_n} - \sum_{i=1}^{n-1} \alpha_{n,i} \kappa_{\boldsymbol{x}_i} \right] e_n$$

where the $\alpha_{n,j}$'s are the solution of a $(n-1) \times (n-1)$ linear system.

Let $\boldsymbol{Q}_{n-1}$ the matrix involved in the above-mentioned system. The block matrix inversion identity can be used to compute $\boldsymbol{Q}_n^{-1}$ from $\boldsymbol{Q}_{n-1}^{-1}$ since

$$\boldsymbol{Q}_n = \begin{pmatrix} \boldsymbol{Q}_{n-1} & \boldsymbol{q}_n \\ \boldsymbol{q}_n^\top & q_{n,n} \end{pmatrix} \quad \Longrightarrow \quad \boldsymbol{Q}_n^{-1} = r_n^{-1} \begin{pmatrix} \boldsymbol{Q}_{n-1}^{-1} r_n + \boldsymbol{z}_n \boldsymbol{z}_n^\top & -\boldsymbol{z}_n \\ -\boldsymbol{z}_n^\top & 1 \end{pmatrix}$$

with

$$\begin{aligned} \boldsymbol{z}_n &= \boldsymbol{Q}_{n-1}^{-1} \boldsymbol{q}_n \\ r_n &= q_{n,n} - \boldsymbol{z}_n^\top \boldsymbol{q}_n \end{aligned}$$

Sparse kernel-based adaptive filtering

Full-order model

$$\psi_n = \sum_{i=1}^{n} \alpha_{n,i}\, \kappa_{\boldsymbol{x}_i}$$

$\Rightarrow$

Reduced-order model

$$\psi_n = \sum_{i=1}^{m} \alpha_{n,i}\, \kappa_{\boldsymbol{x}_{\omega_i}}$$

Let us call $\mathcal{D}_m = \{\kappa_{\boldsymbol{x}_{\omega_1}}, \ldots, \kappa_{\boldsymbol{x}_{\omega_m}}\} \subset \{\kappa_{\boldsymbol{x}_1}, \ldots, \kappa_{\boldsymbol{x}_n}\}$ the dictionary.

The function $\psi_n = \sum_{i=1}^{m-1} \alpha_{n,i}\, \kappa_{\boldsymbol{x}_{\omega_i}} + \alpha_{n,m}\, \kappa_{\boldsymbol{x}_n}$ can be selected as the new model depending if, for instance, one of the following criteria is satisfied

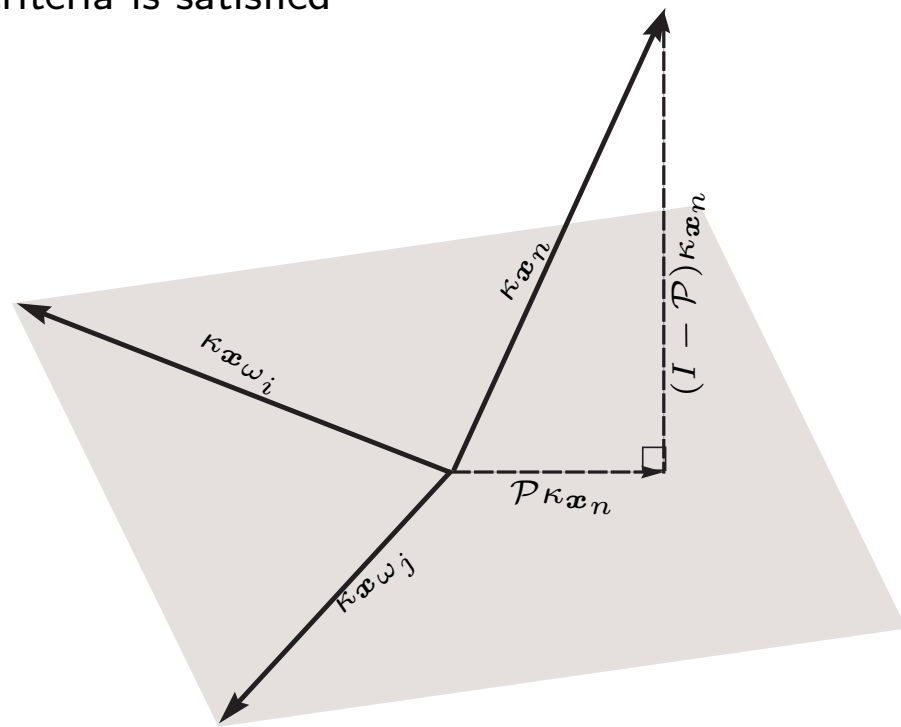
- Short-time criterion [Kivinen, 2004]
- Coherence criterion [Richard, 2008]

$$\max_{i=1,\ldots,m-1} \left|\langle \kappa_{\boldsymbol{x}_n}, \kappa_{\boldsymbol{x}_{\omega_i}} \rangle_{\mathcal{H}}\right| \leq \nu_0$$

- Approximate linear dependence [Engel, 2004]

$$\min_{\boldsymbol{\gamma}} \left\| \kappa_{\boldsymbol{x}_n} - \sum_{i=1}^{m-1} \gamma_i\, \kappa_{\boldsymbol{x}_{\omega_i}} \right\|_{\mathcal{H}}^2 > \eta_0^2$$

- Rényi's quadratic entropy [Suykens, 2002]



Sparse kernel-based adaptive filtering

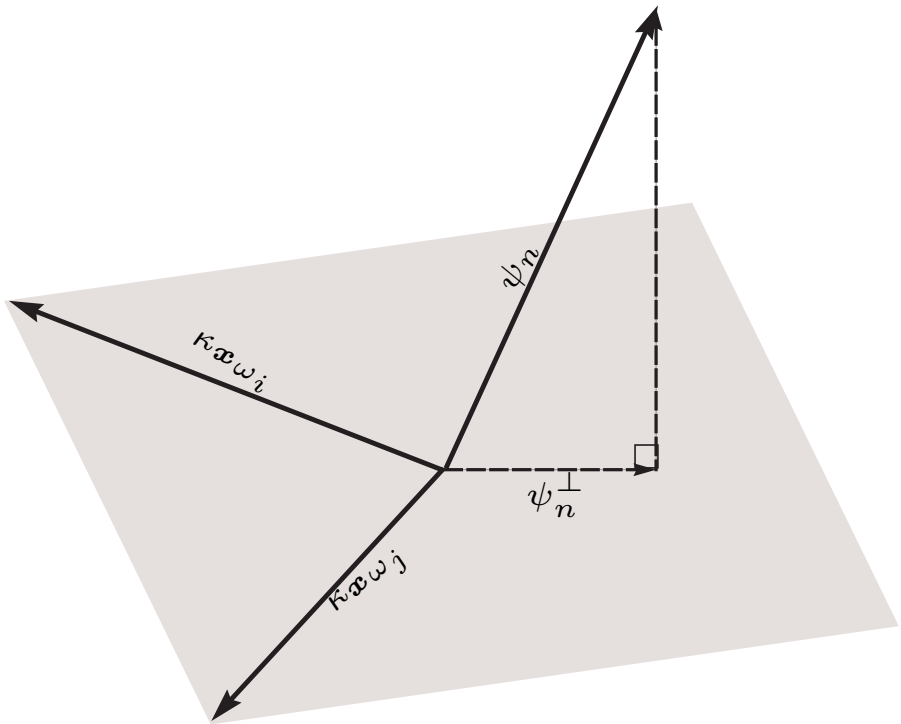
The previously mentioned criteria depends only on the inputs, and not on measurements or estimated error. Output-depend criteria exists. See, e.g., the adapt-then-project sparsification rule [Dodd, 2003].

