

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

Saverio Bolognani\* Sandro Zampieri\*\*

\* *Department of Information Engineering, University of Padova, Padova, Italy (e-mail: saverio.bolognani@dei.unipd.it).*

\*\* *Department of Information Engineering, University of Padova, Padova, Italy (e-mail: zampi@dei.unipd.it).*

---

**Abstract:** We consider a distributed system of  $N$  agents, on which we define a quadratic optimization problem subject to a linear equality constraint. We assume that the nodes can estimate the gradient of the cost function by measuring the steady state response of the system. Even if the cost function cannot be decoupled into individual terms for the agents, and the linear constraint involves the whole system state, we are able to design a distributed, gradient-driven, algorithm, for the solution of the optimization problem. This algorithm belongs to the class of quasi-Newton methods and requires minimal knowledge of the system to behave fairly well. We proved finite time convergence of the algorithm in its centralized version, and we designed its distributed implementation in the case in which a communication graph is given. In this latter case, the tool of average consensus results to be fundamental for the distribution of the algorithm. As a testbed for the proposed method, we consider the problem of optimal distributed reactive power compensation in smart microgrids.

*Keywords:* Distributed optimization; quasi-Newton method; consensus; reactive power compensation; smart microgrids.

---

## 1. INTRODUCTION

Distributed optimization has been a key research subject since when, in computer science, it has been explored the possibility of minimizing a cost function by distributing the processing task among agents (processors). This framework has been first presented in the seminal work by Tsitsiklis et al. (1986) and gave birth to a well known literature on the field, e.g. Bertsekas and Tsitsiklis (1997).

More recently, the problem of distributed optimization have been applied to the more challenging scenario of complex, large-scale systems. In these systems different issues may coexist: agents are in a large, unknown and time-varying, number; individual agents do not know the whole cost function and cannot evaluate it; communication is constrained; sensing and actuation on an underlying physical system has to be performed together with data processing; the physical system is partially unknown. A notable “success story” in this sense is the application of distributed optimization to the Internet: since the work of Kelly et al. (1998), large-scale data networks have been probably the preferred testbed for these algorithms.

Among the main tools for distributed optimization in complex systems are the subgradient methods (see Nedic and Ozdaglar (2009) and references therein). The largest part of these works assume however that the cost function to be minimized is the sum of individual cost functions of the agents (which is true, for example, in the problem of utility maximization in data networks).

Here we consider the simple case of a quadratic convex function, but we do not assume separability of the cost function in individual terms; moreover, a linear equality constraint couples the decision variables of all nodes.

To tackle this problem, we specialized some quite classical tools in convex optimization, quasi-Newton methods (Dennis and Schnabel (1983), Nocedal and Wright (2006)), to the case of a linearly constrained problem. Then we exploit the tool of average consensus (Olfati-Saber et al. (2007), Fagnani and Zampieri (2008)) to apply these methods to a large scale complex system. As a result, we prove global finite time convergence of a quasi-Newton method for constrained minimization, and we show, via some numerical simulations, how the algorithm behaves well when it is distributed among nodes that can only communicate with a subset of neighbors.

As a motivation for this work, we present the problem of optimal reactive power compensation in power distribution networks. This application is part of the extremely important framework of ancillary services in the so called smart-grids (Santacana et al. (2010), Ipakchi and Albuyeh (2009)), which can be considered among the most interesting and intriguing testbeds at the moment.

## 2. PROBLEM FORMULATION

Consider a distributed system described by a state  $\mathbf{q} \in \mathbb{R}^N$ . Each component  $q_i$  of  $\mathbf{q}$  is the (scalar) state of a single agent  $i \in \mathcal{V} = \{1, \dots, N\}$ . Every agent updates

its state synchronously at times  $t_k = kT, k \in \mathbb{Z}$ , where  $T$  is a constant positive integer and  $t_0 = 0$ .

Suppose that an underlying physical system exists, described by the  $N$ -dimensional non linear ODE

$$\dot{\mathbf{v}}(t) = h(\mathbf{v}(t), \mathbf{q}(t)). \quad (1)$$

Assume that its steady state responde to constant input  $\mathbf{q}(t) = \mathbf{q}$  is  $\mathbf{v}(t) = \mathbf{v} = \mathbf{g}(\mathbf{q}) \in \mathbb{R}^N$ . Following Chapter 8.1 in Isidori (1995), this is guaranteed if:

- $h(\mathbf{g}(\mathbf{q}), \mathbf{q}) = 0$ ;
- $\exists \bar{\mathbf{q}}$  such that  $\mathbf{v} = 0$  is exponentially stable for (1) with constant input  $\mathbf{q}(t) = \bar{\mathbf{q}}$  (and  $\mathbf{g}(\bar{\mathbf{q}}) = 0$ ).

Assume that each node is capable of measuring the  $i$ -th element  $v_i$  of  $\mathbf{v}$ .

Let us consider a quadratic cost function of  $\mathbf{q}$

$$F(\mathbf{q}) = \mathbf{q}^T \frac{\mathbf{M}}{2} \mathbf{q} + \mathbf{m}^T \mathbf{q}, \quad \mathbf{M} > 0 \quad (2)$$

whose gradient  $\nabla F(\mathbf{q}) = \mathbf{M}\mathbf{q} + \mathbf{m}$  coincides with the function  $\mathbf{g}(\mathbf{q})$  given above. In other words, by measuring the steady-state response of the system (1) to the constant input  $\mathbf{q}$ , the nodes can estimate (element-wise) the gradient of the cost function (2) in  $\mathbf{q}$ . Notice that as  $\mathbf{g}(\bar{\mathbf{q}}) = 0$ , then  $\bar{\mathbf{q}} = \arg \min F(\mathbf{q})$ .

In this work we focus on the problem of designing a distributed algorithm for the minimization of (2) subject to a linear equality constraint, that is solving

$$\min_{\mathbf{q}} F(\mathbf{q}) \quad \text{subject to} \quad \mathbf{a}^T \mathbf{q} = b. \quad (3)$$

### 2.1 Communication constraints

Communication between agents is constrained to be consistent with a given communication graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ , where  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$  is a set of edges  $(i, j)$ , with  $i, j \in \mathcal{V}$ .

Therefore, for every node  $i$ , there exists a set of neighbors  $\mathcal{N}_i = \{j \in \mathcal{V} | (j, i) \in \mathcal{E}\} \subseteq \mathcal{V}$  which is the set of agents from which node  $i$  can gather data. We assume that  $(i, i) \in \mathcal{E} \forall i$ , and that  $\mathcal{G}$  is strongly connected.

## 3. OPTIMAL REACTIVE POWER COMPENSATION IN MICROGRIDS

As a motivating example, we introduce in this section the problem of optimal reactive power compensation in power distribution networks.

Let us define a *smart microgrid* as 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, and 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. We focus here on reactive power compensation.

