Chapter 5
Distributed Optimization and Games: a Tutorial Overview

Bo Yang and Mikael Johansson

Abstract This chapter provides a tutorial overview of distributed optimization and game theory for decision-making in networked systems. We discuss properties of first-order methods for smooth and non-smooth convex optimization, and review mathematical decomposition techniques. A model of networked decision-making is introduced in which a communication structure is enforced that determines which nodes are allowed to coordinate with each other, and several recent techniques for solving such problems are reviewed. We then continue to study the impact of non-cooperative games, in which no communication and coordination are enforced. Special attention is given to existence and uniqueness of Nash equilibria, as well as the efficiency loss in not coordinating nodes. Finally, we discuss methods for studying the dynamics of distributed optimization algorithms in continuous time.

5.1 Introduction

We are interested in optimization algorithms that can be distributed across many decision-makers. The classical approach to distributed optimization has been decomposition: based on the specific structure of the objective function and constraints, the problem is decomposed into a number of subproblems. These subproblems can be solved independently, but typically require a coordinator to ensure that the local decisions converge to the global optimum; see Figure 5.1. Note how the problem structure imposes a certain computation and communication structure among the individual decision-makers.

Bo Yang
Department of Automation, Shanghai Jiao Tong University, China, e-mail: bo.yang@sjtu.edu.cn

Mikael Johansson
School of Electrical Engineering and ACCESS Linnaeus Center e-mail: mikaelj@ee.kth.se

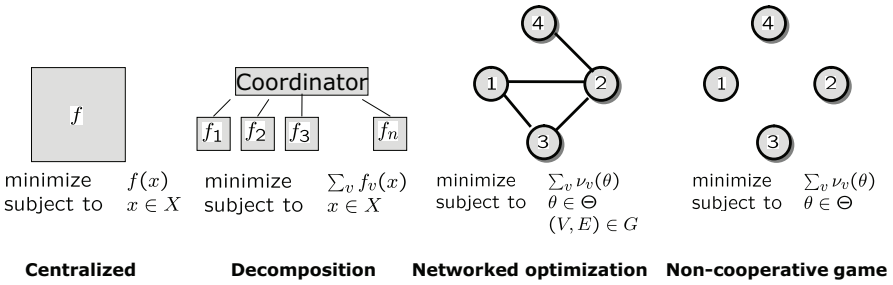

Fig. 5.1 Schematic illustration of centralized optimization, decomposition, networked optimization and non-cooperative optimization

In many emerging applications of distributed optimization, however, the situation is the reverse: the communication and computation structure is given and the implementation of a centralized coordinator is undesirable or infeasible. One example is systems where nodes can only coordinate their decisions with their immediate neighbors. In this case, we are restricted to use optimization algorithms that respect the communication structure; see Figure 5.1. In some applications, it is desirable to avoid coordination altogether, either because it is unlikely that nodes would actually cooperate, or as a way to eliminate complex coordination protocols and associated traffic overhead, see Figure 5.1. We will consider all the three classes of distributed optimization techniques in this chapter: decomposition, networked-optimization, and non-cooperative games. Although we do not make any restrictions on the computational model, it is good to note that often we will not have access to a closed-form mathematical expression for the per-node objective functions. Rather, these can only be evaluated by applying our best current decision to an underlying engineering system and observe its performance. We will hence also investigate the consequences of having a real system acting as oracle for our optimization algorithm.

This chapter is organized as follows: we first discuss gradient and subgradient method for convex optimization. Although these algorithms do not have the best theoretical properties, they are easy to implement and surprisingly robust to noise and errors. We then proceed to study mathematical decomposition techniques, focusing primarily on dual and primal decomposition. Next, we focus on networked optimization problems where an underlying communication graph dictates which nodes can coordinate with each other. Our final methodological discussion concerns game theory. Finally, the chapter is concluded by methods for studying the dynamics of optimization algorithms.

5.2 Convex Optimization Using First-order Methods

We consider optimization problem on the form

$$\begin{aligned} &\text{minimize } f_0(x) \\ &\text{subject to } x \in X \end{aligned} \tag{5.1}$$

Here $x \in \mathbb{R}^n$ is the vector of decision variables that must belong to the feasible set X and f_0 is the convex objective function that we would like to minimize. We will assume that X is closed, convex and non-empty and that $\mathbf{dom} f \subseteq X$. It is sometimes useful to characterize the constraint set explicitly. We then use the notation

$$\begin{aligned} &\text{minimize } f_0(x) \\ &\text{subject to } f_i(x) \leq 0 \quad i = 1,\ldots,m \\ &\qquad\qquad h_i(x) = 0 \quad i = 1,\ldots,p \end{aligned} \tag{5.2}$$

Hence, the feasible set is the set of x that satisfies the constraint equations, $X = \{x \mid f_i(x) \leq 0,\ h_i(x) = 0\}$. Note that for X to be convex, $f_i(x)$ have to be convex and $h_i(x)$ must be affine functions of x. There is a vast literature on methods for solving convex optimization problem; some good starting points include the text books [10, 5, 49, 53] and slightly more advanced text are [57, 24]. In recent years, interior-point methods have become the method of choice for solving large-scale convex programming, due to their attractive theoretical properties and strong practical performance. However, our focus is different: we look for methods that can be distributed across many nodes. In this case, first order (gradient) methods are easier to apply and will be the central workhorse throughout this chapter.

5.2.1 Gradient Methods for Smooth Problems

Let us start with the most basic problem of minimizing a smooth convex function without constraints, *i.e.*

$$\text{minimize } f_0(x)$$

The gradient method then takes the form

$$x^{(t+1)} = x^{(t)} - \alpha^{(t)} \nabla f_0(x^{(t)}) \tag{5.3}$$

i.e. new iterates are computed by adjusting the current iterate in the direction of the negative gradient. In its most basic form the step size is constant, *i.e.* $\alpha^{(t)} = \alpha$, and convergence is guaranteed using the following typical result.

Proposition 5.1. *Assume that $f_0(x)$ is a convex function with $\mathbf{dom} f_0 = \mathbb{R}^n$ whose gradient is Lipschitz continuous with constant $L > 0$, i.e.*

$$\|\nabla f_0(x) - \nabla f_0(y)\|_2 \leq L\|x - y\|_2 \qquad \forall x, y$$

Assume that f_0 has finite optimal value $f_0^\star$ with minimizer $x^\star$. Let $\{x^{(t)}\}$ be a sequence generated by the gradient iteration (5.3) with a constant stepsize $\alpha^{(t)} = \alpha$ satisfying $0 < \alpha \leq 1/L$. Then

$$f_0(x^{(t)}) - f_0^\star \leq \frac{1}{2\alpha t}\|x^\star - x^{(0)}\|_2^2$$

This result tells us several things. First, the gradient iteration improves the objective function value in each step. Second, the iterates converge asymptotically to the optimum. Third, ε accuracy can be achieved in $O(1/\varepsilon)$ iterations for any continuously differentiable function with Lipschitz continuous gradient.

