

# Distributed Quasi-Newton Method and its Application to the Optimal Reactive Power Flow Problem

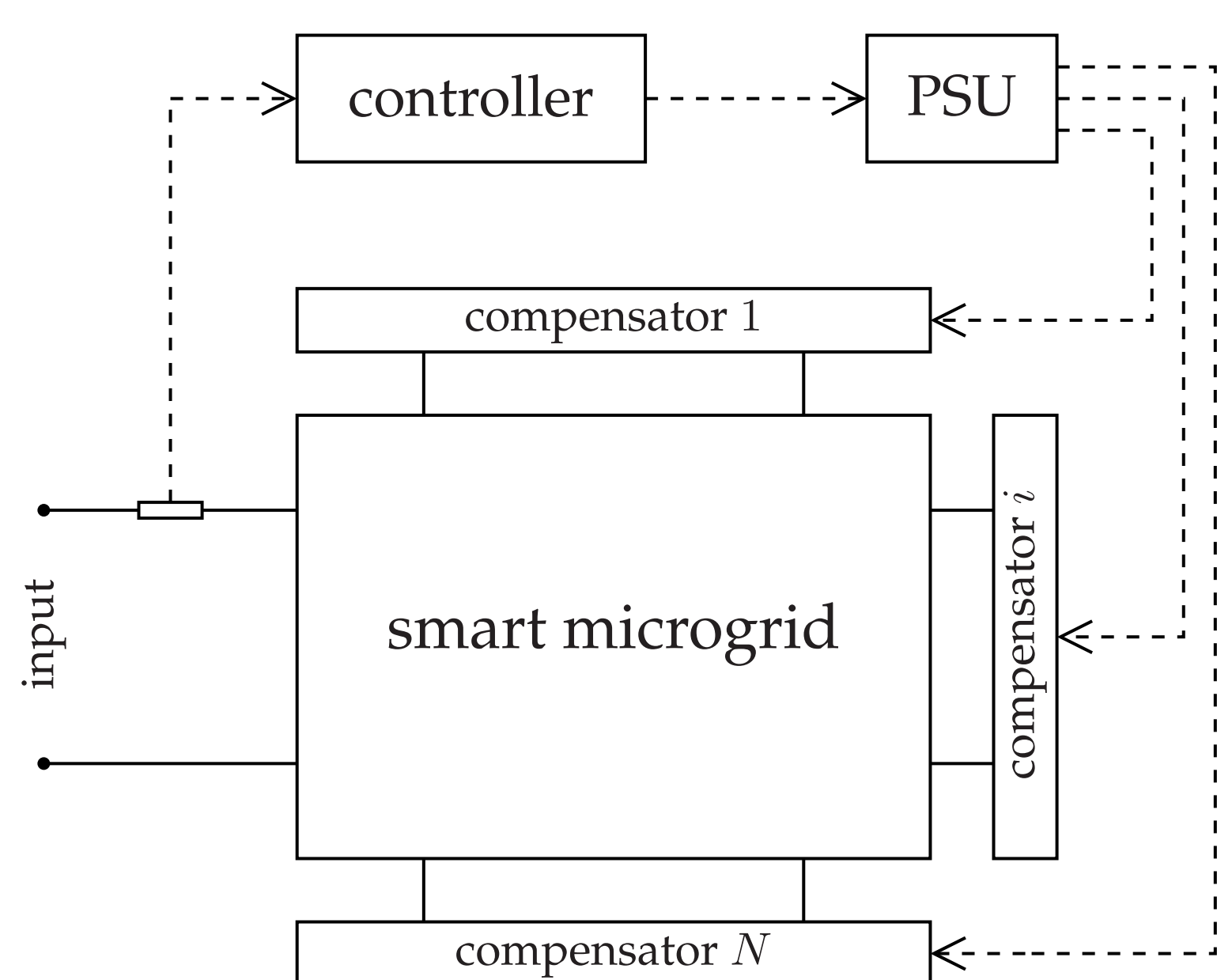
Saverio Bolognani, Sandro Zampieri

## Motivating application

**Smart Microgrid** – a portion of the electrical power distribution network that connects to the larger distribution grid (or to the transmission grid) in one point and that is managed autonomously from the rest of the network. In particular, ancillary services like

- reactive power compensation
- voltage support
- voltage profile quality enhancement

are taken care by some microgrid controllers, whose objective is to provide a high quality of the service to the microgrid users while satisfying some constraint on how the microgrid interfaces with the larger grid.