Consider the steady state of the network, when voltages and currents are sinusoidal signals at frequency  $\omega_0/2\pi$  in any point (at any port) of the network:

$$u_i(t) = U_i \sin(\omega_0 t + \theta_i^u), \quad i_i(t) = I_i \sin(\omega_0 t + \theta_i^i).$$

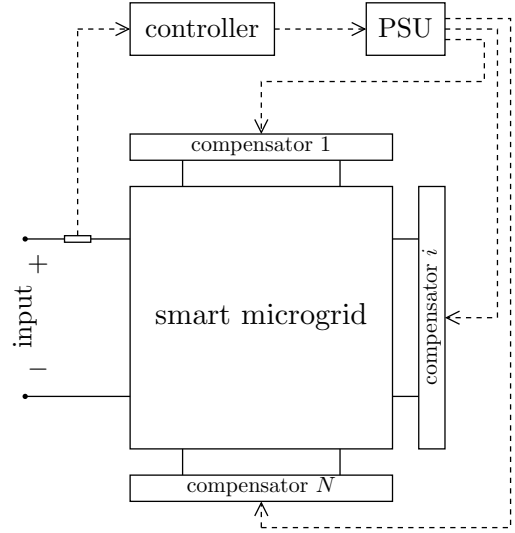


Fig. 1. Schematic representation of the controller structure proposed in Tedeschi et al. (2008).

Both residential and industrial users may require a sinusoidal current which is not in phase with voltage. A convenient description for that consists in stating that they demand reactive power together with active power, associated with out-of-phase and in-phase components of the current respectively. Precisely, the active power  $p$  and reactive power  $q$  delivered at node  $i$  are

$$p_i(t) = U_i I_i \cos \phi, \quad q_i(t) = U_i I_i \sin \phi,$$

where  $\phi$  is the phase difference  $\theta_i^u - \theta_i^i$ .

These power terms can be defined also in the case in which signals are not sinusoidal and in which they can be considered perturbed versions of periodic signals with period  $T_0 = 2\pi/\omega_0$ . Following Tenti and Mattavelli (2003), we define the *homo-integral* of a generic function  $x(t)$  as

$$\hat{x}(t) = \omega_0 (X(t) - \bar{X}(t))$$

where  $X(t) = \int_0^t x(\tau) d\tau$  and  $\bar{X}(t) = \frac{1}{T_0} \int_t^{t+T_0} X(\tau) d\tau$ .

By introducing the scalar product (function of time)

$$\langle x, y \rangle_t = \frac{1}{T_0} \int_t^{t+T_0} x(\tau) y(\tau) d\tau \quad (4)$$

we can then define active and reactive powers in this more general framework as

$$p_i(t) = \langle u, i \rangle_t, \quad q_i(t) = \langle \hat{u}, i \rangle_t.$$

Like active power flows, reactive power flows contribute to power losses on the transmission and distribution lines, cause voltage drop in the network, and may lead to grid instability (see Kundur (1994) and references therein).

Reactive power is not a “real” physical power, meaning that to produce it there is no energy conversion involved nor fuel costs. It is therefore preferable to minimize reactive power flows by producing reactive power as close as possible to the users that need it.

One possible approach has been proposed in Tedeschi et al. (2008), and is sketched in Figure 1. It consists in a centralized controller that measures the reactive power flow at the input port of the microgrid, i.e. where the microgrid connects with the main grid. According to this

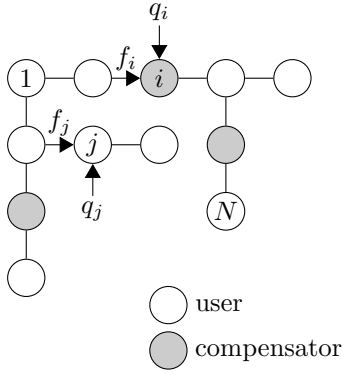


Fig. 2. Tree of users and compensators. The direction of the flow and the naming convention is indicated for node  $i \in \mathcal{C}$  and  $j \in \mathcal{U}$ .

measurement, the controller produces a reference for the amount of reactive power that has to be produced inside the microgrid. This reference has to be split by a power sharing unit (PSU) among those agents in the network that can produce a commanded amount of reactive power (*compensators*), in a way that minimizes reactive power flows in the microgrid. The number of compensators in a microgrid can be very large, as the electronic interface of any distributed generator (wind turbines, combined heat and power generators, micro hydroelectric, solar panels) can also produce reactive power at no additional cost.

Let the electrical connections in the microgrid be described by a tree of  $\bar{N}$  agents (see Figure 2). Each agent injects a quantity  $q_i(t)$  of reactive power into the network.  $N$  of them (the compensators, whose indices belong to  $\mathcal{C}$ ) can be commanded to inject a given amount of reactive power, while the other nodes (users, whose indices belong to  $\mathcal{U}$ ) inject (or are supplied with, if negative) a fixed and unknown amount. Flows on the tree edges are oriented outbound with respect to the tree root (node 1), and indexed as in Figure 2. Reactive power obeys regular flow conservation equations.

As power losses are a quadratic function of the reactive power flowing on a line, the optimization problem of having minimal power losses in the microgrid corresponds to minimizing

$$F(f_2, \dots, f_N) = \sum_{i=2}^N f_i^2 k_i$$

where  $k_i$  is the resistance of the edge  $i$  (which goes linearly with the length of the line). The constraints are

- $\sum_{i \in \mathcal{C} \cup \mathcal{U}} q_i = 0$
- $f_i = f_i(\mathbf{q}) = \sum_{j \in \mathcal{F}_i} q_j$ , for  $i = 2, \dots, N$

where the first constraint enforces reactive power conservation in the network (assuming that the amount of reactive power lost in the distribution lines is negligible compared to the amount supplied to users), and the second set of constraints allows to express the flow on each edge as the sum of the reactive power injected by a subset  $\mathcal{F}_i$  of the nodes, as illustrated in Figure 3.

By stacking all the flows in a vector  $\mathbf{f}$  we can then obtain the linear form  $\mathbf{f} = \mathbf{A}\mathbf{q} + \mathbf{B}\mathbf{q}'$ , where  $\mathbf{q}$  and  $\mathbf{q}'$  contain all  $q_i$ 's for  $i \in \mathcal{C}$  and  $i \in \mathcal{U}$ , respectively. By defining

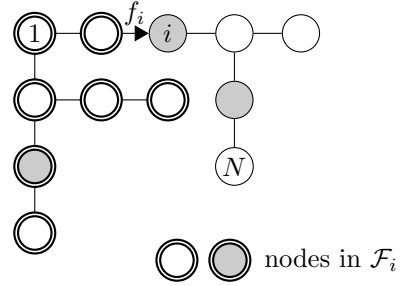


Fig. 3. The subset  $\mathcal{F}_i$ , defined as the set of nodes containing the root for which the edge  $i$  is the only bridge.