Practical performance can be improved by time-varying step sizes, either found via line search in each iteration or predetermined ("open-loop"). Common open-loop step-size sequences include the square summable but not summable,

$$\sum_{t=1}^{\infty} \alpha^{(t)} = \infty, \quad \sum_{t=1}^{\infty} (\alpha^{(t)})^2 < \infty \tag{5.4}$$

such as $\alpha^{(t)} = a/(b+t)$ for $a > 0, b \geq 0$, or diminishing stepsizes

$$\sum_{t=1}^{\infty} \alpha^{(t)} = \infty, \quad \lim_{t\to\infty} \alpha^{(t)} = 0 \tag{5.5}$$

such as $\alpha^{(t)} = a/\sqrt{t}$ for $a > 0$. However, the complexity of the method is not improved but $O(1/\varepsilon)$ iterations are still needed to achieve ε accuracy.

An important performance measure of optimization algorithms is their convergence rate. If we assume that $f_0(x)$ is strongly convex, *i.e.* that there exist a positive constant l such that

$$l\|x - x^\star\|_2^2 \leq f_0(x - x^\star)$$

and that f_0 is twice continuously differentiable with Lipschitz-continuous gradient. Then, in a neighborhood of $x^\star$ where f can be approximated as

$$f_0(x) \approx f_0(x^\star) + f_0'(x)(x - x^\star) + f_0''\|x - x^\star\|_2^2$$

one can show that the distance between optimal point and the iterates produced by the gradient method decreases geometrically

$$\|x^{(t)} - x^\star\| \leq cq^t$$

for some positive constants c and q with $q < 1$. In the optimization literature, one then says that the method converges linearly, since the distance to the optimal set in iteration t is a linear function of the distance to the optimal set in iteration $t-1$. The convergence rate q of the basic gradient method can be improved by multi-step methods. One of the first such method was the heavy-ball method due to Polyak:

$$x^{(t+1)} = x^{(t)} - \alpha^{(t)} \nabla f_0(x^{(t)}) + \beta^{(t)}(x^{(t)} - x^{(t-1)}) \tag{5.6}$$

The following result reveals the optimal constant step-size parameters α and β and quantifies the speed-up compared to the classical gradient iteration [53].

Proposition 5.2. *Let $x^\star$ be a nonsingular minimum point of $f_0(x) : \mathbb{R}^n \mapsto \mathbb{R}$ and assume that f_0 is twice continuously differentiable and satisfies*

$$lI_n \succeq f_0''(x^\star) \leq LI_n$$

Then, we can find and $\varepsilon > 0$ such that for any $x(0), x(1)$ with $\|x(0) - x^\star\|_2 \leq \varepsilon$ and $\|x(1) - x^\star\|_2 \leq \varepsilon$, both the gradient and the heavy-ball method produce iterates that converge to $x^\star$ with geometric progression

$$\|x^{(t)} - x^\star\| \leq c(\delta)(q+\delta)^t, \qquad 0 \leq q < 1,\ 0 < \delta < 1 - q$$

For the gradient method, the smallest achievable value of q is $(L-l)/(L+l)$ obtained for $\alpha^{(t)} = 2/(L+l)$. For the heavy-ball method, the smallest achievable value of q is $(\sqrt{L} - \sqrt{l})/(\sqrt{L} + \sqrt{l})$ obtained for the step size parameters $\alpha^{(t)} = 4/(\sqrt{L} + \sqrt{l})^2$, $\beta^{(t)} = (\sqrt{L} - \sqrt{l})^2/(\sqrt{L} + \sqrt{l})^2$.

If the problem is ill-conditioned, *i.e.* if L/l is large, then heavy-ball improves the value of q by roughly a factor of $\sqrt{L/l}$. Since multi-step methods can improve the convergence rate of the gradient iteration, it is interesting to see if also the complexity can be improved from $O(1/\varepsilon)$ to the theoretical lower bound $O(1/\sqrt{\varepsilon})$. The answer to this question was given by Nesterov [47], who constructed a family of multi-step methods on the form

$$\begin{aligned} x^{(t)} &= \hat{x}^{(t-1)} - \alpha^{(t)} \nabla f_0(\hat{x}^{(t-1)}) \\ \hat{x}^{(t)} &= x^{(t)} + \beta^{(t)}(x^{(t)} - x^{(t-1)}) \end{aligned} \tag{5.7}$$

For appropriate choices of $\alpha^{(t)}$ and $\beta^{(t)}$, these methods have complexity $O(1/\sqrt{\varepsilon})$ and are thus order optimal. In most cases, $\alpha^{(t)}$ and $\beta^{(t)}$ are time-varying, which sometimes makes it inconvenient to use them in distributed optimization, but when f_0 is strongly convex, the constant step-sizes $\alpha^{(t)} = 1/L$ and $\beta^{(t)} = (\sqrt{L} - \sqrt{l})/(\sqrt{L} + \sqrt{l})$ also yield an optimal scheme.

The gradient methods can also be extended to deal with problems with constraints. In this case, one typically uses *projected gradient* methods,

$$x^{(t+1)} = P_X\{x^{(t)} - \alpha^{(t)} \nabla f_0(x^{(t)})\} \tag{5.8}$$

where $P_X\{x\}$ denotes the (Euclidean) projection of x onto the constraint set X. Similar convergence results hold as for the unprojected gradient method (see, *e.g.*, [5]).

Example 5.1. To get a feel for the different gradient methods introduced above, consider the problem of minimizing

$$f_0(x) = \sum_{k=1}^{n} k^2 \frac{x_k^2}{2}$$

Since the Lipschitz constant equals n^2, the gradient method uses $\alpha^{(t)} = 1/(n+1)^2$; the heavy-ball method uses $\alpha^{(t)} = 4/(n+1)^2, \beta^{(t)} = (n-1)^2/(n+1)^2$, and Nesterov's method uses the updates above with $\alpha^{(t)} = 1/n$. The figures below show the evolution objective function for $n = 1$ (left) and $n = 7$ (right). We can notice several things: first, the heavy-ball method is perfectly tuned towards a second-order quadratic function and has the best convergence rate for $n = 2$ and Nesterov's method does not show any advantage over the gradient scheme. Note how Nesterov's method is not a descent method as the objective function sometimes increases. Moving to $n = 7$ we see that the asymptotic convergence rate of the heavy-ball method is still superior, but that is too aggressive initially. The improvement of Nesterov's scheme compared to the ordinary gradient method is now significantly improved.