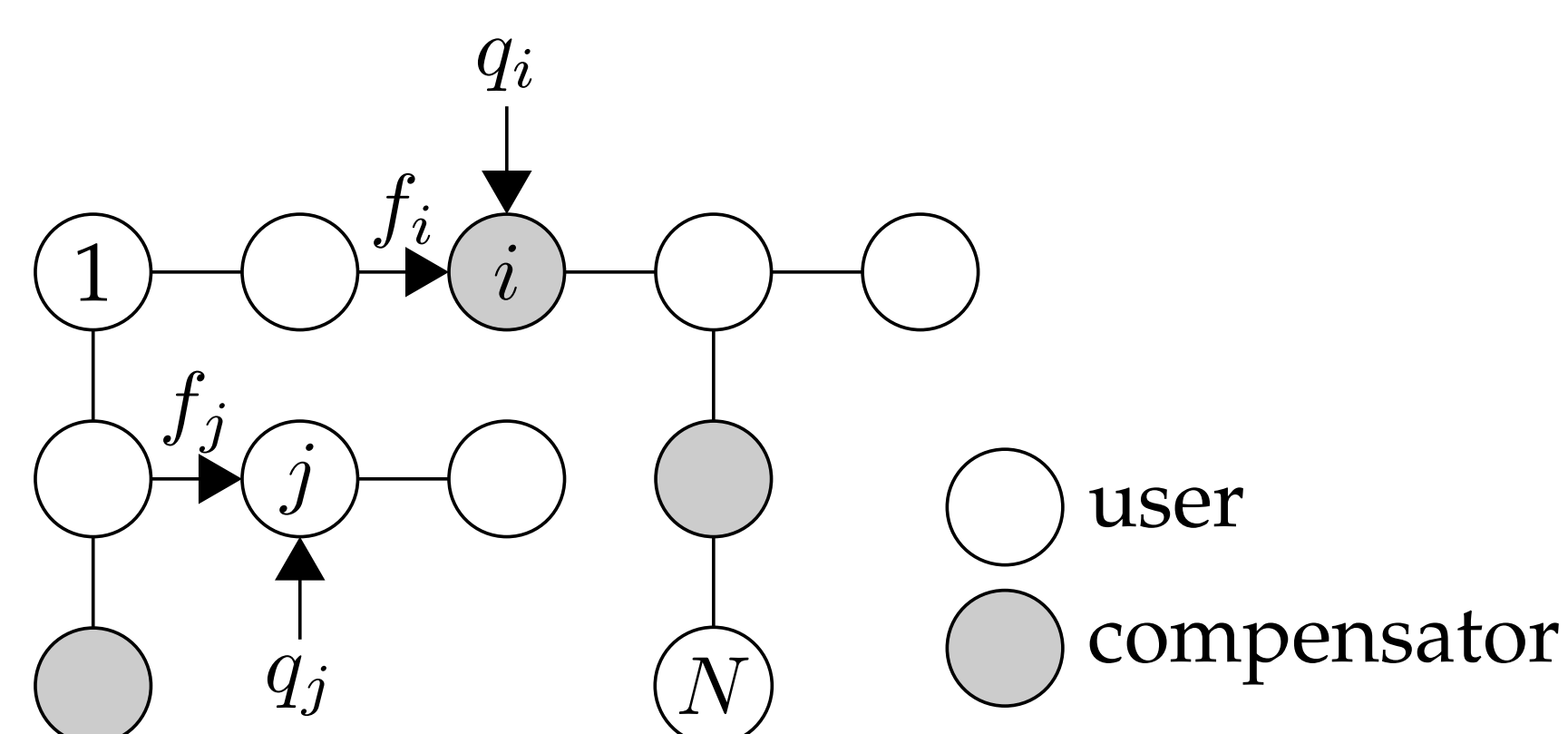


## Reactive power compensation

Users may require a sinusoidal current which is not in phase with voltage. They therefore demand active power (associated with in-phase current) and **reactive power** (associated with out-of-phase current). Reactive power flows contribute to

- power losses on the lines
- voltage drop in the network
- grid instability.