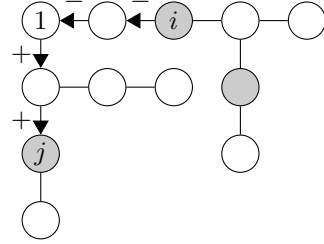


Fig. 4. Dependence of the gradient  $\mathbf{g}$  on the flows in the tree; e.g.  $g_i - g_j$  equals to the flow on the path from  $i$  to  $j$ . According to the given convention, flows have to be counted with negative sign when going upwards in the tree, as the signs show.

$\mathbf{K} = 2 \text{diag}(k_2, \dots, k_N)$ , we can rewrite the problem as

$$\begin{aligned} \min F(\mathbf{q}) &= \mathbf{q}^T \frac{\mathbf{A}^T \mathbf{K} \mathbf{A}}{2} \mathbf{q} + \mathbf{q}'^T \mathbf{B}^T \mathbf{K} \mathbf{A} \mathbf{q} \\ &= \mathbf{q}^T \frac{\mathbf{M}}{2} \mathbf{q} + \mathbf{m}^T \mathbf{q} \\ \text{subject to } \mathbf{1}^T \mathbf{q} &= -\mathbf{1}^T \mathbf{q}' = c \end{aligned}$$

and therefore we have casted the optimal reactive power flow problem into the framework described in Section 2.

The gradient of  $F(\mathbf{q})$  can be expressed as  $\mathbf{g}(\mathbf{q}) = \mathbf{A}^T \mathbf{K} \mathbf{A} \mathbf{q} + \mathbf{A}^T \mathbf{K} \mathbf{B} \mathbf{q}' = \mathbf{A}^T \mathbf{K} \mathbf{f}$ , where

$$\mathbf{A}^T \mathbf{K} \mathbf{f} = \begin{bmatrix} \vdots \\ \sum_{i \in \mathcal{E} - \mathcal{P}_i} k_i f_i \\ \vdots \end{bmatrix} \quad (5)$$

and  $\mathcal{P}_i \subseteq \mathcal{E}$  is the path from the root to node  $i$ . Let us consider the difference  $g_i(\mathbf{q}) - g_j(\mathbf{q})$ , where  $g_i(\mathbf{q})$  and  $g_j(\mathbf{q})$  are the elements of  $\mathbf{g}(\mathbf{q})$  corresponding to compensators  $i$  and  $j$ . As Figure 4 shows, this difference corresponds to

$$g_i(\mathbf{q}) - g_j(\mathbf{q}) = \sum_{\ell \in \mathcal{P}_{ij}} \delta_\ell(i, j) k_\ell f_\ell(\mathbf{q}) \quad (6)$$

where  $\mathcal{P}_{ij} \subseteq \mathcal{E}$  is the path from node  $i$  to node  $j$ , and  $\delta_\ell(i, j) \in \{+1, -1\}$  depends on whether the edge  $i$  appears in forward or backward direction in the path from  $i$  to  $j$ .

Let us suppose that each node is capable of measuring the root-mean-square value  $v_i(t) = \|u_i\|_t$  of the voltage at its location, and consider the steady state response of the network to a constant  $\mathbf{q}$  (corresponding to sinusoidal voltages and currents across the network). The voltage drop across edge  $(i, j)$  is described by  $v_j - v_i = f_j - f'_j$  where  $i$  is the parent of node  $j$ , units of measurement

have been properly normalized, and  $f'_j$  is the voltage drop due to the active power flow on edge  $(i, j)$ . If the lines' reactance is larger than the lines' resistance, then  $f'_j \ll f_j$  and therefore  $f_j \approx v_j - v_i$ . By this assumption, we have  $g_i - g_j \approx v_i - v_j$  and therefore

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

where  $\mathbf{v}$  is the vector of all voltage measurements and  $\xi$  is an unknown scalar. While we did not consider this uncertainty of the gradient estimate in Section 2, this uncertainty is not harmful, as the uncertain term  $\xi \mathbf{1}$  is orthogonal to the constraint  $\mathbf{1}^T \mathbf{q} = c$ .

Notice that by solving the constrained optimization problem, we obtain orthogonality of the gradient  $\mathbf{g}$  with respect to the constraint, i.e.  $\mathbf{g} = \mu \mathbf{1}$  for some  $\mu$ . This corresponds to having constant voltage across the whole network. Therefore by neglecting the contribution of active power flows to voltage drop, we are considering the two problems of minimizing power losses and of achieving optimal (flattest) voltage profile, equivalent. These two objectives are not equivalent if the resistance of the lines is not negligible compared with their reactance.

#### 4. ITERATIVE OPTIMIZATION ALGORITHMS

Two main issues arise when trying to design a distributed optimization algorithm for large-scale complex systems like the one described in Section 3.

The first issue is the fact that the agents do not know the structure of the whole system (the number of agents, their connections in the communication graph, the underlying physical system, etc.). Every agent has a local knowledge of this information, and in many cases there is no central unit that has a broader view of the system. Moreover, it is often the case that the structure of the systems changes in time, due to some external events, reconfiguration, node appearance and disappearance.

The second issue is given by the communication capabilities among nodes, resulting in constraints on information and decision sharing among the agents. Agents willing to coordinate their behavior and exchange data may be forced to interact with a smaller subset of neighbors, and in some cases to deal with quantization, data rate constraints, and unreliable communication.

We will deal with the issue of algorithm distribution among agents in the next section. In this section we will focus on the first issue, reviewing some optimization methods for convex optimization, specializing them to the linearly constrained case, and discussing their effectiveness for large-scale complex systems.

The class of optimization problems introduced in Section 2 is generally tackled via gradient-driven iterative methods (see for example Boyd and Vandenberghe (2008)). A general formulation of the iterative update law of these methods is the following:

$$\begin{aligned} \mathbf{q}^+ &= \mathbf{q} - \mathbf{\Gamma} \mathbf{g}(\mathbf{q}) \\ \mathbf{\Gamma}^+ &= \Phi(\mathbf{q}, \mathbf{\Gamma}, \mathbf{g}(\mathbf{q})). \end{aligned} \quad (7)$$

where  $\mathbf{\Gamma}$  is an  $N \times N$  gain matrix<sup>1</sup>.

<sup>1</sup> In this and in the following sections, we introduced the shorter notation  $\mathbf{q}^+ = \mathbf{q}(t_{k+1})$  and  $\mathbf{q} = \mathbf{q}(t_k)$  (and similarly for other

In the special, although interesting, case in which feasibility of the decision variable is required at any iteration, the gain matrix  $\mathbf{\Gamma}$  must satisfy

$$\mathbf{a}^T \mathbf{\Gamma} \mathbf{u} = 0 \quad \text{for all } \mathbf{u} \in \mathbb{R}^N.$$

The next lemma show what is the best choice for  $\mathbf{\Gamma}$ , if the Hessian  $\mathbf{M}$  is fully known.