The function $\psi_n = \sum_{i=1}^{m-1} \alpha_{n,i}\, \kappa_{\boldsymbol{x}_{\omega_i}} + \alpha_{n,m}\, \kappa_{\boldsymbol{x}_n}$ is selected as the new model if

$$\max_{\phi_k \in \text{span}\mathcal{D}_{m\setminus n}} \frac{|\langle \psi_n, \phi_k \rangle_{\mathcal{H}}|}{\|\psi_n\|_{\mathcal{H}} \|\phi_k\|_{\mathcal{H}}} \leq \nu_0$$

Sufficient condition

$$|\text{cor}(\psi_n, \psi_n^{\perp})| \triangleq \frac{|\langle \psi_n, \psi_n^{\perp} \rangle_{\mathcal{H}}|}{\|\psi_n\|_{\mathcal{H}} \|\psi_n^{\perp}\|_{\mathcal{H}}} \leq \nu_0$$



Experimentation (I)

Problem setup

Consider the nonlinear system described by the difference equation

$$x_n = \left(0.8 - 0.5\,\exp(-x_{n-1}^2)\right) x_{n-1} - \left(0.3 + 0.9\,\exp(-x_{n-1}^2)\right) x_{n-2} + 0.1\,\sin(x_{n-1}\pi)$$

Given $d_n = x_n + z_n$, with $z_n \sim \mathcal{N}(0, 0.01)$, the problem is to estimate a nonlinear model of the form $x_n = \psi(x_{n-1}, x_{n-2})$.

The Gaussian kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathrm{e}^{-3.73\,\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}$ is considered, see [Dodd, 2003].

The following algorithms are considered hereafter, with their proper sparsification rule:

- KNLMS [Richard, 2008], with the coherence rule
- NORMA [Kivinen, 2004], with the short-time window
- KRLS [Engel, 2004], with the approximate linear dependence rule
- SSP [Dodd, 2003], with the adjust-then-project rule

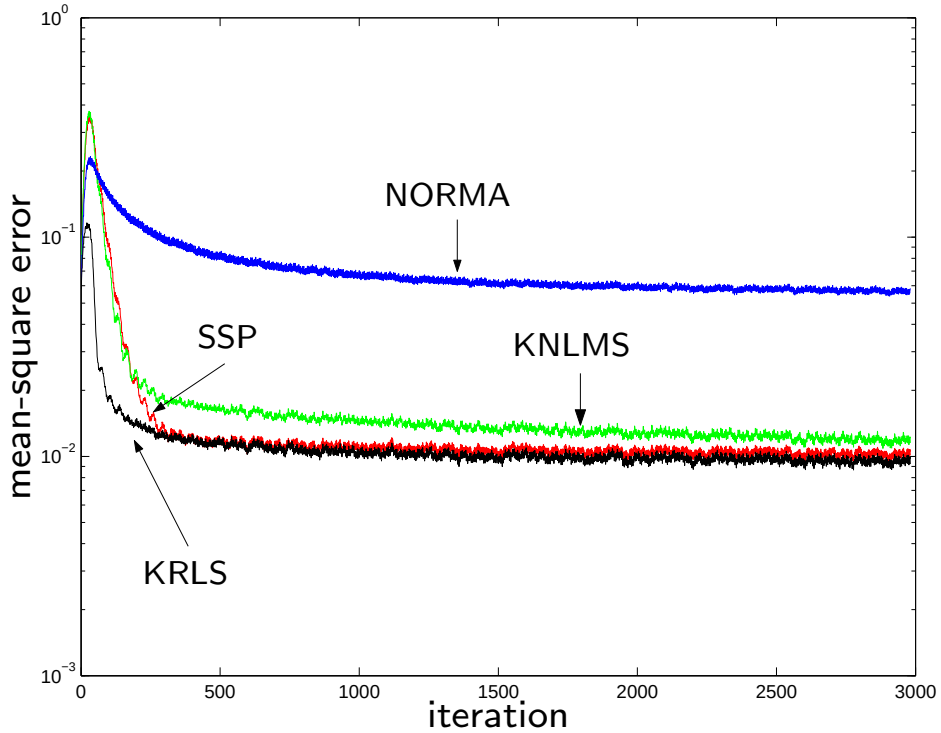
Experimentation (I)

Estimated computational cost per iteration and experimental setup.

Algorithm	$\times$	$+$	Parameter settings	m
NORMA	$2m$	m	$\lambda = 0.98$, $\eta_n = 1/\sqrt{n}$	38
KNLMS	$3m + 1$	$3m$	$\mu_0 = 0.5$, $\epsilon = 0.03$, $\eta = 0.09$	21.3
SSP	$3m^2 + 6m + 1$	$3m^2 + m - 1$	$\kappa = 0.001$, $\eta = 0.1$	23.8
KRLS	$4m^2 + 4m$	$4m^2 + 4m + 1$	$\nu = 0.6$	22.1

Experimentation (I)

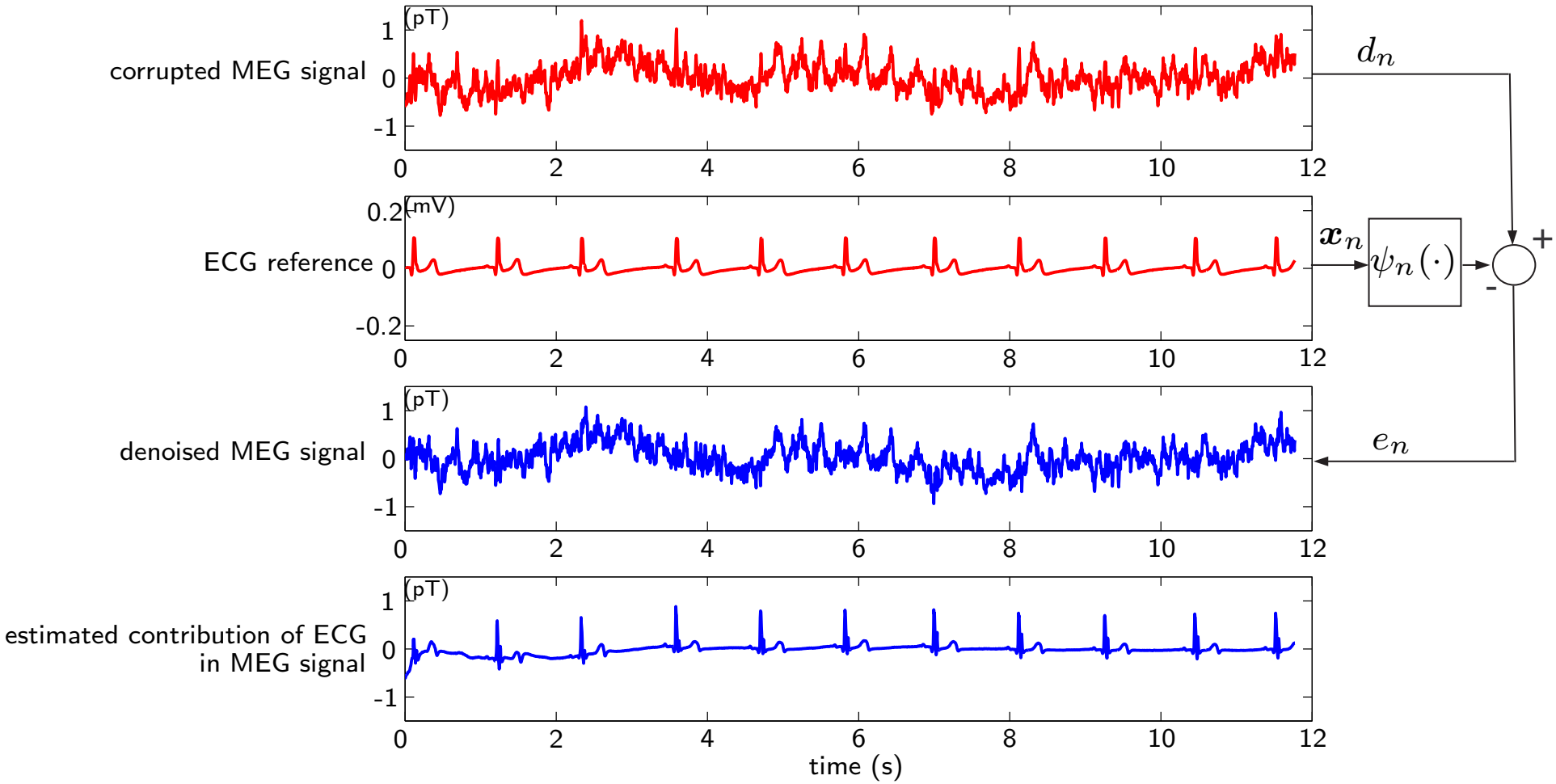
Mean-square error for KNLMS, NORMA, SSP and KRLS obtained by averaging over 200 experiments.