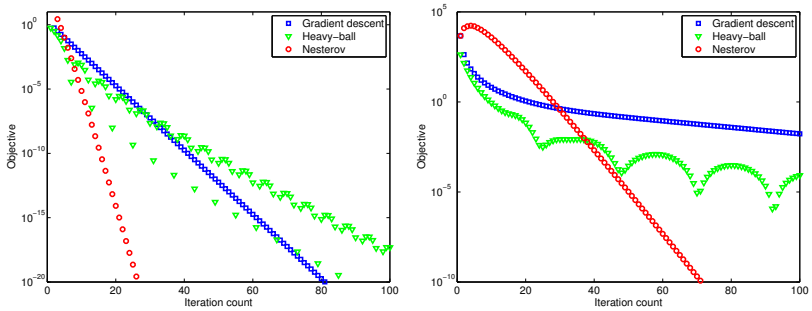

Fig. 5.2 Comparison of ordinary gradient descent, the heavy-ball method, and Nesterov's scheme

5.2.2 Subgradient Methods for Non-smooth Problems

In many situations when we do distributed optimization, we will need to work with objective functions that are not smooth. One natural extension of the gradient of a convex function is the concept of a *subgradient*. Like the gradient, the subgradient of a convex function is a global underestimator, *i.e.*

Definition 5.1. A vector $g \in \mathbb{R}^n$ is a *subgradient* of f at x if

$$f(y) \geq f(x) + g^T(y-x) \text{ for all } y \in \mathbf{dom} f$$

The set of all g satisfying the inequality is called the *subdifferential* of f at x and denoted $\partial f(x)$.

The *subgradient method* for unconstrained minimization of a non-smooth convex function now takes the form

$$x^{(t+1)} = x^{(t)} - \alpha^{(t)} g^{(t)} \tag{5.9}$$

where $g^{(t)} \in \partial f(x^{(t)})$, and the result corresponding to Proposition 5.1 reads

Proposition 5.3. *Assume that $f_0(x)$ is a convex function with $\mathbf{dom} f_0 = \mathbb{R}^n$ and that f_0 is Lipschitz continuous with constant $L > 0$, i.e.*

$$\|f_0(x) - f_0(y)\|_2 \leq L\|x - y\|_2 \qquad \forall x, y$$

Assume that f_0 has finite optimal value $f_0^\star$ with minimizer $x^\star$. Let $\{x^{(t)}\}$ be a sequence of generated by the subgradient iteration (5.9) with a constant stepsize $\alpha^{(t)} = \alpha$. Then

$$\min_{0 \leq k \leq t} f_0(x^{(k)}) - f_0^\star \leq \frac{\|x^\star - x^{(0)}\|_2^2 + t\alpha^2 L^2}{2\alpha t}$$

There are several important differences between this result and the corresponding result for the gradient method: the analysis only establishes properties of the *best* iterate found so far and says nothing about the most recent iterate. In fact, the subgradient method is not a descent method and the objective function typically does not improve in each iteration Moreover, under fixed stepsize even the best iterate does not converge to the optimum when $t \to \infty$: all we can say is that $\liminf_{t\to\infty} f_0(x^{(t)})$ is approximately $L^2\alpha/2$ suboptimal. If we change from constant to divergent stepsize sequences (5.4) or (5.5), then $\liminf_{t\to\infty} f_0(x^{(t)}) \to f_0^\star$. It is possible to show that the subgradient method can achieve ε-convergence in $O(1/\varepsilon^2)$ iterations, which is considerably slower than the gradient methods (see, *e.g.* [62]). Only recently, Nesterov has demonstrated how smoothing techniques can be applied to non-smooth optimization to recover the $O(1/\varepsilon)$ complexity of the basic gradient method [48].

It is possible to make slightly stronger statements about the subgradient method by considering the *Cesàro* averages

$$\tilde{x}^{(t)} = \frac{\sum_{k=0}^{t} \alpha^{(k)} x^{(k)}}{\sum_{k=0}^{t} \alpha^{(k)}}$$

Under the same conditions as Proposition 5.3, if $x(t)$ are generated by the subgradient iteration (5.9) with a constant stepsize $\alpha^{(k)} = \alpha$, then

$$f_0(\tilde{x}^{(t)}) - f_0^\star \leq \frac{\|x(0) - x^\star\|_2^2 + t\alpha^2 L^2}{2\alpha t}$$

This inequality establishes that $\lim_{t\to\infty} f_0(\tilde{x}^{(t)}) \leq f^\star + \alpha L^2/2$. Similarly, the corresponding analysis for divergent step-size sequences (5.4) or (5.5) establishes that the Cesàro averages converge asymptotically to the optimal set.

As for the gradient method, the subgradient method can be extended to handle constrained minimization by projecting the iterates onto the constraint set. The *projected subgradient method* takes the form

$$x^{(t+1)} = P_X \left\{ x^{(t)} - \alpha^{(t)} g^{(t)} \right\}$$

Again, very convergence results similar to those of the (unprojected) subgradient method can be established also for the projected subgradient method [62].

5.2.3 Incremental Subgradient Methods

In many applications, we will encounter objective functions on the form

$$f_0(x) = \sum_{i=1}^{N} f_{0i}(x)$$

where each component f_{0i} can be evaluated in parallel (on different machines, or by different decision-makers). The projected subgradient method

$$x^{(t+1)} = P_X \left\{ x^{(t)} - \alpha^{(t)} \sum_{i=1}^{N} g_{0i}^{(t)} \right\}$$

would need to collect all the subgradients g_{0i} of the individual objective function components f_{0i} at the current iterate $x^{(t)}$ to perform an iteration. In contrast, *incremental subgradient methods* cycle through the components and make incremental changes of the iterate in the direction of the negative subgradient of each component,

$$\begin{aligned} x_i^{(t)} &= P_X \left\{ x_{i-1}^{(t)} - \alpha^{(t)} g_{0i}^{(t)} \right\}, \; i = 1, \ldots, N \\ x_0^{(t+1)} &= x_N^{(t)} \end{aligned} \tag{5.10}$$

Here $x_i^{(t)}$ is the iterate computed by accounting for component i, and the subgradient is evaluated at the iterate produced by component $i-1$ at the same iteration, *i.e.*

$$g_{0i}^{(t)} \in \partial f_{0i}(x_{i-1}^{(t)})$$

The first line of (5.10) describes how the inner iterations cycle through components, one-by-one in a given order, and make incremental updates in the iterate. The second line describes how a new round of iterations are started by passing the iterate produced by the last component to the first. By analyzing how the iterates behave at the beginning of each iteration round, very similar convergence results as for the standard subgradient method can be obtained

Proposition 5.4. *Assume that $f_{0i}(x)$ are convex functions with $X \subseteq \mathbf{dom} f_{0i}$ and that f_{0i} are Lipschitz continuous with constants $L_i > 0$, i.e.*

$$\|f_{0i}(x) - f_{0i}(y)\|_2 \leq L_i \|x - y\|_2 \qquad \forall x, y$$

Assume that $f_0 = \sum_i f_{0i}$ has finite optimal value $f_0^\star$ with minimizer $x^\star$. Let $\{x^{(t)}\}$ be a sequence of generated by the incremental subgradient iteration (5.10) with a constant stepsize $\alpha^{(t)} = \alpha$. Then

$$\min_{0 \leq k \leq t} f_0(x_0^{(k)}) - f_0^\star \leq \frac{\|x^\star - x_0^{(0)}\|_2^2 + t\alpha^2 L^2}{2\alpha t}$$

where $L = \sum_i L_i$.