*Lemma 1.* (Constrained Newton algorithm). Let  $\mathbf{q}$  be a feasible point for the optimization problem (3). Assume

$$\mathbf{\Gamma} = \mathbf{M}^{-1} - \frac{\mathbf{M}^{-1} \mathbf{a} \mathbf{a}^T \mathbf{M}^{-1}}{\mathbf{a}^T \mathbf{M}^{-1} \mathbf{a}}. \quad (8)$$

Then  $\mathbf{q}^+$  defined by (7) is the solution of the constrained optimization problem (i.e. the algorithm converges in one step).

**Proof.** To show that  $\mathbf{q}^+$  is the solution of the constrained optimization problem, we have to show that

- $\mathbf{q}^+$  is feasible
- the gradient  $\mathbf{g}(\mathbf{q}^+)$  is orthogonal to the constraint.

The first claim is true for any  $\mathbf{\Gamma}$  in the form  $\mathbf{H} - \frac{\mathbf{H} \mathbf{a} \mathbf{a}^T \mathbf{H}}{\mathbf{a}^T \mathbf{H} \mathbf{a}}$ , for any  $\mathbf{H}$ , as

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

For the second claim we have to prove that  $\mathbf{g}(\mathbf{q}^+) \in \ker(\mathbf{a}^T)^\perp = \text{Im}(\mathbf{a})$ . Indeed, we have

$$\begin{aligned} \mathbf{g}(\mathbf{q}^+) &= \mathbf{m} + \mathbf{M} \mathbf{q}^+ \\ &= \mathbf{m} + \mathbf{M} \mathbf{q} - \mathbf{M} \mathbf{M}^{-1} \mathbf{g}(\mathbf{q}) \\ &\quad + \frac{\mathbf{M} \mathbf{M}^{-1} \mathbf{a}}{\mathbf{a}^T \mathbf{M}^{-1} \mathbf{a}} \mathbf{a}^T \mathbf{M}^{-1} \mathbf{g}(\mathbf{q}) \\ &= \mathbf{a} \frac{\mathbf{a}^T \mathbf{M}^{-1} \mathbf{g}(\mathbf{q})}{\mathbf{a}^T \mathbf{M}^{-1} \mathbf{a}} \in \text{Im}(\mathbf{a}). \end{aligned}$$

□

If we know an approximation  $\mathbf{H}$  of the inverse of the Hessian  $\mathbf{M}$ , we can plug  $\mathbf{H}$  in (8), and we obtain the *approximate Newton update step*

$$\begin{aligned} \mathbf{q}^+ &= \mathbf{q} - \left( \mathbf{H} - \frac{\mathbf{H} \mathbf{a} \mathbf{a}^T \mathbf{H}}{\mathbf{a}^T \mathbf{H} \mathbf{a}} \right) \mathbf{g}(\mathbf{q}) \\ &= \mathbf{q} - \mathbf{H} \mathbf{g}(\mathbf{q}) + \frac{\mathbf{a}^T \mathbf{H} \mathbf{g}(\mathbf{q})}{\mathbf{a}^T \mathbf{H} \mathbf{a}} \mathbf{H} \mathbf{a}. \end{aligned} \quad (9)$$

In the update step (9) two parts can be recognized:

$$\Delta \mathbf{q} = \mathbf{q}^+ - \mathbf{q} = \Delta \mathbf{q}_{\text{desc}} + \Delta \mathbf{q}_{\text{proj}} \quad (10)$$

where

- $\Delta \mathbf{q}_{\text{desc}} = -\mathbf{H} \mathbf{g}(\mathbf{q})$  is a descent step towards the optimum of the unconstrained quadratic problem
- $\Delta \mathbf{q}_{\text{proj}} = \frac{\mathbf{a}^T \mathbf{H} \mathbf{g}(\mathbf{q})}{\mathbf{a}^T \mathbf{H} \mathbf{a}} \mathbf{H} \mathbf{a}$  projects  $\mathbf{q} + \Delta \mathbf{q}_{\text{desc}}$  on the constraint (as proof of Lemma 1 shows, feasibility of  $\mathbf{q}^+$  is guaranteed regardless of the choice of  $\mathbf{H}$ ).

According to the available level of knowledge of the system, different degrees of approximation of the inverse of  $\mathbf{M}$  can be achieved, resulting in different algorithms.

On one hand, we saw in Lemma 1 that if  $\mathbf{M}^{-1}$  is completely known, it can be exploited to obtain the fastest quantities that appear in the algorithms), when this does not lead to confusion.

(one-step) convergence. However, Newton method requires that node  $i$  knows the whole  $i$ -th row of  $\mathbf{M}^{-1}$ . This may not be possible in large-scale systems, and jeopardizes the possibility of node insertion and removal.

On the other hand, when minimal knowledge is available, a diagonal  $\mathbf{H} = \alpha \mathbf{I}$  can be used, obtaining the specialization of *steepest descent method* to the linearly constrained case. Unfortunately, the steepest descent method may require a large number of iterations to converge, depending on the condition number of the Hessian  $\mathbf{M}$  (Boyd and Vandenberghe (2008)). This results to be a major problem in our framework, where estimating the gradient  $\mathbf{g}(\mathbf{q})$  for a given  $\mathbf{q}$  requires that the systems is driven into the state  $\mathbf{q}$ , and consists in measuring the steady state response of a dynamical system (therefore introducing an implicit tradeoff between accuracy and time delay in the measurement).

*Quasi-Newton methods* (see for example Dennis and Schnabel (1983) and Nocedal and Wright (2006)) have instead the useful feature of building an estimate of the inverse of the Hessian from the previous steps of the algorithm. In the framework of complex systems, these methods deserve special attention, as they require minimal knowledge of the problem, and they can deal with time-varying structures via their adaptive behavior.

Consider the following specialization of *Broyden's algorithm* (belonging to the class of quasi-Newton methods) for the constrained optimization problem (3):

$$\begin{aligned} \mathbf{q}^+ &= \mathbf{q} - \mathbf{G}\mathbf{d} \\ \mathbf{G}^+ &= \mathbf{G} + \frac{[\Delta\mathbf{q} - \mathbf{G}\Delta\mathbf{d}]\Delta\mathbf{d}^T}{\Delta\mathbf{d}^T\Delta\mathbf{d}} \end{aligned} \quad (11)$$

where  $\mathbf{d} = \Omega_{\mathbf{a}}\mathbf{g}$ ,  $\Omega_{\mathbf{a}} = \mathbf{I} - \mathbf{a}\mathbf{a}^T/\mathbf{a}^T\mathbf{a}$ , is the projection of the gradient on the constraint, and

$$\begin{aligned} \Delta\mathbf{d} &= \mathbf{d}^+ - \mathbf{d} \\ \Delta\mathbf{q} &= \mathbf{q}^+ - \mathbf{q}. \end{aligned}$$

Suppose that  $\mathbf{G}$  is initialized as  $\alpha\mathbf{I}$ , for some  $\alpha > 0$ , and that it is not updated if  $\|\Delta\mathbf{d}\| = 0$ .<sup>2</sup>

It is easy to see that (11) is a special case of (7), in which  $\mathbf{\Gamma} = \mathbf{G}\Omega_{\mathbf{a}}$  and in which the rank-1 update for  $\mathbf{G}$  satisfies the *secant condition*

$$\mathbf{G}^+\Delta\mathbf{d} = \Delta\mathbf{q}.$$

The following lemmas will be helpful in proving the global convergence of this algorithm<sup>3</sup>.