Experimentation (II)

Problem setup

Given the MEG signal and the ECG signal, estimate the contribution of the former into the latter for online denoising.



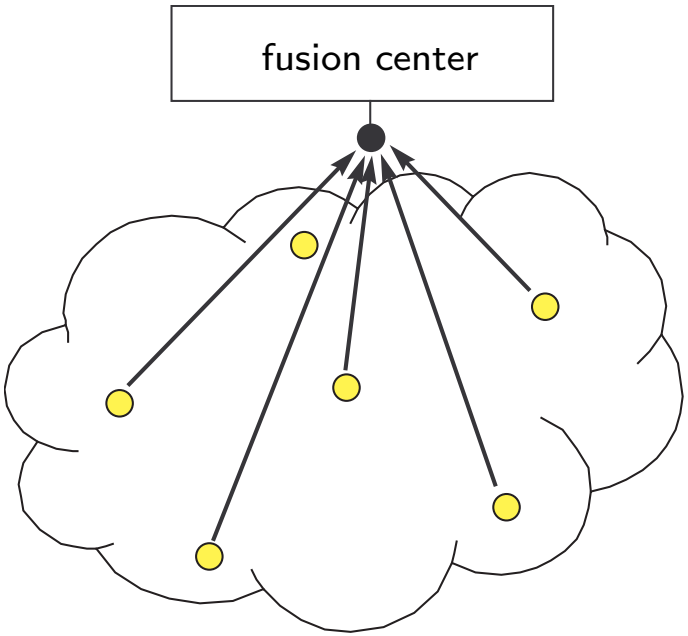
Cooperation strategies

Let $d_{i,j}$ denote the j-th measurement taken at the i-th sensor. We would like to solve a problem of the form

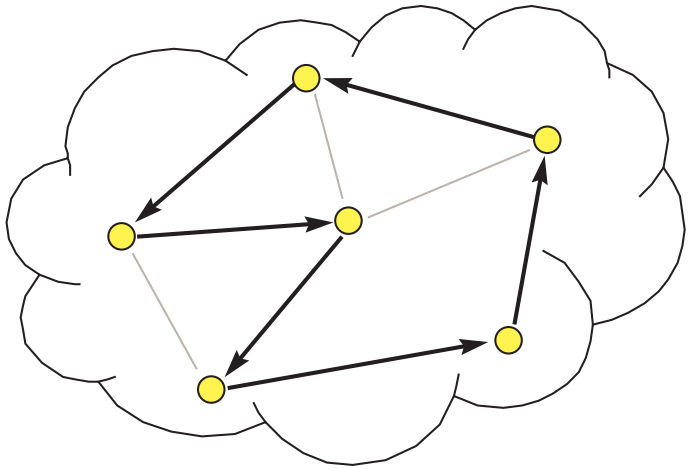
$$\psi^o = \arg\min_{\psi \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} J(\psi(\boldsymbol{x}_i), \{d_{i,j}\}_{j=1}^{M})$$

Two major learning strategies can be considered

- learning with a fusion center [Nguyen, 2005], ...
- distributed learning with in-network processing [Rabbat, 2004], [Predd, 2005], ...



$\mathcal{E} \propto MN^{1+1/d}$

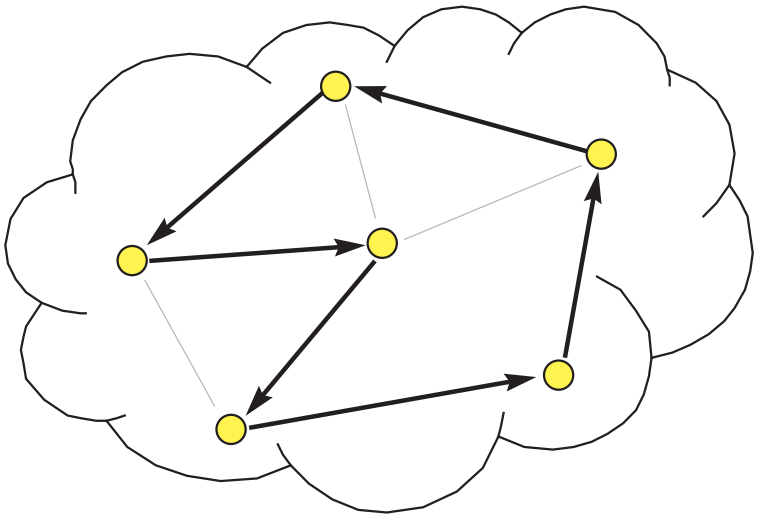


$\mathcal{E} \propto N\epsilon^{-2}$

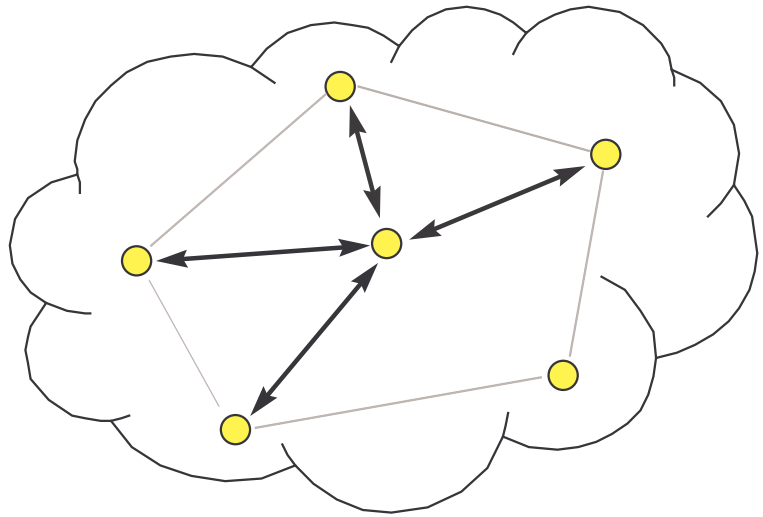
Cooperation strategies

In-network processing suggests two major cooperation strategies

- incremental mode of cooperation [Rabbat, 2004], [Sayed, 2005], ...
- diffusion mode of cooperation [Predd, 2005], [Sayed, 2006], ...