Analysis for diminishing step-size rules can be found in, *e.g.* [45]. Several variations, including methods where a new ordering of the components are chosen at random at the beginning of each outer iterations, have also been proposed [45]. It turns out that the expected convergence rate is better for the randomized method than the deterministic ones that uses the same fixed update order throughout.

Example 5.2. To illustrate the subgradient and incremental subgradient methods, consider the problem of minimizing

$$f_0(x) = \sum_{k=1}^{n} k|x_k|$$

Since the f_0 has Lipschitz constant n, targeting an accuracy of ε gives a constant step-size of $\alpha = 2\varepsilon/n^2$. We consider $n = 4$ and $\varepsilon = 0.1$ which gives $\alpha = 1/80$. Figure 5.3 shows how the objective function evolves for fixed (dark color) and diminishing (light color) stepsize $\alpha^{(t)} = 1/t$. In both cases, the iterates oscillate around the optimum: for fixed stepsize, there is a sustained oscillation with fixed amplitude (below the target accuracy, as predicted by theory); for diminishing stepsize, the magnitude of the oscillations decreases (due to the decreasing stepsize of the divergent step-size rules) and the best function value found during the iterations will asymptotically converge to the optimum.

5.3 Decomposition Techniques

The basic idea of decomposition techniques is to exploit problem structure to divide a complex optimization problem into subproblems that are easier to solve. The subproblems are then coordinated towards the globally optimal solution by the (repeated) solution of a master problem, see Figure 5.1.

One can essentially trace two different motivations for using decomposition in the mathematical programming literature. One line of work is motivated by the need to solve very large-scale optimization problems, *e.g.* [15, 4, 37, 21, 6]. Another line

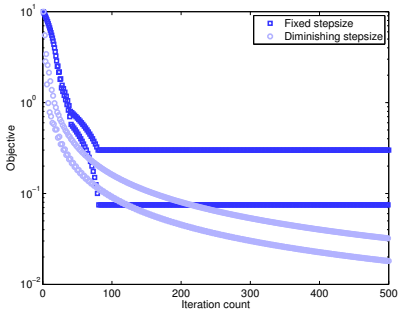

Fig. 5.3 Comparison of subgradient method for fixed and diminishing stepsizes

of work is motivated by decentralization of decision-making in organizations or engineering systems, *e.g.* [3, 65, 14, 25]. In the first line of work, subproblems are typically easier to solve simply because they are smaller (in terms of decision variables or constraints) or because they have a special structure that can be exploited. In the other line of work, it is not the computations that complicate the original problem but the fact that the overall problem combines variables (decisions) that belong to separate components in an underlying system. Implementation advantages are then obtained if we can find a problem decomposition that places the main computations within individual components and restricts the coordination overhead among these.

In this section, we will review the basic primal and dual decomposition techniques and some variations.

5.3.1 Dual Decomposition

From a large-scale optimization perspective, the basic idea with dual decomposition is to exploit the structure of the dual optimization problem to improve computational efficiency. The theoretical foundation for this is one of Lagrangean duality. Associated to the optimization problem (5.2) is the *Lagrangean*

$$L(x,\lambda,\mu) = f_0(x) + \sum_i \lambda_i f_i(x) + \sum_i \mu_i h_i(x)$$

Here, the constraints of the original problem have been relaxed and are accounted for by adding a weighted sum of the constraint functions to the objective. Note that if $\lambda \succeq 0$ and $x \in X$, then $L(x,\lambda,\mu) \leq f_0(x)$, and that the *dual function*

$$q(\lambda,\mu) = \inf_x L(x,\lambda,\mu) \tag{5.11}$$

is a lower bound to the optimal value of the original problem. It is hence natural to try to maximize this lower bound, which leads to the dual problem

$$\begin{aligned} &\text{maximize } q(\lambda,\mu) \\ &\text{subject to } \lambda \succeq 0 \end{aligned} \tag{5.12}$$

The fact that the optimal value of this program is always a lower bound to the original problem is called *weak duality*. For convex problems, which is the focus of this chapter, relatively mild conditions guarantee that the optimal value of the dual problem coincides with the optimal value of the (original) primal problem. We then say that *strong duality* holds. Conditions for strong duality are known as constraint qualification, and the most well-known result is due to Slater.

Proposition 5.5. *Consider the convex optimization problem (5.2). If there exists a strictly feasible point $\check{x}$ satisfying*

$$\begin{aligned} f_i(\check{x}) &< 0 \qquad i = 1,\dots,m \\ h_i(\check{x}) &= 0 \qquad i = 1,\dots,p \end{aligned}$$

then strong duality holds.

For alternative constraint qualification theorems and stronger version of Slater's conditions, see *e.g.* [5].

To solve the dual problem using a first-order method, we need a gradient or subgradient of the dual function at the current dual variables (λ,μ). To this end, the following result is useful.

Proposition 5.6. *The dual function (5.11) is concave in λ and μ, and a subgradient of $-q(\lambda,\mu)$ at (λ,μ) is given by*

$$-\Big(f_1(x^\star(\lambda,\mu)),\,\cdots,\,f_m(x^\star(\lambda,\mu)),\,h_1(x^\star(\lambda,\mu)),\,\cdots,\,h_p(x^\star(\lambda,\mu))\Big)$$

where $x^\star(\lambda,\mu) = \arg\inf_x L(x,\lambda,\mu)$. If $f_0(x)$ is strictly convex in all variables, then the dual function is continuously differentiable.

This result tells us that the dual problem is always convex (in fact, it is convex even if the original problem is not), and hence can be solved using gradient or subgradient techniques depending on if the original problem is strictly convex or not.

As we already mentioned, the basic idea of dual decomposition is to explore structure in the dual function to improve computational efficiency and/or ensure decentralization of decisions. The following two examples, taken from [9], illustrate the ideas.