*Lemma 2.* For any  $\mathbf{G}(t_k)$  returned by the algorithm (11), and for all  $\mathbf{u} \in \mathbb{R}^N$ ,

$$\mathbf{a}^T\mathbf{u} = 0 \quad \Rightarrow \quad \mathbf{a}^T\mathbf{G}(t_k)\mathbf{u} = 0.$$

**Proof.** Consider the base case  $\mathbf{G}(0) = \alpha\mathbf{I}$ . We have

$$\mathbf{a}^T\mathbf{G}(0)\mathbf{u} = \alpha\mathbf{a}^T\mathbf{u} = 0.$$

Let us now suppose that the condition is verified for  $\mathbf{G}(t_k)$ , and consider  $\mathbf{G}(t_{k+1})$ . We have

$$\begin{aligned} \mathbf{a}^T\mathbf{G}(t_{k+1})\mathbf{u} &= \mathbf{a}^T\mathbf{G}(t_k)\mathbf{u} + \frac{\mathbf{a}^T\Delta\mathbf{q}(t_k)\Delta\mathbf{d}(t_k)\mathbf{u}}{\Delta\mathbf{d}^T(t_k)\Delta\mathbf{d}(t_k)} \\ &\quad - \frac{\mathbf{a}^T\mathbf{G}(t_k)\Delta\mathbf{d}(t_k)\Delta\mathbf{d}^T(t_k)\mathbf{u}}{\Delta\mathbf{d}^T(t_k)\Delta\mathbf{d}(t_k)} \\ &= 0, \end{aligned}$$

where we used  $\Delta\mathbf{q}(t_k) = \mathbf{G}(t_k)\mathbf{d}(t_k)$ , and the fact that  $\mathbf{a}^T\Delta\mathbf{d}(t_k) = 0$  and  $\mathbf{a}^T\mathbf{d}(t_k) = 0$ . Therefore by induction the thesis is verified.  $\square$

Lemma 2 guarantees that  $\mathbf{a}^T\Delta\mathbf{q} = 0$ , or in other words it guarantees that the update step for  $\mathbf{q}$  returns always a feasible point for the constrained optimization problem, if  $\mathbf{q}(0)$  is feasible.

*Lemma 3.* The estimate  $\mathbf{G}(t_{k+1})$  has full rank as long as  $\mathbf{G}(0)$  is full rank and  $\mathbf{d}(t_j) \neq 0$  for  $0 \leq t_j \leq t_k$ .

**Proof.** By Sherman-Morrison formula,  $\mathbf{G}(t_{k+1})$  has full rank whenever  $\mathbf{G}(t_k)$  is full rank and

$$\Delta\mathbf{d}^T(t_k)\mathbf{G}(t_k)^{-1}\Delta\mathbf{q}(t_k) \neq 0.$$

By substituting we have

$$\begin{aligned} \Delta\mathbf{d}^T(t_k)\mathbf{G}(t_k)^{-1}\Delta\mathbf{q}(t_k) &= -\Delta\mathbf{d}^T(t_k)\mathbf{d}(t_k) \\ &= \mathbf{d}^T(t_k)\mathbf{G}(t_k)\mathbf{M}\Omega_{\mathbf{a}}\mathbf{d}(t_k) \\ &= \mathbf{d}^T(t_k)\mathbf{G}(t_k)\mathbf{M}\mathbf{d}(t_k) \end{aligned}$$

which is zero if and only if  $\mathbf{d}(t_k) = 0$ . Therefore by induction  $\mathbf{G}(t_{k+1})$  has full rank.  $\square$

*Lemma 4.* (Lemma 2.1 in Gay (1979)). Consider the  $k$ -th iteration of algorithm (11). If

$$\mathbf{d}(t_k) \text{ and } \Delta\mathbf{d}(t_{k-1}) \text{ are linearly independent,}$$

then for  $1 \leq j \leq \lfloor (k+1)/2 \rfloor$ , the  $j+1$  vectors

$$[\Omega_{\mathbf{a}}\mathbf{M}\mathbf{G}(t_{k-2j+1})]^i \mathbf{d}(t_{k-2j+1}), \quad 0 \leq i \leq j,$$

are linearly independent.

**Proof.** The proof is given in Gay (1979).  $\square$

We can now state the following result on the global, finite-time convergence of (11).

*Theorem 5.* Consider the algorithm (11) initialized with  $\mathbf{G}(0) = \alpha\mathbf{I}$ , with  $\alpha > 0$ , and  $\mathbf{q}(0)$  any feasible state. Then the algorithm converges in at most  $2N$  steps to the solution of the constrained quadratic problem (3).

**Proof.** By Lemma 4, there exists  $k$  with  $1 \leq k \leq 2N$  such that  $\mathbf{d}(t_k)$  and  $\Delta\mathbf{d}(t_{k-1})$  are linearly dependent. Trivially, if  $\mathbf{d}(t_k) = 0$ , this solves the optimization problem (3), as the gradient is orthogonal to the constraint and Lemma 2 ensures that  $\mathbf{q}(t_k)$  is a feasible point. If instead  $\Delta\mathbf{d}(t_{k-1}) = 0$ , then from the definition of  $\mathbf{d}$  we have  $\Omega_{\mathbf{a}}\mathbf{M}\Delta\mathbf{q}(t_{k-1}) = 0$ , and therefore

$$\mathbf{M}\mathbf{q}(t_k) = \mathbf{M}\mathbf{q}(t_{k-1}) + \beta\mathbf{a} \text{ for some } \beta \in \mathbb{R}.$$

Being  $\mathbf{M}$  invertible, this means that  $\Delta\mathbf{q}(t_{k-1}) = \beta\mathbf{M}^{-1}\mathbf{a}$ . By left multiplying both terms by  $\mathbf{a}^T$  and using Lemma 2, we get

$$\beta\mathbf{a}^T\mathbf{M}^{-1}\mathbf{a} = \mathbf{a}^T\Delta\mathbf{q}(t_{k-1}) = 0.$$

Therefore  $\beta = 0$  and  $\Delta\mathbf{q}(t_{k-1}) = 0$ . By Lemma 3 and (11), this implies that  $\mathbf{d}(t_j) = 0$  for some  $j \leq k-1$ , and therefore the solution has been reached in at most  $k \leq 2N$  steps. As a last case, suppose that  $\mathbf{d}(t_k)$  and  $\Delta\mathbf{d}(t_{k-1})$  are both non zero, but they are linearly dependent. Therefore there exists  $\lambda \neq 0$  such that

$$\mathbf{d}(t_k) = \lambda\Delta\mathbf{d}(t_{k-1}).$$

<sup>2</sup> It will be clear in the proof of Theorem 5 that if  $\|\Delta\mathbf{d}\| = 0$ , then the algorithm has converged.