Incremental

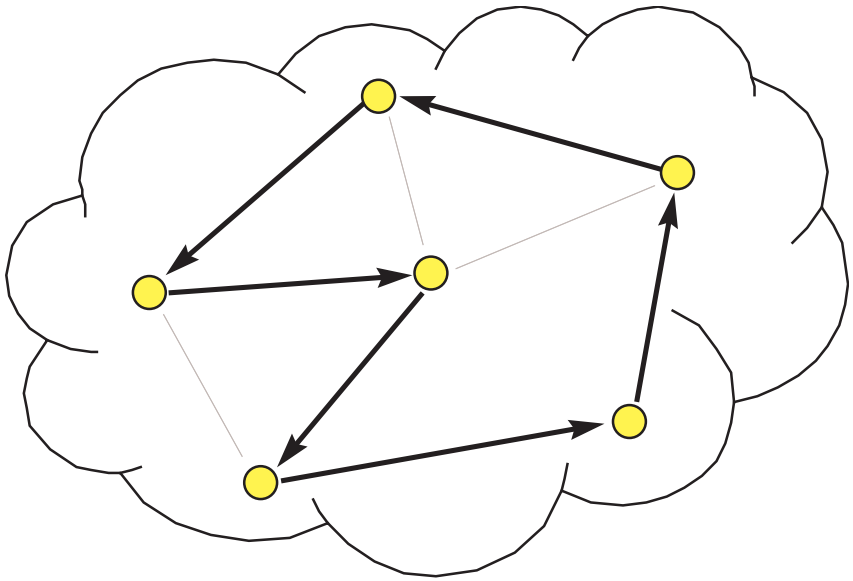


Diffusion

Incremental mode of cooperation

At each instant i, each node n uses local data realizations and the estimate $\psi_{n-1,i}$ received from its adjacent node to perform three tasks:

1. Evaluate a local error quantity
2. Update the estimate $\psi_{n,i}$
3. Pass the updated estimate to its neighbor node $n+1$.



Steepest-descent solution

- Consider a network with N nodes, see Fig. after. Each node n has access to time realizations $\{\boldsymbol{x}_n, d_{n,i}\}$ of spatial data with $n = 1, \ldots, N$.
- The least-squares error criterion $J(\psi) = \sum_{n=1}^{N} E|\langle \psi, \kappa_{\boldsymbol{x}_n} \rangle_{\mathcal{H}} - d_n|^2$ can be decomposed as a sum of N individuals cost functions $J_n(\psi)$, one for each node n.

$$J(\psi) = \sum_{n=1}^{N} J_n(\psi) \quad \text{with} \quad J_n(\psi) = E|\langle \psi, \kappa_{\boldsymbol{x}_n} \rangle_{\mathcal{H}} - d_n|^2$$

- Any kernel-based technique described previously can be used to optimize $J(\psi)$. Let us consider the more simple one, namely, the steepest-descent algorithm.

$$\psi_i = \psi_{i-1} - \frac{\mu}{2} \sum_{n=1}^{N} \nabla J_n(\psi_{i-1})$$

with

$$\nabla J_n(\psi_{i-1}) = 2E\left(\psi_{i-1}(\boldsymbol{x}_n) - d_n\right) \kappa_{\boldsymbol{x}_n}$$

Steepest-descent algorithm (Functional counterpart of [Sayed, 2007])

For each time instant i, repeat

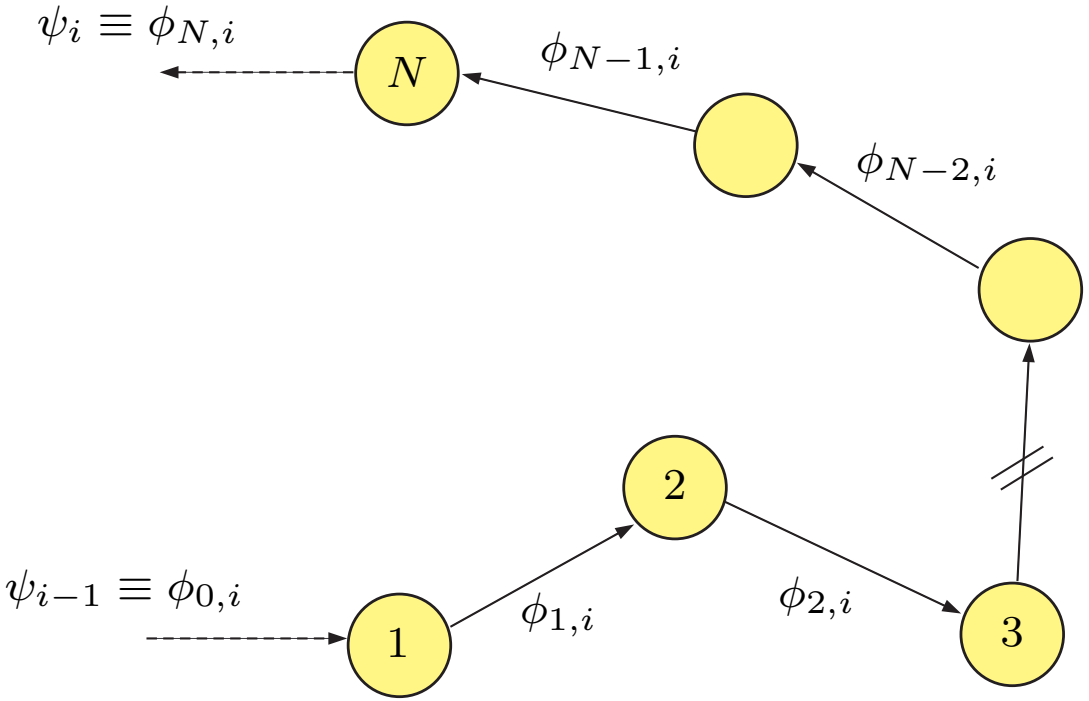
$$\begin{aligned} \phi_{0,i} &= \psi_{i-1} \\ \phi_{n,i} &= \phi_{n-1,i} - \mu \nabla J_n(\psi_{i-1}), \quad n = 1, \ldots, N \\ \psi_i &= \phi_{N,i} \end{aligned}$$

Least-mean-square solution

KLMS algorithm (Functional counterpart of [Sayed, 2007])

For each time instant i, repeat

$$\phi_{0,i} = \psi_{i-1}$$
$$\phi_{n,i} = \phi_{n-1,i} - \mu(\psi_{i-1}(\boldsymbol{x}_n) - d_{n,i})\,\kappa_{\boldsymbol{x}_n}, \quad n = 1, \ldots, N$$
$$\psi_i = \phi_{N,i}$$



Incremental adaptive solution

- KLMS algorithm requires the nodes to have access to global information ψ_{i-1}.
- To overcome this drawback, we evaluate ∇J_n at the local estimate $\phi_{n-1,i}$ received from the node $n-1$.