It is preferable to **minimize reactive power flows** by producing it close to where is needed. Many agents can produce reactive power – basically the electronic interfaces of every micro generator (wind, solar, etc.).



$$\min_{\mathbf{1}^T \mathbf{q} = c} F = \sum_{\text{edges}} k_i f_i^2 = \mathbf{q}^T \frac{\mathbf{M}}{2} \mathbf{q} + \mathbf{m}^T \mathbf{q}$$

- $c$  is the total demanded reactive power
- $k_i$  is the resistance of edge  $i$
- $\mathbf{M}$  depends on the grid topology only
- $\mathbf{m}$  depends on topology and demand.

The **gradient**  $\mathbf{g} = \mathbf{M}\mathbf{q} + \mathbf{m}$  can be estimated element-wise from **steady state voltage measure**  $\mathbf{v}$ , up to an unknown constant  $\xi$

$$\mathbf{g}(\mathbf{q}) = \mathbf{v}(\mathbf{q}) + \xi \mathbf{1}.$$

## References

- [1] S. Bolognani, S. Zampieri. Distributed Quasi-Newton Method and its Application to the Optimal Reactive Power Flow Problem in Proceedings of NECSYS 2010 (Annecy, France)

## Gradient-driven algorithms

**Iterative algorithm**

$$\mathbf{q}^+ = \mathbf{q} - \Gamma \mathbf{g}(\mathbf{q})$$

$$\Gamma^+ = \Phi(\mathbf{q}, \Gamma, \mathbf{g}(\mathbf{q})),$$

where  $\Gamma$  is a gain matrix that satisfies  $\mathbf{1}^T \Gamma = \mathbf{0}$ .

**Constrained Newton Algorithm**

$$\Gamma = \mathbf{M}^{-1} - \frac{\mathbf{M}^{-1} \mathbf{1} \mathbf{1}^T \mathbf{M}^{-1}}{\mathbf{1}^T \mathbf{M}^{-1} \mathbf{1}}$$

achieves fastest (one-step) convergence.

Fast convergence is critical, as **measuring the gradient corresponds to actuating the system and waiting for its steady state response**.

## Quasi-Newton Method

When  $\mathbf{M}$  is unknown, it can be estimated from previous steps of the algorithm. Consider the following specialization of **Broyden's method** to the constrained case:

$$\mathbf{q}^+ = \mathbf{q} - \mathbf{G} \mathbf{d}$$

$$\mathbf{G}^+ = \mathbf{G} + \frac{[\Delta \mathbf{q} - \mathbf{G} \Delta \mathbf{d}] \Delta \mathbf{d}^T}{\Delta \mathbf{d}^T \Delta \mathbf{d}},$$

where  $\mathbf{d} = (\mathbf{I} - \mathbf{1} \mathbf{1}^T / N) \mathbf{g}$  is the projection of the gradient on the constraint, and  $\Delta \mathbf{d} = \mathbf{d}^+ - \mathbf{d}$ ,  $\Delta \mathbf{q} = \mathbf{q}^+ - \mathbf{q}$ .

**Theorem** The constrained Broyden method, initialized with  $\mathbf{G}(0) = \alpha \mathbf{I}$  and  $\mathbf{q}(0)$  any feasible state, converges to the solution of the quadratic problem in at most  $2N$  steps.

## Distributed Quasi Newton Methods

The main contribution of this work is the design of a **distributed version of Broyden's method** in which every agent only needs

- information that can be gathered from the **set of neighbors**
- some shared piece of information (**fusion vector**) on which all nodes agree.

$$\mathbf{q}^+ = \mathbf{q} - \mathbf{H} \mathbf{g}(\mathbf{q}) + \frac{\mathbf{1}^T \mathbf{H} \mathbf{g}(\mathbf{q})}{\mathbf{1}^T \mathbf{H} \mathbf{1}} \mathbf{H} \mathbf{1}$$

$$\mathbf{H}^+ = \mathbf{H} + \mathcal{P}_{\mathcal{E}} \left[ \mathbf{D}^+ (\Delta \mathbf{q} - \mathbf{H} \Delta \mathbf{g}(\mathbf{q})) \Delta \mathbf{g}^T(\mathbf{q}) \right],$$

where  $\mathcal{P}_{\mathcal{E}}$  is the projection operator on the sparsity constraint induced by the set of edges  $\mathcal{E}$

$$(\mathcal{P}_{\mathcal{E}}(\mathbf{A}))_{ij} = \begin{cases} \mathbf{A}_{ij}, & \text{if } (i, j) \in \mathcal{E} \\ 0, & \text{if } (i, j) \notin \mathcal{E}, \end{cases}$$

and where  $\mathbf{D}^+$  is the diagonal matrix defined as

$$(\mathbf{D}^+)_{ii} = \begin{cases} 1/\mathbf{g}^{(i)T} \mathbf{g}^{(i)} & \text{if } \mathbf{g}^{(i)} \neq \mathbf{0} \\ 0, & \text{if } \mathbf{g}^{(i)} = \mathbf{0}, \end{cases}$$