<sup>3</sup> For these lemmas and for Theorem 3 we need to express time dependance explicitly.

From the algorithm equations and from the secant condition we have that  $\Delta \mathbf{d}(t_{k-1}) = \Omega_{\mathbf{a}} \mathbf{M} \Delta \mathbf{q}(t_{k-1}) = \Omega_{\mathbf{a}} \mathbf{M} \mathbf{H}(t_k) \Delta \mathbf{d}(t_{k-1})$ . The same is then true for  $\mathbf{d}(t_k)$ , yielding  $\mathbf{d}(t_k) = \Omega_{\mathbf{a}} \mathbf{M} \mathbf{G}(t_k) \mathbf{d}(t_k)$ . By rearranging the algorithm equations it is easy to see that  $\mathbf{d}(t_{k+1}) = \mathbf{d}(t_k) + \Omega_{\mathbf{a}} \mathbf{M} \Delta \mathbf{q}(t_k)$  and therefore

$$\mathbf{d}(t_{k+1}) = \mathbf{d}(t_k) - \Omega_{\mathbf{a}} \mathbf{M} \mathbf{G}(t_k) \mathbf{d}(t_k) = 0.$$

Even in this case,  $\mathbf{d}(t_{k+1}) = 0$  together with Lemma 2 guarantee that  $\mathbf{q}(t_{k+1})$  is the solution of the constrained optimization problem.  $\square$

## 5. ALGORITHM DISTRIBUTION

In this section we deal with the problem of distributing the algorithm among the nodes, in a way that is consistent with the capabilities of the nodes to gather information from neighbors according to the communication constraints given by the problem.

Consider the decomposition of the generic algorithm (7) into update laws for the single agents. The generic agent  $i$  has to perform the update

$$\begin{aligned} q_i^+ &= q_i - \Gamma_i^T \mathbf{g}(\mathbf{q}) \\ \Gamma_i^+ &= \Phi_i(\mathbf{q}, \Gamma, \mathbf{g}(\mathbf{q})). \end{aligned} \quad (12)$$

where  $\Gamma_i^T$  is the  $i$ -th row of  $\Gamma$ . It is not possible, in general, to implement (12) in a distributed manner, as both the update for  $q_i$  and for the vector  $\Gamma_i$  require quantities that are not available at node  $i$ :  $\mathbf{q}$ ,  $\mathbf{g}(\mathbf{q})$ , and  $\Gamma$ .

In the special case in which at every iteration of the algorithm

$$\Gamma_{ij} \neq 0 \quad \Rightarrow \quad (i, j) \in \mathcal{E}, \quad (13)$$

then the update law for  $q_i$  only requires information that can be gathered from the set of neighbors  $\mathcal{N}_i$ . The issue of distributing the algorithm among the agents reduces then to a sparsity condition on  $\Gamma$  (and a similar condition can be stated for the update law for  $\Gamma_i$ ).

However, note that the decision variables in the optimization problem in (3) are coupled in two ways: by a possibly non diagonal Hessian  $\mathbf{M}$ , and by the *complicating constraint* in which all the variables appear. For this reason, it is inherently hard to solve the optimization problem via some purely local laws.

To tackle this issue, we introduce some shared piece of information among all agents: let  $\mathbf{x} \in \mathbb{R}^p$  be an auxiliary *fusion* vector, function of the whole system state, and suppose that all the nodes agree on the value of  $\mathbf{x}$ .

The local update laws (12) can then be replaced by

$$\begin{aligned} q_i^+ &= q_i - \gamma_i(q_j, g_j(\mathbf{q}), j \in \mathcal{N}_i; \boldsymbol{\eta}_i, \mathbf{x}) \\ \Gamma_i^+ &= \phi_i(q_j, g_j(\mathbf{q}), j \in \mathcal{N}_i; \boldsymbol{\eta}_i, \mathbf{x}). \end{aligned} \quad (14)$$

where  $\boldsymbol{\eta}_i$  is a local parameter vector. The fusion vector  $\mathbf{x}$  is a function of all the data available in the system:

$$\mathbf{x} = f(\mathbf{q}, \mathbf{g}(\mathbf{q}), \boldsymbol{\eta}_i, i \in \mathcal{V}). \quad (15)$$

Algorithm (14) is now consistent with the communication graph, because the update laws for both  $q_i$  and  $\boldsymbol{\eta}_i$  are function of local data ( $q_i, g_i(\mathbf{q}), \boldsymbol{\eta}_i$ ), of data that can be gathered from neighbors  $j \in \mathcal{N}_i$ , and of the fusion vector  $\mathbf{x}$ .

Of course, the way in which  $\mathbf{x}$  is computed and the way in which all agents agree on its value is a key point in the design of the algorithm. For example a finer time scale might exist: the fusion vector  $\mathbf{x}$  can be obtained as the result of another distributed algorithm, which exploits the same communication graph. This faster algorithm is initialized locally on the basis of the data stored in the nodes and on the basis of measured steady state of the underlying system. As it runs at a much faster pace, by the end of the period of time  $T$  it is able to implement the function  $f$  of the data and to guarantee that all nodes agree on a common  $\mathbf{x}$ .

Among the main algorithms that can be exploited to obtain the fusion vector  $\mathbf{x}$ , the tool of *average consensus* (as described for example in Olfati-Saber et al. (2007) and in Fagnani and Zampieri (2008)) is of particular interest.

Average consensus can be quite useful when dealing with optimization problems subject to linear equality constraints. Consider indeed the decomposition (10) of the update step in an approximate Newton method, consisting in an unconstrained descent step followed by its projection over the constraint:

$$\begin{aligned} \mathbf{q}^+ &= \mathbf{q} - \mathbf{H}\mathbf{g}(\mathbf{q}) + \frac{\mathbf{a}^T \mathbf{H}\mathbf{g}(\mathbf{q})}{\mathbf{a}^T \mathbf{H}\mathbf{a}} \mathbf{H}\mathbf{a} \\ &= \mathbf{q} - \mathbf{H}\mathbf{g}(\mathbf{q}) + x \mathbf{H}\mathbf{a}. \end{aligned}$$

The proposed decomposition has the major advantage that the scalar  $x$  on which all nodes have to agree, can be obtained via average consensus, once the communication graph is strongly connected. Indeed, if every node is capable of initializing a vector

$$\mathbf{z}^{(i)}(0) = \begin{bmatrix} a_i \mathbf{H}_i \mathbf{g}(\mathbf{q}) \\ a_i \mathbf{H}_i \mathbf{a} \end{bmatrix}, \quad (16)$$

where  $a_i$  and  $\mathbf{H}_i$  are the  $i$ -th component of  $\mathbf{a}$  and the  $i$ -th row of  $\mathbf{H}$ , respectively, then by running an average consensus algorithm the nodes eventually agree on the quantity

$$\bar{\mathbf{z}} = \begin{bmatrix} \frac{1}{N} \mathbf{a}^T \mathbf{H}\mathbf{g}(\mathbf{q}) \\ \frac{1}{N} \mathbf{a}^T \mathbf{H}\mathbf{a} \end{bmatrix}$$

from which every node can obtain the desired quantity  $x = \frac{\mathbf{a}^T \mathbf{H}\mathbf{g}(\mathbf{q})}{\mathbf{a}^T \mathbf{H}\mathbf{a}}$ .