Example 5.3 (Dual decomposition of coupling constraint). To illustrate the dual decomposition technique, consider the following problem

$$\begin{aligned} &\text{minimize } \varphi_1(x_1)+\varphi_2(x_2) \\ &\text{subject to } x_1+x_2 \le x_{\text{tot}} \end{aligned}$$

If it were not for the coupling constraint on the total resource, the problem would be separable and the optimal allocations $x_1^\star$ and $x_2^\star$ could easily be found. Introducing a dual variable λ for the total resource constraint, we find the Lagrangian

$$\begin{aligned} L(x,\lambda) &= \varphi_1(x_1) + \varphi_2(x_2) + \lambda(x_1 + x_2 - _{\text{tot}}) = \\ &= \varphi_1(x_1) + \lambda x_1 + \varphi_2(x_2) + \lambda x_2 - \lambda x_{\text{tot}} \end{aligned}$$

which is separable in x_1 and x_2. Hence, so is the dual

$$\begin{aligned} q(\lambda) = \inf_x L(x,\lambda) &= \underbrace{\inf_{x_1}\{\varphi_1(x_1) + \lambda x_1\}}_{q_1(\lambda)} + \underbrace{\inf_{x_2}\{\varphi_2(x_2) + \lambda x_2\}}_{q_2(\lambda)} - \lambda x_{\text{tot}} \\ &= q_1(\lambda) + q_2(\lambda) - \lambda x_{\text{tot}} \end{aligned}$$

The dual function can thus be evaluated in parallel, by letting one decision-maker find the x_1 that minimizes $\varphi_1(x_1) + \lambda x_1$ and another decision-maker find the x_2 that minimizes $\varphi_2(x_2) + \lambda x_2$, and then compute the sum above. Coordination of the two decision-makers is done by adjusting λ to maximize $g(\lambda)$ (*i.e.* to solve the dual problem), *e.g.* using the subgradient iteration

$$\lambda^{(t+1)} = \max\left\{\lambda^{(t)} + \alpha^{(t)}\left(x_{\text{tot}} - x_1^\star(\lambda^{(t)}) - x_2^\star(\lambda^{(t)})\right), 0\right\}$$

Convergence of the Lagrange multipliers (and hence of the dual function) is guaranteed using the classical results on projected gradient and subgradient iterations.

The above example also illustrates why dual decomposition is sometimes referred to as *price-directive decomposition*. Interpreting the dual variable as the unit price for the common resource, the system is directed towards its optimal operation by appropriate pricing of the common resource. Constraints on the common resource are not explicitly enforced, but the demand is asymptotically aligned with the supply using a simple pricing strategy: increase the prices if the resource is in shortage and decrease the price if the resource is in excess. To exercise this pricing interpretation further, and to be able to make a link to the game-theoretic methods descried later in this chapter, we consider the following application to rate allocation in communication networks [30].

Example 5.4. Consider a communication system where N flows are routed through the same bottleneck link with capacity c. To find how to optimally allocate the available bandwidth to the flows, consider the following optimization problem

$$\begin{aligned} &\text{maximize}_x \ \textstyle\sum_i u_i(x_i) \\ &\text{subject to} \ \ \textstyle\sum_i x_i \le c \qquad\qquad , \\ &\qquad\qquad x_i \ge 0,\ i = 1,\cdots,N \end{aligned} \tag{5.13}$$

Here, x_i is the rate allocated to flow i and the constraints encode that rates are positive and the total communication rate cannot exceed capacity. Associated to each flow i is a concave, strictly increasing, and continuously differentiable utility function with domain $x_i \geq 0$.. The utility function $u_i(\cdot)$ is used to represent the degree of satisfaction when user i is allocated rate x_i. Concavity here corresponds to the assumption of *elastic* traffic. Note that (5.13) is a convex optimization problem that can be put into our standard form by replacing maximize $\sum_i u_i(x_i)$ with minimize $\sum_i -u_i(x_i)$ and introducing $f_i(x_i) = -u_i(x_i)$. Hence, the problem (5.13) can be solved by dual decomposition with Lagrange multiplier λ partial Lagrangian

$$L(x,\lambda) = \left\{ \sum_i -u_i(x_i) + \lambda \left(\sum_i x_i - c \right) \;\middle|\; x_i \geq 0,\ i = 1,\ldots,N \right\}.$$

In the dual decomposition framework, each user is charged a common price λ per unit flow and sets its rate to maximize $u_i(x_i) - \lambda x_i$. One could also imagine alternative mechanisms for allocating the capacity. One such mechanism is biding: each user i submits a bid $b_i \geq 0$ of the amount of money that the user is willing to pay to get a share of the bandwidth. Each user is charged the same price μ per unit flow, leading to a rate allocation of $x_i = b_i/\mu$. In this case, given a price $\mu \geq 0$, user i acts to maximize the following payoff function over $b_i \geq 0$:

$$w_i(b_i,\mu) = u_i\left(\frac{b_i}{\mu}\right) - b_i. \tag{5.14}$$

It was shown in [30] that if the link manager sets price to "clear the market", *i.e.*

$$\mu = \frac{\sum_i b_i}{c}. \tag{5.15}$$

there is a pair (b,μ) solving $(5.14)-(5.15)$, which also solves (5.13). In the above situation, user is assumed to be a price taker, *i.e.* the payoff function w_i (5.14) takes the price μ as a fixed parameter. Price-anticipating users will realize that μ is set according to (5.15). This makes the model a game between N users, which will be discussed more detailed in Section 5.5.2.3

The next example illustrates how dual decomposition can be used when variables couple the objective functions in an otherwise decoupled optimization problem.

Example 5.5 (Dual decomposition of coupling variable). The dual decomposition technique can also be used to decouple problems in which the same decision variable appears in several objective functions. One of the simplest instances is

$$\text{minimize } \varphi_1(x) + \varphi_2(x)$$

To decouple the problem, introduce x_1 and x_2 to reflect the two decision-makers' respective view of the optimal variable x, *i.e.* consider the equivalent problem

$$\begin{aligned} &\text{minimize } \varphi_1(x_1)+\varphi_2(x_1) \\ &\text{subject to } x_1 = x_2 \end{aligned}$$

The problem now has the same form as the previous example, and the same simple steps yield an algorithm where decision-makers optimize there individual decision variables, and the optimal value is found by adjusting the "consistency price" μ.

A drawback with dual decomposition is that a solution to the dual problem (5.12), even under strong duality, only provides the optimal value of the primal problem (5.2) but not necessarily the optimal primal decision variables. For non-optimal values of λ and μ, the primal iterates $x^\star(\lambda,\mu) = \arg\inf_x L(x,\lambda,\mu)$ are typically not even feasible to the original problem. Hence, if constraint violations cannot be tolerated, the dual decomposition method needs to be complemented by a method for recovering primal feasible solutions. When the dual function is differentiable, the primal iterates will converge to their optimal values as the dual variables approach optimality. However, as illustrated next, the primal iterates typically do not converge when the dual function is non-smooth.