$\mathbf{g}^{(i)}$  being the part of the gradient available to  $i$ .

This algorithm can be distributed among nodes obtaining the following update laws.

$$q_i^+ = q_i - \mathbf{H}_i^T \mathbf{g}^{(i)} + x \mathbf{H}_i^T \mathbf{1}^{(i)}$$

$$\mathbf{H}_i^+ = \mathbf{H}_i + \left[ \Delta q_i - \mathbf{H}_i^T \Delta \mathbf{g}^{(i)} \right] \frac{\Delta \mathbf{g}^{(i)}}{\Delta \mathbf{g}^{(i)T} \Delta \mathbf{g}^{(i)}},$$

where  $x = \bar{z}_1 / \bar{z}_2$ , and  $\bar{z}_{1,2}$  are the elements of the 2-dimensional vector resulting from the average consensus algorithm initialized as

$$\mathbf{z}^{(i)}(0) = \begin{bmatrix} \mathbf{H}_i^T \mathbf{g}^{(i)} \\ \mathbf{H}_i^T \mathbf{1}^{(i)} \end{bmatrix}.$$

## Distributing the algorithm

The generic agent  $i$  has to perform the update

$$q_i^+ = q_i - \Gamma_i^T \mathbf{g}(\mathbf{q})$$

$$\Gamma_i^+ = \Phi_i(\mathbf{q}, \Gamma, \mathbf{g}(\mathbf{q})).$$

In general, it is not possible to implement it in a distributed manner. Let us introduce a **fusion vector**  $\mathbf{x}$  on which all nodes agree

$$\mathbf{x} = f(\mathbf{q}, \mathbf{g}(\mathbf{q}), \boldsymbol{\eta}_i, i \in \mathcal{V}),$$

and consider the local update laws

$$q_i^+ = q_i - \gamma_i(q_j, g_j(\mathbf{q}), j \in \mathcal{N}_i; \boldsymbol{\eta}_i, \mathbf{x})$$

$$\boldsymbol{\eta}_i^+ = \phi_i(q_j, g_j(\mathbf{q}), j \in \mathcal{N}_i; \boldsymbol{\eta}_i, \mathbf{x}),$$

where  $\boldsymbol{\eta}_i$  is a local parameter vector and  $\mathcal{N}_i$  is the set of neighbors of node  $i$ .

For example, let  $\mathbf{H}$  be an approximation of  $\mathbf{M}^{-1}$  satisfying the **sparsity constraint**

$$\mathbf{H}_{ij} \neq 0 \Rightarrow \text{nodes } i \text{ and } j \text{ can communicate}$$

and consider the update law

$$\mathbf{q}^+ = \mathbf{q} - \mathbf{H} \mathbf{g}(\mathbf{q}) + \frac{\mathbf{1}^T \mathbf{H} \mathbf{g}(\mathbf{q})}{\mathbf{1}^T \mathbf{H} \mathbf{1}} \mathbf{H} \mathbf{1}.$$