Therefore if we choose an estimate  $\mathbf{H}$  of  $\mathbf{M}^{-1}$  that satisfies the sparsity constraint given by the communication graph, i.e.

$$\mathbf{H}_{ij} \neq 0 \quad \Rightarrow \quad (i, j) \in \mathcal{E},$$

then both the computation of  $\Delta \mathbf{q}_{\text{desc}}$  and the initialization of  $\mathbf{z}$  for the computation of  $\Delta \mathbf{q}_{\text{proj}}$  requires only communication between neighbors in  $\mathcal{G}$ .

## 6. DISTRIBUTED QUASI-NEWTON METHODS

The approach presented in last section for algorithm distribution will now be applied to the Broyden quasi-Newton method proposed in Section 4.

Let us define  $\mathbf{g}^{(i)}$  the the part of the gradient corresponding to the set of the neighbors  $\mathcal{N}_i = \{j_1, \dots, j_{n_i}\}$  of node  $i$ :

$$\mathbf{g}^{(i)} \in \mathbb{R}^{n_i}, \quad g_\ell^{(i)} = g_{j_\ell} \text{ for } \ell = 1, \dots, n_i. \quad (17)$$

Following the sparse secant method proposed in Dennis and Schnabel (1983) (Theorem 11.2.1) and including a projection step as proposed in Section 5, let us consider the following algorithm:

$$\begin{aligned} \mathbf{q}^+ &= \mathbf{q} - \mathbf{H}\mathbf{g}(\mathbf{q}) + \frac{\mathbf{a}^T \mathbf{H}\mathbf{g}(\mathbf{q})}{\mathbf{a}^T \mathbf{H}\mathbf{a}} \mathbf{H}\mathbf{a} \\ \mathbf{H}^+ &= \mathbf{H} + \mathcal{P}_{\mathcal{E}} \left[ \mathbf{D}^+ (\Delta\mathbf{q} - \mathbf{H}\Delta\mathbf{g}(\mathbf{q})) \Delta\mathbf{g}^T(\mathbf{q}) \right] \end{aligned} \quad (18)$$

where  $\mathcal{P}_{\mathcal{E}}$  is the projection operator on the sparsity constrain induced by  $\mathcal{E}$ :

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

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

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

It is easy to see that (18) corresponds to (11) when the graph is complete, with the only difference that in the complete-graph case, thanks to Lemma 2, the projection on the constrain is performed on the measured gradient and not on the descent step.

Algorithm (18) can be distributed among the agents obtaining the following update equations for the update of  $q_i$  and  $\mathbf{H}_i \in \mathbb{R}^{n_i}$ .

$$\begin{aligned} q_i^+ &= q_i - \mathbf{H}_i^T \mathbf{g}^{(i)}(\mathbf{q}) + x \mathbf{H}_i^T \mathbf{a}^{(i)} \\ \mathbf{H}_i^+ &= \mathbf{H}_i + \left[ \Delta q_i - \mathbf{H}_i^T \Delta \mathbf{g}^{(i)}(\mathbf{q}) \right] \frac{\Delta \mathbf{g}^{(i)}(\mathbf{q})}{\Delta \mathbf{g}^{(i)T}(\mathbf{q}) \Delta \mathbf{g}^{(i)}(\mathbf{q})} \end{aligned}$$

where  $\mathbf{a}^{(i)}$  is defined similarly to the definition (17) of  $\mathbf{g}^{(i)}$ , and  $x = \bar{z}_1/\bar{z}_2$ , where  $\bar{z}_{1,2}$  are the elements of the 2-dimensional vector resulting from the average consensus algorithm initialized as

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

Both the measure of  $\mathbf{v}$  for the estimation of  $\mathbf{g}(\mathbf{q})$ , and the initialization of  $\mathbf{z}$ , take place as soon as the steady state response of the underlying system is available. Note that the update laws for  $q_i$ 's and  $\mathbf{H}_i$ 's are consistent with the communication graph, and memory requirements for every node scale with the number of neighbors in the communication graph.

The effectiveness of the quasi-Newton algorithm subject to sparsity constraints depends of course on the structure of the inverse of the Hessian, and therefore on the particular optimization problem that the algorithm has to solve. In the next section we will show how the problem of optimal reactive power compensation is a notable example in this sense.

The proof that we presented for the case of a complete communication graph cannot be adapted to this other case, as it strongly relies on some algebraic properties of the descent step that are lost as soon as we enforce sparsity on the Hessian inverse estimates. Our claim is that in the case of non-complete communication graph only local convergence is achieved; simulation results (presented in the next section) show that the algorithm behaves well on the testbed that we are considering.

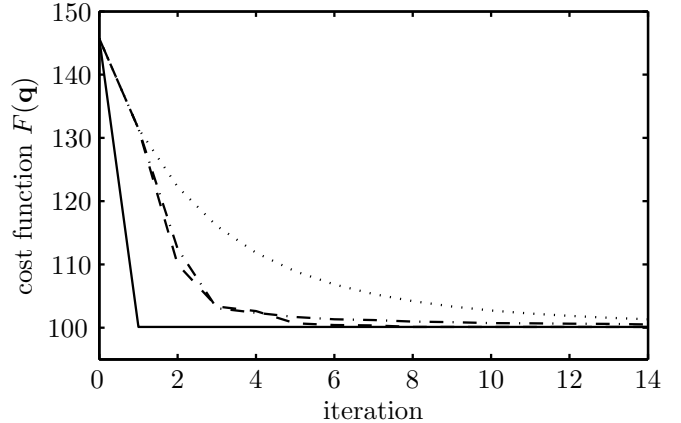


Fig. 5. Comparison of different algorithms: Newton method (solid), quasi-Newton method with complete communication graph (dashed), quasi-Newton method with sparsity constraint (dash-dotted), steepest descent with fixed step length (dotted).

## 7. SIMULATIONS

Some simulation results will now be presented, showing how the proposed algorithms behave when applied to the problem of optimal reactive power compensation presented in Section 3.