Example 5.6. Consider the following specific instance of the problem in Example 5.3

$$\begin{aligned} &\text{minimize } 2|x_1-2|+4|x_2-4| \\ &\text{subject to } x_1+x_2 = 5 \\ &\qquad\qquad 0 \le x_1 \le 10,\ 0 \le x_2 \le 10 \end{aligned}$$

Clearly, the optimal solution is $x_1^\star = 1, x_2^\star = 4$ with optimal value $f^\star = 2$. The full light blue lines in Figure 5.2 (left) show the evolution of the dual function for the step-size rule $\alpha(t) = 1/(1+t)$. Note that the dual function is a lower bound to the optimal value and that the dual problem converges to the optimal value. However, as revealed in Figure 5.2(right), the primal variables do not converge but oscillate, in this case, between their minimal value and unconstrained optimum.

For comparison, we change the objective function to

$$2(x_1-2)^2+4(x_2-4)^2$$

which is smooth and strictly convex. The optimal solution is now $x_1^\star = 4/3, x_2^\star = 11/3$ and the optimal value is $f^\star = 4/3$. The dashed lines in Figure 5.2(left) show how the dual objective converges. In contrast to the non-smooth example, the iterates now also converge to their optimal values, see Figure 5.2(right).

Techniques for recovering primal optimal solutions are developed and reviewed in [35, 46]. Since differentiability of the dual function gives several advantages, many techniques have been proposed to ensure dual differentiability. A simple approach is to regularize the objective function, *e.g.* to replace $f_0(x)$ by $f_0(x)+\varepsilon\|x\|_2^2$ for some small positive constant. A more elegant approach is to use proximal optimization or augmented Lagrangian techniques described next.

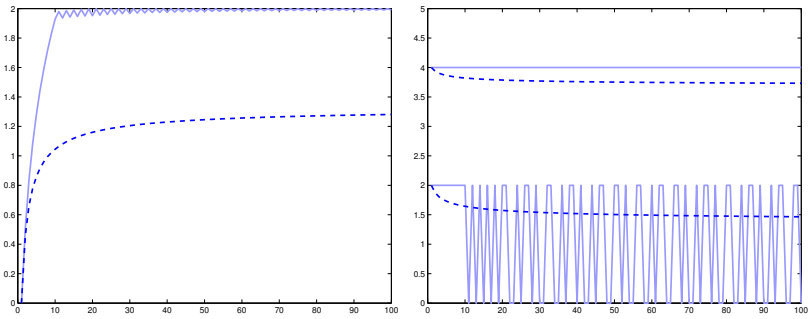

Fig. 5.4 When the dual function is non-smooth, the dual objective (left, full lines) converges while the primal iterates (right, full lines) do not. When the dual function is differentiable, both objective and iterates converge asymptotically (dashed lines)

5.3.2 Augmented Lagrangian and Proximal Point Methods

.

The basic idea of the proximal point method is to ensure strict convexity by introducing an artificial variable y and re-write the original problem (5.1) as

$$\begin{aligned} &\text{minimize } f_0(x) + \frac{1}{2c}\|x-y\|_2^2 \\ &\text{subject to } x \in X,\ y \in \mathbb{R}^n \end{aligned} \tag{5.16}$$

for some positive constant $c > 0$. This problem can be considered as one in y only:

$$\begin{aligned} &\text{minimize } f_c(y) \\ &\text{where} \quad f_c(y) = \inf_{x\in X} f_0(x) + \frac{1}{2c}\|y-x\|_2^2 \end{aligned} \tag{5.17}$$

Now, $f_c(y)$ is continuously differentiable with

$$\nabla f_c(y) = \frac{1}{c}\,(y - x^\star(y))$$

where

$$x^\star(y) = \arg\inf_{x\in X} f_0(x) + \frac{1}{2c}\|y-x\|_2^2$$

so the problem is readily solved using the gradient iteration

$$y^{(t+1)} = y^{(t)} - c\nabla f_c(y^{(t)}) = x^\star(y^{(t)})$$

A related technique is the method of augmented Lagrangians, which is most readily explained on convex problems with equality constraints

$$\begin{aligned} &\text{minimize } f(x) \\ &\text{subject to } Ax = b \end{aligned} \tag{5.18}$$

We re-write the problem as

$$\begin{aligned} &\text{minimize } f(x) + \tfrac{1}{2c}\|Ax-b\|_2^2 \\ &\text{subject to } Ax = b \end{aligned} \tag{5.19}$$

introduce multipliers μ for the constraints and form the *augmented Lagrangian*

$$L_a(x,\mu) = f(x) + \mu^T(Ax-b) + \frac{1}{2c}\|Ax-b\|_2^2$$

Now, the method of multipliers is essentially dual decomposition applied to (5.19), *i.e.* we run the iterations

$$\begin{aligned} x^{(t+1)} &= \arg\min_x L_a(x,\mu^{(t)}) \\ \mu^{(t+1)} &= \mu^{(t)} + c(Ax^{(t+1)} - b) \end{aligned}$$

It is possible to show that these iterations are equivalent to what would have been derived using proximal minimization of the dual of (5.18), see *e.g.* [6, Ch. 3]. Next, we illustrate how the techniques work on simple problem from Example 5.5.

Example 5.7. Consider the problem from Example 5.5, introduce an auxillary variable y, and re-write the problem as

$$\begin{aligned} &\text{minimize } \varphi_1(x_1) + \varphi_2(x_2) \\ &\text{subject to } x_1 = y \\ &\qquad\qquad x_2 = y \end{aligned}$$

The augmented Lagrangian is thus

$$L_a(x,y,\mu) = \varphi_1(x_1) + \varphi_2(x_2) + \mu_1(x_1-y) + \mu_2(x_2-y) + \frac{1}{2c}\left((x_1-y)^2 + (x_2-y)^2\right)$$

and performing alternating minimization of the primal variables, we find

$$\begin{aligned} x_i^{(t+1)} &= \arg\min_z \varphi_i(z) + \mu_i^{(t)} z + \frac{1}{2c}(z-y)^2, \quad i = 1,2 \\ y^{(t+1)} &= \frac{x_1^{(t+1)} + x_2^{(t+1)}}{2} - c\frac{\mu_1^{(t)} + \mu_2^{(t)}}{2} \\ \mu_i^{(t+1)} &= \mu_i^{(t)} + (x_i^{(t+1)} - y^{(t+1)}), \quad i = 1,2 \end{aligned}$$

In fact, the iterations can be simplified by noting that at optimality, $\mu_1^\star + \mu_2^\star = 0$ and that if we initialize the multipliers such that $\mu_1^{(0)} + \mu_2^{(0)} = 0$, it will hold that $\mu_1^{(t)} + \mu_2^{(t)} = 0$ for all t. Thus, the simplified iterations can be written as

$$x_i^{(t+1)} = \arg\min_z \left\{ \varphi_i(z) + \mu_i^{(t)} z + \frac{1}{2c}(z - \bar{x}^{(t)})^2 \right\}, \quad i = 1,2$$
$$\mu_i^{(t+1)} = \mu_i^{(t)} + (x_i^{(t+1)} - \bar{x}^{(t+1)}), \quad i = 1,2$$

where $\bar{x}^{(t)} = (x_1^{(t)} + x_2^{(t)})/2$.