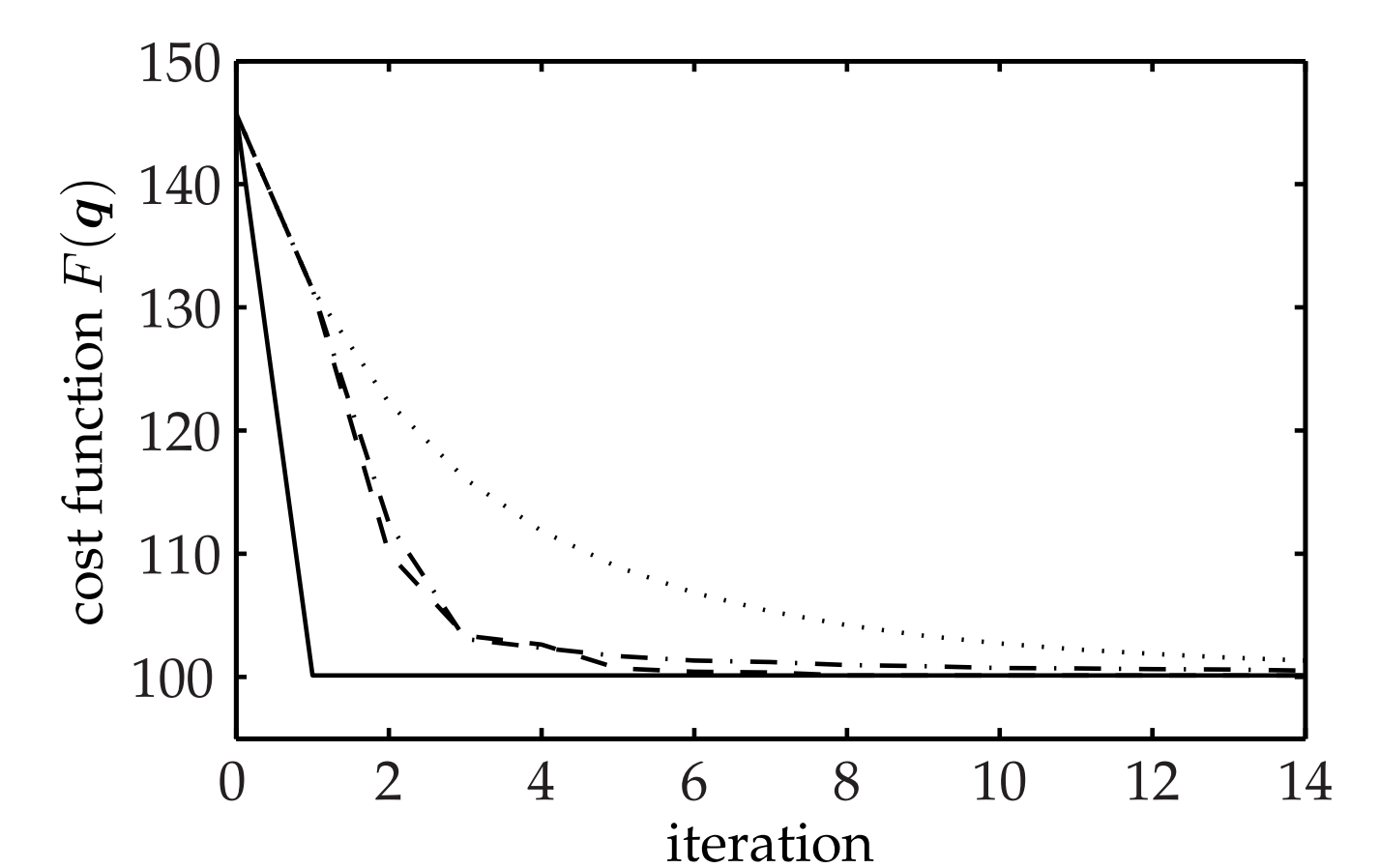
Because of the sparsity of  $\mathbf{H}$ , both  $\mathbf{H} \mathbf{g}$  and  $\mathbf{H} \mathbf{1}$  can be computed distributively. Moreover, they can agree on the  $x = \frac{\mathbf{1}^T \mathbf{H} \mathbf{g}(\mathbf{q})}{\mathbf{1}^T \mathbf{H} \mathbf{1}}$  by running an **average consensus** algorithm between two consecutive iterations of the algorithm and by computing  $x$  from the consensus vector  $\bar{\mathbf{z}}$

$$\mathbf{z}^{(i)}(0) = \begin{bmatrix} \mathbf{H}_i \mathbf{g}(\mathbf{q}) \\ \mathbf{H}_i \mathbf{1} \end{bmatrix} \longrightarrow \bar{\mathbf{z}} = \begin{bmatrix} \frac{1}{N} \mathbf{1}^T \mathbf{H} \mathbf{g}(\mathbf{q}) \\ \frac{1}{N} \mathbf{1}^T \mathbf{H} \mathbf{1} \end{bmatrix}.$$