Incremental algorithm (Functional counterpart of [Sayed, 2007])

For each time instant i, repeat

$$\begin{aligned}
&\phi_{0,i} = \psi_{i-1} \\
&\phi_{n,i} = \phi_{n-1,i} - \mu(\phi_{n-1,i}(\boldsymbol{x}_n) - d_{n,i})\,\kappa_{\boldsymbol{x}_n}, \qquad n = 1, \ldots, N \\
&\psi_i = \phi_{N,i}
\end{aligned}$$

Severe drawback

The order N of the kernel expansion models ψ and ϕ is the number of sensors ...

A sparse modeling approach

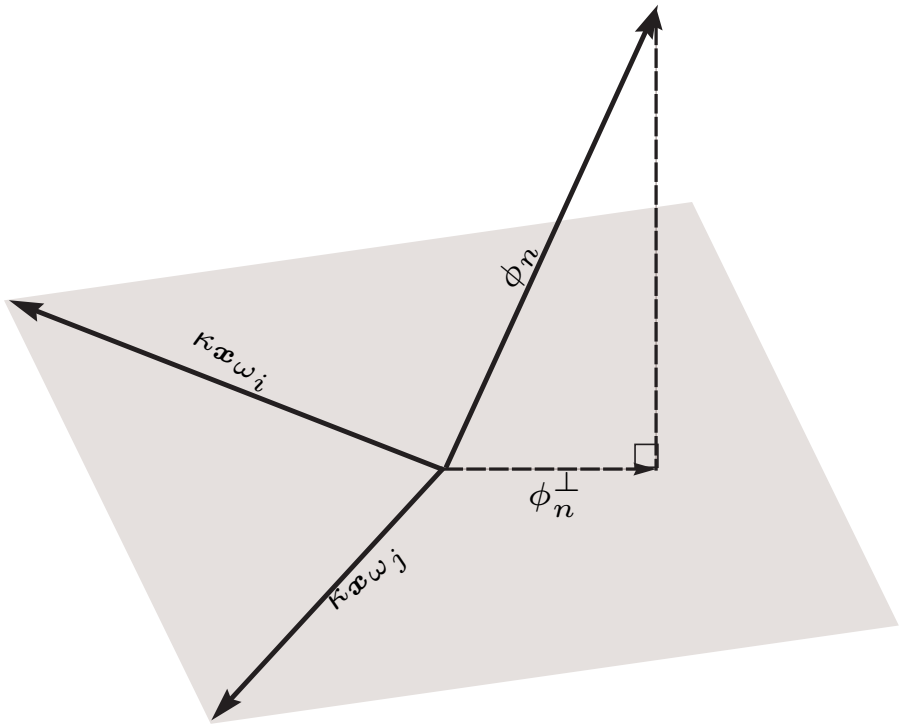
We shall now consider the adapt-then-project criterion described previously.

The function $\phi_n = \sum_{j=1}^{m-1} \alpha_{n,j}\, \kappa_{\boldsymbol{x}_{\omega_j}} + \alpha_{n,m}\, \kappa_{\boldsymbol{x}_n}$ is selected as the new model depending if

$$\max_{\phi_k \in \text{span}\mathcal{D}_{\omega/\{n\}}} \frac{|\langle \phi_n, \phi_k \rangle_{\mathcal{H}}|}{\|\phi_n\|_{\mathcal{H}} \|\phi_k\|_{\mathcal{H}}} \leq \nu_0$$

Sufficient condition

$$|\text{corr}(\phi_n, \phi_n^{\perp})| \triangleq \frac{|\langle \phi_n, \phi_n^{\perp} \rangle_{\mathcal{H}}|}{\|\phi_n\|_{\mathcal{H}} \|\phi_n^{\perp}\|_{\mathcal{H}}} \leq \nu_0$$



A sparse modeling approach

Finally, we conclude this section with the following decentralized algorithm characterized by a model-order control mechanism.

Incremental Adapt-then-project algorithm

For each time instant i, repeat

$$\phi_{0,i} = \psi_{i-1}$$

$$\phi_{n,i} = \phi_{n-1,i} - \mu(\phi_{n-1,i}(\boldsymbol{x}_n) - d_{n,i})\,\kappa_{\boldsymbol{x}_n}, \qquad n = 1, \ldots, N$$

if $|\text{corr}(\phi_{n,i}, \phi^{\perp}_{n,i})| > \nu_0$, replace $\phi_{n,i}$ by $\phi^{\perp}_{n,i}$

$$\psi_i = \phi_{N,i}$$

Experiment I

Consider the following problem of monitoring a diffusion phenomenon governed by the generic partial differential equation

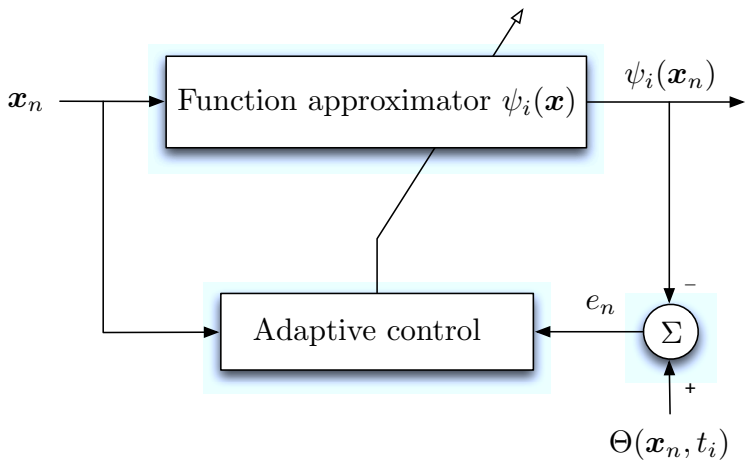
$$\frac{\partial \Theta(\boldsymbol{x}, t)}{\partial t} - \nabla(D\, \nabla \Theta(\boldsymbol{x}, t)) = Q(\boldsymbol{x}, t)$$

with Θ the temperature, D the diffusion coefficient/matrix, and Q a heat source.

Application Two heat sources, the first one from $t = 1$ to $t = 100$, and the second one from $t = 100$ to $t = 200$.

Given $d_{n,i} = \Theta(\boldsymbol{x}_n, t_i) + z_{n,i}$, estimate $\Theta(\boldsymbol{x}_n, t_i)$ via $\psi_i(\boldsymbol{x}_n)$

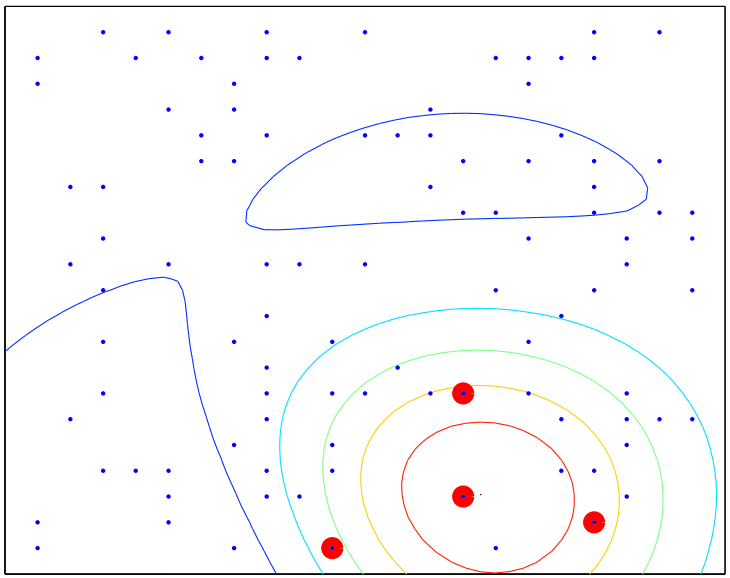
Setup: 100 sensors deployed in a square region with conductivity $D = 0.1$, Gaussian kernel with $\sigma_0 = 0.5$, coherence threshold $\nu_0 = 0.985$, step size $\mu = 0.5$