Although the proximal and augmented Lagrangian techniques are guaranteed to converge for all values of c, the choice of c influences both the numerical conditioning of the subproblems and the converge speed of the method, see [6].

5.3.3 Primal Decomposition

Primal decomposition is also called *resource-directive* decomposition. Rather than introducing a pricing scheme for the common resources, the primal decomposition approach sequentially updates the resource allocation to minimize the global system objective. Contrary to dual decomposition, the iterates generated by primal decomposition techniques are always feasible (by construction) and converge asymptotically to their optimal values.

The theory for primal decomposition is built around the concept of primal function of an optimization problem. The *primal function* $p(u)$ of the problem (5.2) is the optimal value of the perturbed problem

$$\begin{aligned} &\text{minimize } f_0(x) \\ &\text{subject to } f_i(x) \le u_i,\ i = 1,\dots,m \\ &\qquad\qquad h_i(x) = 0,\ \ i = 1,\dots,p \end{aligned} \tag{5.20}$$

The domain of p is the set of perturbations u for which there is a feasible primal solution x and, hence, $p(u) < \infty$. We denote the domain of p by P:

$$P = \{u \mid p(u) < \infty\}$$

Contrary to the dual function (which is always concave), convexity of the primal function and its domain requires convexity of the original problem:

Proposition 5.7. *Consider the convex optimization problem (5.2) and assume that $p(u) > -\infty$ for all $u \in P$. Then P is a convex set and $p(u)$ is convex over P.*

In order to minimize $p(u)$ we also need to be able to compute (at least) a subgradient. The following characterization is then useful.

Proposition 5.8. *Consider the convex optimization problem (5.2) with optimal dual variables* $(\lambda^\star, \mu^\star)$. *Then*

$$p(0) \leq p(u) + u^T \lambda^\star$$

i.e., $-\lambda^\star$ *is a subgradient of* p *at* $u = 0$.

Next, we demonstrate how primal decomposition can be applied to Example 5.3.

Example 5.8. The key trick in applying primal decomposition to a problem with a complicating constraint such as Example 5.3 is to introduce variables r_i that represent the amount of the common resource allocated to subproblem i and re-write it as

$$\begin{aligned} &\text{minimize } \varphi_1(x_1) + \varphi_2(x_2) \\ &\text{subject to } x_1 \leq r_1, \quad x_2 \leq r_2 \\ &\qquad r_1 + r_2 \leq x_{\text{tot}} \end{aligned}$$

For notational simplicity, introduce the constant $r_{\text{tot}} = x_{\text{tot}}$ and define functions $v_i(r_i) = \inf\{\varphi_i(x_i) \,|\, x_i \leq r_i\}$. The problem can then be equivalently written as

$$\begin{aligned} &\text{minimize } \textstyle\sum_i v_i(r_i) \\ &\text{subject to } \textstyle\sum_i r_i \leq r_{\text{tot}} \end{aligned} \tag{5.21}$$

Now, from the results on the primal function, v_i is a convex, possibly non-smooth function and a subgradient of v_i at r_i is given by the optimal dual variable $\lambda_i^\star(r_i)$ associated with the inequality constraint of the primal minimization problem

$$\begin{aligned} &\text{minimize } \varphi_i(x_i) \\ &\text{subject to } x_i \leq r_i \end{aligned}$$

Hence, (5.21) is a convex optimization problem that can be solved using a projected subgradient method

$$r^{(t+1)} = P_R\{r^{(t)} + \alpha^{(t)} \lambda^\star(r^{(t)})\}$$

where $r^{(t)} = (r_1^{(t)}, r_2^{(t)})$, $\lambda^\star(r^{(t)}) = (\lambda_1^\star(r_1^{(t)}), \lambda_2^\star(r_2^{(t)}))$ and $P_R\{\cdot\}$ denotes projection onto the total budget constraint $r_1 + r_2 \leq r_{\text{tot}}$.

It is useful to compare the solution mechanisms suggested by primal and dual decomposition. In primal decomposition, the master problem assigns resources to the subsystems. The subsystems then optimize their operation to "perform their best" with the amount of resources they have been assigned (compute the optimal x_i and the associated value of $v_i(r_i)$ above), and return a Lagrange multiplier vector back

to the coordinator. It is well-known from sensitivity theory that the Lagrange multipliers indicate the potential cost reduction that can be achieved by an additional amount of resource. In our case, if we assume that φ_i are (in addition to convex) smooth and decreasing, and r_i are scalar variables, then the first-order optimality conditions yield that $\lambda_i(r_i) = -f'(r_i)$. In other words, the Lagrange multipliers signal exactly the amount of cost reduction that a subsystem could generate if it would obtain a small additional amount of resource. When the master problem updates the resource allocation via the projected (sub)gradient iteration, it effectively shifts resources from subsystems with small predicted cost reduction to subsystems with large predicted cost reduction. The local decisions (iterates x_i) taken by nodes are, by construction, feasible to the original problem.

In the dual decomposition method, on the other hand, the coordinator announces a dual variable λ which, by similar reasoning as above, can be interpreted as the unit cost of the common resource. Subsystems then optimize their operation accounting both for their operational cost ($\varphi_i(x_i)$) and the resource cost (λx_i). The coordinator then updates the price to align supply and demand: increase the price if the resource is overbooked and decrease the price otherwise. The iterates are, in general, not guaranteed to be feasible to the original problem.

5.4 Networked Optimization

With a basic understanding of first-order methods and decomposition techniques, we are ready to consider *networked optimization* problems. Contrary to decomposition techniques, in which the problem structure determines how the original problem is divided into subproblems, networked optimization problems arise when the communication structure (which decision-makers are allowed to coordinate with each other) is fixed and the computation structure has to be tailored to match. This class of optimization algorithms have also been termed *multi-user* optimization, since the set-up can be used to model a multi-agent system in which agents cooperate and exchange information with neighbors to find a globally optimal decision [33]. Although decomposition techniques will turn out to be useful also in this context, decomposition does not necessarily yield distributed optimization algorithms unless the master- and sub-problems can be distributed.

To study networked optimization, we consider problems on the form

$$\begin{aligned} &\text{minimize } \textstyle\sum_{v\in\mathscr{V}} f_v(x_v,\theta) \\ &\text{subject to } x_v \in X_v,\ \theta \in \Theta \end{aligned} \tag{5.22}$$

with an associated *communication graph* $\mathscr{G} = (\mathscr{V},\mathscr{E})$; see Figure 5.1. The vertices of the graph represent decision-makers, and the edges encode which agents can exchange information to coordinate their decisions. Each node has a set of local (private) decision variables x_v. These variables are under exclusive control of node

v and only influences the local loss function $f_v(x_v, \theta)$. The global variables θ, on the other hand, impact the loss functions of multiple nodes. We focus on convex problems and assume that the constraint sets X_v and Θ are non-empty, closed and convex and that the local loss functions $f_v(x_v, \theta)$ are convex in (x_v, θ).