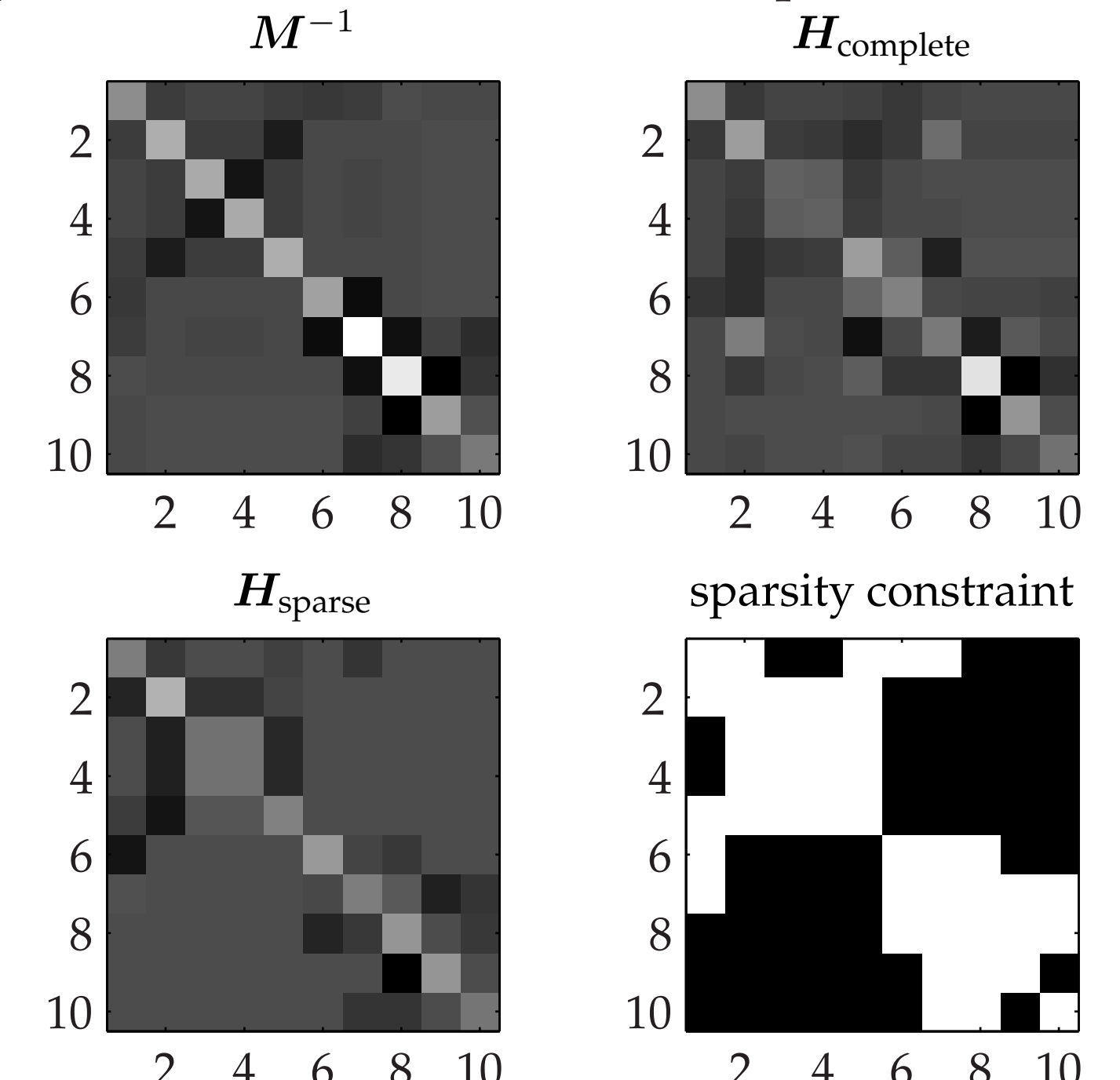
Note that the **memory requirements scale linearly with the number of neighbors**.

The effectiveness of the proposed algorithm depends on **how well  $\mathbf{M}^{-1}$  can be approximated as a sparse matrix consistent with the graph**.

**Numerical simulations** show how the proposed algorithm behaves when applied to the problem of **optimal reactive power compensation**.



Newton method (solid), quasi-Newton method with complete communication (dashed), quasi-Newton method with sparsity constraint (dash-dotted), steepest descent (dotted).



Estimates of the inverse of the Hessian, together with the real one and with the sparsity constraint.