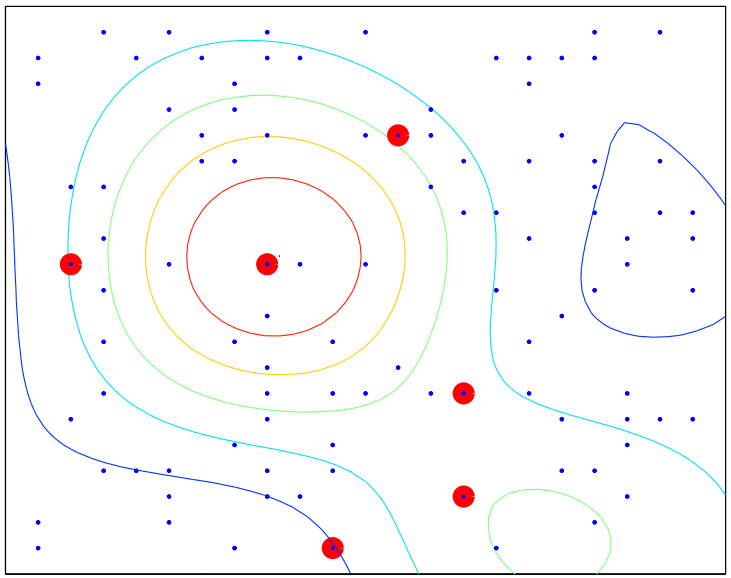
Experiment I

Snapshots of the evolution of the estimated temperature. The selected sensors at these instances are shown with big red dots, whereas the remaining sensors are represented by small blue dots.

$t = 100$



$t = 150$

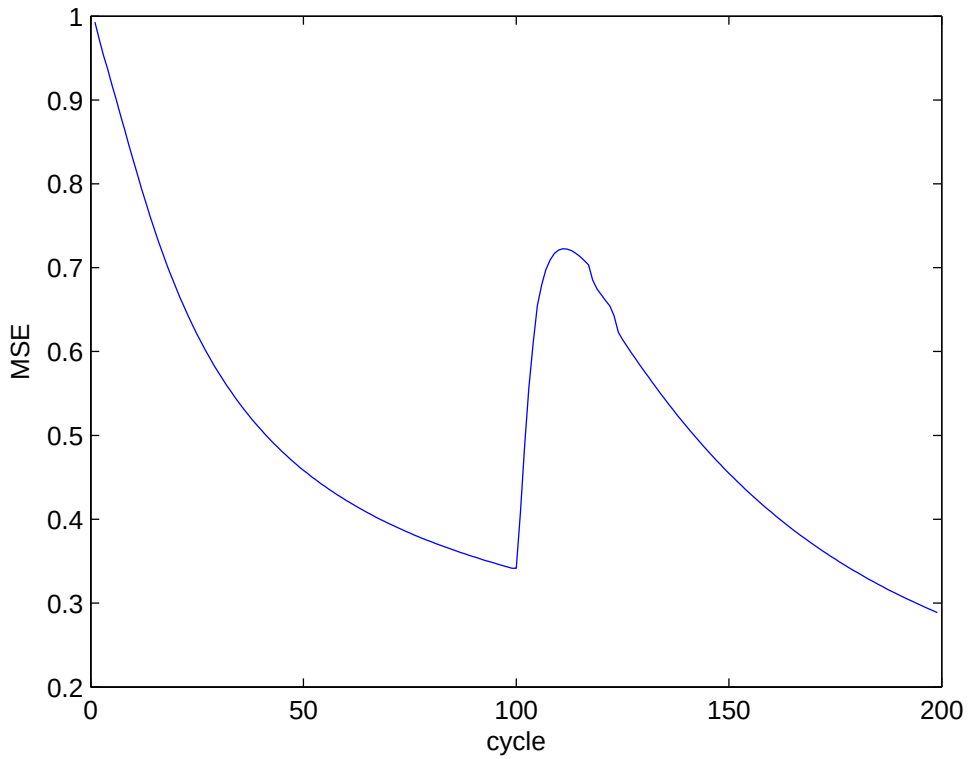


Experiment I

The convergence of the algorithm is illustrated below, where we show the evolution over time of the normalized mean-square prediction error over a grid, defined by

$$\frac{1}{K}\sum_{k=1}^{K}\frac{(\Theta(\boldsymbol{x}_k,t_i)-\psi_{i-1}(\boldsymbol{x}_k))^2}{\Theta(\boldsymbol{x}_k,t_i)^2}.$$

The abrupt change in heat sources at $t=100$ is clearly visible, and highlights the convergence behavior of the proposed algorithm.

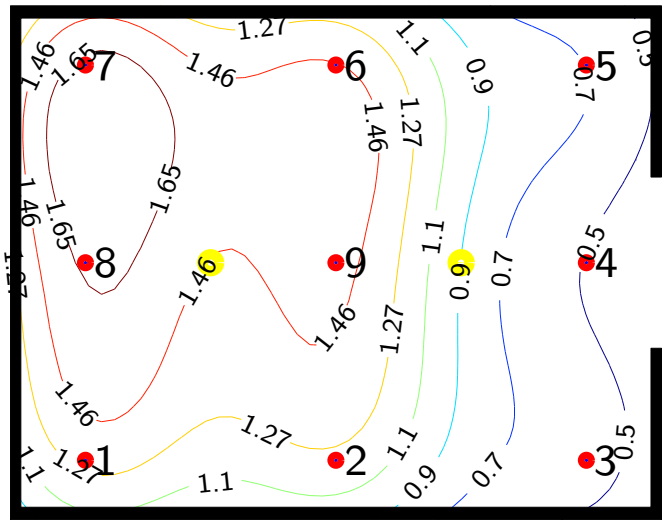


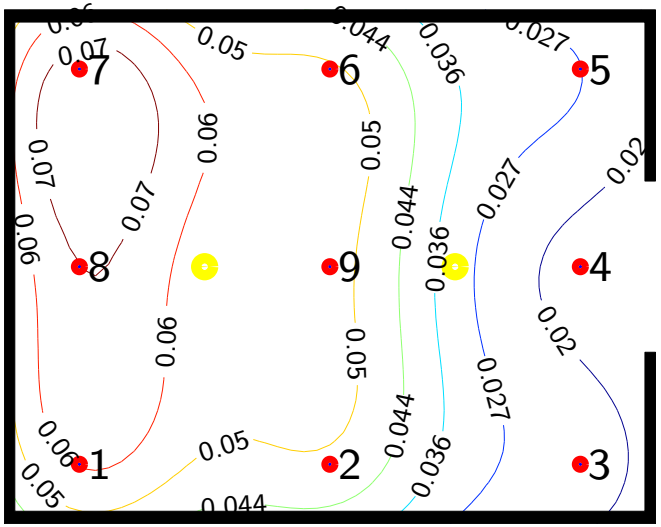
Experiment I

Experiment II

Consider the problem of heat propagation in a partially bounded conducting medium. The heat sources are simultaneously turned on or off over periods of 10 time steps