We will assume that communications take place through the electrical lines, considering both the case in which every node can communicate with any other node (complete communication graph) and the case in which communication between two nodes is possible only if the electric path between these two nodes is short enough. This is the simplest model for a promising communication solution for smart grids, which is Power Line Communication.

Consider a microgrid of 33 nodes, 10 of which are compensators. Reactive power demands have been chosen according to a unitary variance normal distribution with negative mean  $-\sigma$  (standard deviation). The initial injected reactive power of the compensators have been normally distributed too.

The electrical connection tree has height 6, and every node that is not a leaf has an average of 2.4 children. Nodes that are connected by a path of length not greater than 4 are able to communicate. The induced communication graph results to be connected, but not complete.

In Figure 5 the behavior of the different proposed algorithms has been plotted. For the sake of fairness, the step length for the fixed-step-length, steepest descent algorithm, has been optimized for fastest convergence, and the same initial condition has been set for quasi-Newton methods. Note however that this choice for the steplength requires global knowledge of the problem: if this knowledge is not available or it is approximate, more conservative choices would be preferred (corresponding to a smaller steplength) and therefore the advantage of quasi-Newton methods would be even larger.

In the top left quadrant of Figure 6, the inverse of the Hessian has been plotted. One can see how the sparsity constraint induced by the communication graph (and plotted in the lower right quadrant) is meaningful in

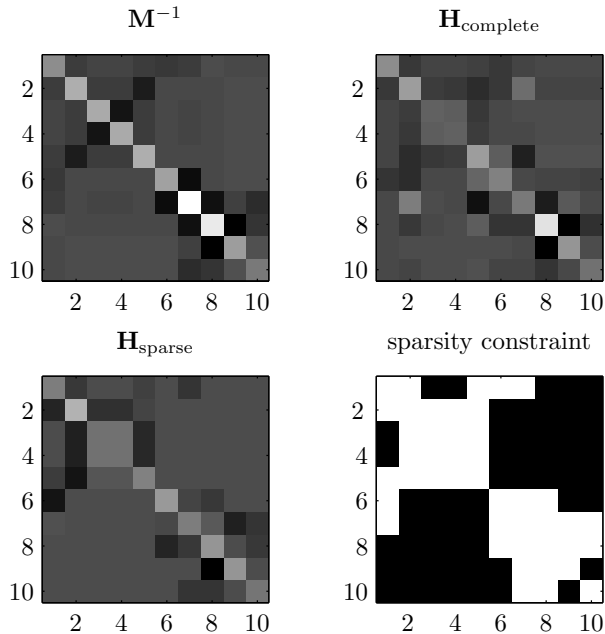


Fig. 6. Estimates of the inverse of the Hessian returned from the quasi-Newton algorithm, together with the real one and with the sparsity constraint.

describing the largest elements (in absolute value) of the inverse of Hessian. The other two quadrant of Figure 6 show the estimates of the Hessian returned by the quasi-Newton methods when the algorithm has converged.

## 8. CONCLUSION

In this paper we developed a distributed algorithm for the solution of a quadratic, convex, optimization problem with a linear equality constraint. Distributing the algorithm among the agents is complicated by the fact of having a non separable cost function and a global constraint. These two aspects have been tackled, respectively, by the construction of an estimate of the Hessian's inverse which is consistent with the communication graph, and by exploiting consensus to perform the projection step.

As a notable example of application of this algorithm, we described the problem of optimal reactive power compensation in a microgrid. As the gradient can be estimated locally (up to a term orthogonal to the constraint) and because the Hessian inverse results to be well described by an approximation consistent with the sparsity constraint induced by the communication graph, the proposed quasi-Newton method behaves well on this testbed.

While we considered a static optimization problem in this work, it is interesting to include the possibility of a dynamic optimization problem. This would allow us, for example, to include slowly time varying reactive power demands in our testbed, which is a very reasonable assumption. As the estimate of the Hessian's inverse obtained by the algorithm remains valid even if demands change (because reactive power demands affect the linear term and not the quadratic term of the cost function), we expect the quasi-Newton method to perform much better than simpler fixed-step-length, steepest descent algorithms.

Another direction of investigation is the introduction of approximate averaging in the consensus algorithms: while we assumed that the consensus returns the average of the nodes' initial conditions, this is an approximation because of the finite available time. The effects of this approximation is one of the very next directions of investigation.

## ACKNOWLEDGEMENTS

The authors wish to thank Carlo Fischione for his useful collaboration on the review of available distributed optimization techniques and for the kind hospitality at KTH.

## REFERENCES

- Bertsekas, D.P. and Tsitsiklis, J.N. (1997). *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific.
- Boyd, S. and Vandenberghe, L. (2008). *Convex Optimization*. Cambridge University Press.
- Dennis, Jr., J.E. and Schnabel, R.B. (1983). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall.
- Fagnani, F. and Zampieri, S. (2008). Randomized consensus algorithms over large scale networks. *IEEE Journal on Selected Areas in Communications*, 26(4), 634 – 649.
- Gay, D.M. (1979). Some convergence properties of broyden's method. *SIAM Journal on Numerical Analysis*, 16(4), 623–630.
- Ipakchi, A. and Albuyeh, F. (2009). Grid of the future. are we ready to transition to a smart grid? *IEEE Power and Energy Magazine*, 7(2), 52 –62.
- Isidori, A. (1995). *Nonlinear Control Systems*, volume 1. Springer Verlag, London, 3rd edition.
- Kelly, F.P., Maulloo, A.K., and Tan, D.K.H. (1998). Rate control for communication networks: Shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49, 237–252.
- Kundur, P. (1994). *Power System Stability and Control*. McGraw-Hill.
- Nedic, A. and Ozdaglar, A. (2009). Distributed sub-gradient methods for multi-agent optimization. *IEEE Transactions of Automatic Control*, 54(1), 48–61.
- Nocedal, J. and Wright, S.J. (2006). *Numerical Operation*. Springer.
- Olfati-Saber, R., Fax, J.A., and Murray, R.M. (2007). Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1), 215 – 233.
- Santacana, E., Rackliffe, G., Tang, L., and Feng, X. (2010). Getting smart. with a clearer vision of the intelligent grid, control emerges from chaos. *IEEE Power and Energy Magazine*, 8(2), 41–48.
- Tedeschi, E., Tenti, P., and Mattavelli, P. (2008). Synergistic control and cooperative operation of distributed harmonic and reactive compensators. In *Proceedings of the Power Electronic Specialists Conference*.
- Tenti, P. and Mattavelli, P. (2003). A time-domain approach to power term definitions under non-sinusoidal conditions. In *6th International Workshop on Power Definitions and Measurements under Non-Sinusoidal Conditions*. Milano, Italy.
- Tsitsiklis, J.N., Bertsekas, D.P., and Athans, M. (1986). Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9), 803–812.