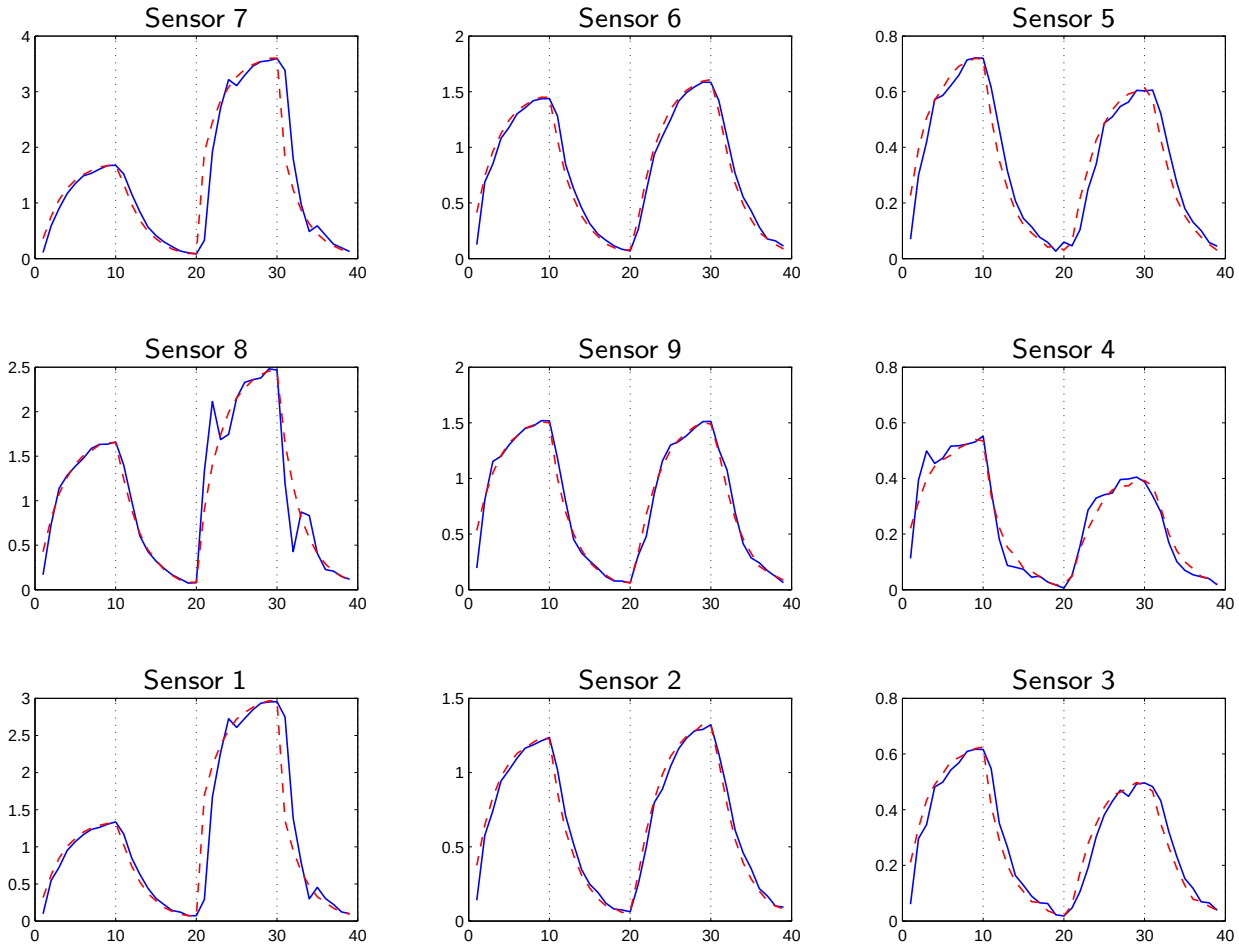
Spatial distribution of temperature estimated at time instants 10 (left) and 20 (right), when the heat sources are turned off and on.





Experiment II

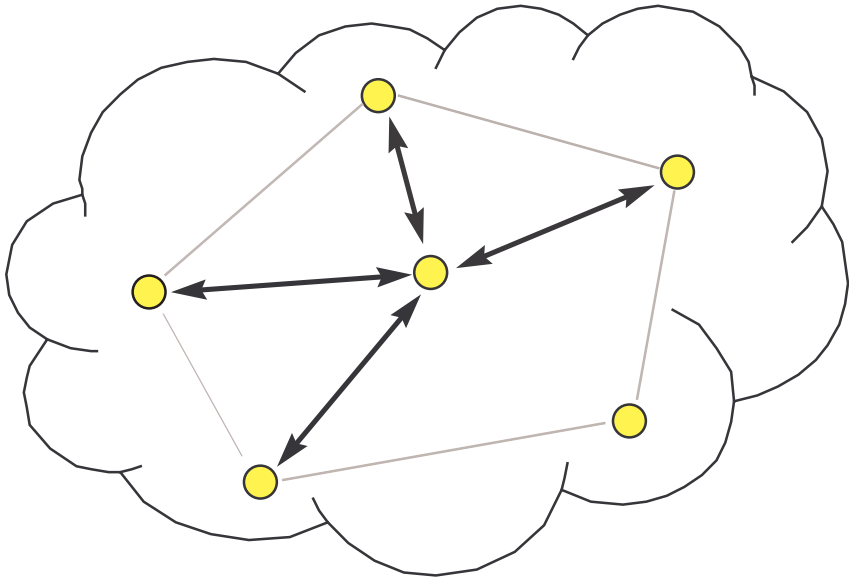
Evolution of the predicted (solid blue) and measured (dashed red) temperatures by each sensor.



Diffusion mode of cooperation

At each time instant i, each node n perform three tasks:

1. Use local data realizations to update its own estimate
2. Consult peer nodes from its neighborhood to get their updated estimates
3. Generate an aggregate estimate



Local versus global optimization

- Seek the optimal ψ^o that minimizes the following global cost function

$$J(\psi) = \sum_{n=1}^{N} E|\langle \psi, \kappa_{\boldsymbol{x}_n} \rangle_{\mathcal{H}} - d_n|^2$$

- The cost function $J(\psi)$ can be rewritten as follows, for any node $k \in \{1, \ldots, N\}$

$$J(\psi) = J_k(\psi) + \sum_{\substack{n=1 \\ n \neq k}}^{N} J_n(\psi)$$

with $J_k(\psi) = \sum_{n \in \mathcal{N}_k} c_{n,k} E|\langle \psi, \kappa_{\boldsymbol{x}_n} \rangle_{\mathcal{H}} - d_n|^2$.

Here, $c_{n,k}$ is the (n,k)-th component of C, which satisfies the following constraints: $c_{n,k} = 0$ if $n \notin \mathcal{N}_k$, $C\mathbb{1} = \mathbb{1}$ and $\mathbb{1}^\top C = \mathbb{1}^\top$.

- Let $\psi_n^o = \arg\min_{\psi \in \mathcal{H}} J_n(\psi)$. Finally, it is equivalent to minimize $J(\psi)$ or the following cost function

$$J_k^\ell(\psi) = \sum_{n \in \mathcal{N}_k} c_{n,k} E|\langle \psi, \kappa_{\boldsymbol{x}_n} \rangle_{\mathcal{H}} - d_n|^2 + \sum_{\substack{n=1 \\ n \neq k}}^{N} \|\psi - \psi_n^o\|_{\mathcal{H}_n}^2$$

Diffusion adaptive solution

- Minimizing $J_k^{\ell}(\psi)$ requires the nodes to have access to global information. To facilitate distributed implementations, consider the relaxed local cost function

$$J_k^r(\psi) = \sum_{n\in\mathcal{N}_k} c_{n,k} E|\langle \psi, \kappa_{\boldsymbol{x}_n}\rangle_{\mathcal{H}} - d_n|^2 + \sum_{n\in\mathcal{N}_k/\{k\}} b_{n,k}\|\psi - \psi_n^o\|_{\mathcal{H}}^2$$

  where ψ_n^o is now the best estimate available at node n.

- The iterative steepest-descent approach for minimizing $J_k^r(\psi)$ can be expressed in the following form

$$\psi_{k,i} = \psi_{k,i-1} - \frac{\mu}{2}\nabla J_k^r(\psi_{k,i-1})$$

  with

$$\nabla J_k^r(\psi_{k,i-1}) = \sum_{n\in\mathcal{N}_k} 2\, c_{n,k} E\left(\psi_{k,i-1}(\boldsymbol{x}_n) - d_n\right)\kappa_{\boldsymbol{x}_n} + \sum_{n\in\mathcal{N}_k/\{k\}} 2\, b_{n,k}(\psi_{k,i-1} - \psi_n^o)$$

- Incremental solutions are useful for minimizing sums of convex functions. They are based on the principle of iterating sequentially over each sub-gradient

Diffusion adaptive solution

Incremental algorithms are useful for minimizing sums of convex functions. They consist of iterating sequentially over each sub-gradient, in some predefined order.

Adapt-then-Combine kernel LMS

For each time instant i and each node k, repeat

$$\phi_{k,i} = \psi_{k,i-1} - \mu_k \sum_{n\in\mathcal{N}_k} c_{n,k} \left(\psi_{k,i-1}(\boldsymbol{x}_n) - d_{n,i}\right) \kappa_{\boldsymbol{x}_n}$$

$$\psi_{k,i} = \sum_{n\in\mathcal{N}_k} b_{n,k}\, \phi_{k,i}$$

Combine-then-Adapt kernel LMS

For each time instant i and each node k, repeat

$$\phi_{k,i-1} = \sum_{n\in\mathcal{N}_k} b_{n,k}\, \psi_{k,i-1}$$

$$\psi_{k,i} = \phi_{k,i-1} - \mu_k \sum_{n\in\mathcal{N}_k} c_{n,k} \left(\phi_{k,i-1}(\boldsymbol{x}_n) - d_{n,i}\right) \kappa_{\boldsymbol{x}_n}$$

Remark:
Measurements and regressors are not exchanged between the nodes if $c_{n,k} = \delta_{n,k}$

Experiment I

Consider again the following problem of monitoring a diffusion phenomenon governed by the generic partial differential equation

$$\frac{\partial \Theta(\boldsymbol{x},t)}{\partial t} - \nabla(D\,\nabla\Theta(\boldsymbol{x},t)) = Q(\boldsymbol{x},t)$$

with Θ the temperature, D the diffusion coefficient/matrix, and Q a heat source.

Application Two heat sources, the first one from $t = 1$ to $t = 100$, and the second one from $t = 100$ to $t = 200$.

Given $d_{n,i} = \Theta(\boldsymbol{x}_n, t_i) + z_{n,i}$, estimate $\Theta(\boldsymbol{x}_n, t_i)$ via $\psi_i(\boldsymbol{x}_n)$

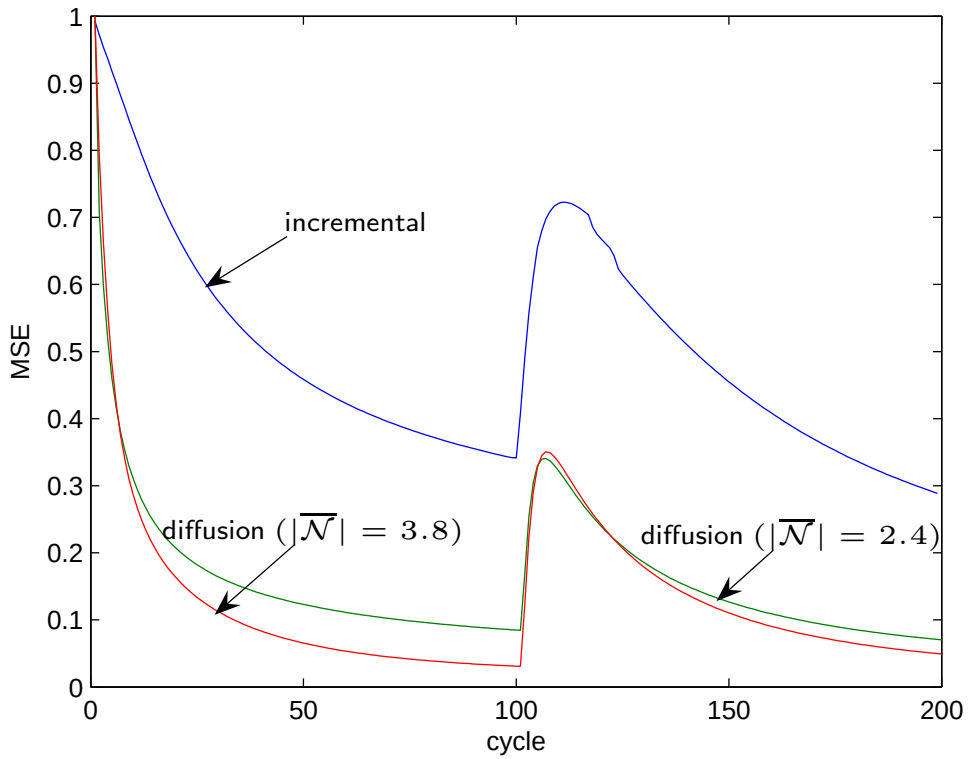
Setup: One hundred sensors deployed in a square region with conductivity $D = 0.1$, Gaussian kernel with $\sigma_0 = 0.5$, step-size $\mu = 0.5$

Experiment I

The convergence of the algorithm is illustrated below, where we show the evolution over time of the normalized mean-square prediction error over a grid, defined by

$$\frac{1}{K}\sum_{k=1}^{K}\frac{(\Theta(\boldsymbol{x}_k,t_i)-\psi_{i-1}(\boldsymbol{x}_k))^2}{\Theta(\boldsymbol{x}_k,t_i)^2}.$$

The better performance of the diffusion-based algorithm is clearly visible... at the expense of a higher communication cost.



Conclusion and future work

In this talk, the following topics have been addressed within the context of distributed learning in wireless sensor networks

- two cooperation modes, incremental and diffusion
- non-linear algorithms based on the RKHS framework
- sparsification strategies

Applications to diffusion process monitoring with dynamic sources were considered, and simulation results showed the relevance of the proposed approaches.

Extensions to this work include

- statistical analysis
- specific kernel design
- application to other problems
- ...