We can eliminate the local variables by introducing new loss functions

$$\nu_v(\theta) = \inf_{x_v \in X_v} f_v(x_v, \theta)$$

and reformulate (5.22) as

$$\begin{aligned} &\text{minimize } \textstyle\sum_{v \in \mathscr{V}} \nu_v(\theta) \\ &\text{subject to } \theta \in \Theta \end{aligned} \tag{5.23}$$

Clearly, under our assumptions $\nu_v : \mathbb{R}^n \mapsto \mathbb{R}$ are convex but possibly non-smooth.

5.4.1 Networked Optimization Via Dual Decomposition

The most direct approach to networked optimization is dual decomposition. First, introduce local decision variables θ_v at each node $v \in \mathscr{V}$ and rewrite (5.23) as

$$\begin{aligned} &\text{minimize } \textstyle\sum_{v \in \mathscr{V}} \nu_v(\theta_v) \\ &\text{subject to } \theta_v = \theta_w \qquad \forall (v,w) \in E \\ &\qquad\qquad \theta_v \in \Theta \qquad \forall v \in V \end{aligned} \tag{5.24}$$

The problem would be separable if it were not for the edge-wise coupling constraints. It is thus natural to relax these constraints by dual decomposition. To this end, introduce multipliers $\mu_{(v,w)}$ for all $(v,w) \in \mathscr{E}$ and form the partial Lagrangean

$$L(\theta, \mu) = \{\sum_v \nu_v(\theta_v) + \mu_{(v,w)}(\theta_v - \theta_w) \mid \theta_v \in \Theta\}$$

with associated dual function

$$g(\mu) = \inf_{\theta_v \in \Theta_v} \sum_v \nu_v(\theta_v) + \theta_v \sum_w (\mu_{(v,w)} - \mu_{(w,v)})$$

Since the problem is equality-constrained, the dual problem is unconstrained and can be solved using a standard subgradient optimization

$$\mu_{(v,w)}^{(t)} = \mu_{(v,w)}^{(t)} + \alpha^{(t)}(\theta_v^\star(\mu^{(t)}) - \theta_w^\star(\mu^{(t)})) \tag{5.25}$$

where $\theta^\star(\mu^{(t)}) = \arg\inf_\theta L(\theta, \mu^{(t)})$. The following example illustrates how the technique applies to a networked least-squares problem.

Example 5.9. Consider a networked least-squares problem, where

$$f_v(x_v, \theta) = \frac{1}{2}(\theta - z_v)^2$$

Introducing local variables θ_v and following the procedure outlined above, we find

$$v_v(\theta_v) = \frac{1}{2}(\theta_v - z_v)^2$$

In this case, it is also possible to find an explicit expression for $\theta_v^\star(\mu^{(t)})$,

$$\theta_v^\star(\mu^{(t)}) = z_v - \sum_w (\mu_{(v,w)}^{(t)} - \mu_{(w,v)}^{(t)})$$

This expression, together with the iteration (5.25), defines a networked optimization algorithm that is guaranteed to find the optimal solution, provided that the step-length sequence $\{\alpha^{(t)}\}$ is chosen appropriately.

To implement the algorithm, note that each link needs information about $\theta_v^\star$ and $\theta_w^\star$, *i.e.* the current version of the local decision variables of the two nodes connected to the link. To compute these decisions, on the other hand, nodes need to know the Lagrange multipliers of all links that they are connected to.

Since the optimization problem is strictly convex, the dual function is smooth we can use a higher-order method to accelerate convergence. We omit the details here but evaluate both the direct and the accelerated method to a ring network of $N = 100$ nodes. Figure 5.5 shows how the methods converge, with a significant speed-up advantage of the accelerated method.

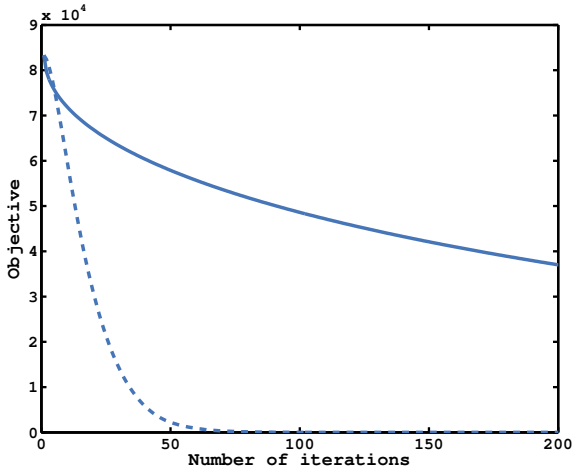

Fig. 5.5 Distributed least-squares: gradient (full) and accelerated gradient (dashed)

While dual decomposition is classical, the first applications to networked optimization (and estimation) in the spirit above the authors are aware of are the work by Fawal, Georges and Bornard [18] and the one by Rabbat and Nowak [54] (who also analyze convergence of the method for time-varying graphs). For ease of exposition, we have introduced one Lagrange multiplier for every edge in the graph, but it is possible to only introduce dual variables for a subset of links (as long as the underlying graph is connected) [60]. Finally, it is also possible to apply alternative decomposition techniques, such as the augmented Lagrangian method, to networked optimization [18, 60].

5.4.2 Consensus-subgradient Schemes

In the dual decomposition approach, asymptotic agreement on the global decision variable is enforced by adjusting Lagrange multipliers. However, agreement among nodes in a network can be achieved by many different techniques [50, 52, 68], indicating that there could be a rich family of algorithms for networked optimization. We will consider one such class of algorithms which combines subgradient optimization and consensus algorithms. Although consensus algorithms are described in depth in other chapters in this book, we will give a brief overview for completeness.

The consensus problem considers the design of protocols that ensure that all nodes in a network agree on a common quantity. In the area of multi-agent system, a lot of attention has been focused on conditions where linear iteration on the form

$$x_v(t+1) = x_v(t) - \sum_{w:(v,w)\in\mathscr{E}} W_{(vw)}(x_v - x_w) \tag{5.26}$$

converge so that all node variables equal the average of the initial values, *i.e.*

$$\lim_{t\to\infty} x_v(t) \to \frac{1}{|\mathscr{V}|}\sum_v x_v(0)$$

It is convenient to write the iterations in matrix form:

$$x(t+1) = Wx(t), \tag{5.27}$$

where the i-th element of the vector $x(t)$ corresponds to the local value at node i, $x_i(t)$. We make the following assumptions on W.

Assumption 5.1 (Consensus Matrix Properties) *The weight matrix W satisfies*

$$
\begin{aligned}
&i) \quad [W]_{ij} = 0, \text{ if } (i,j) \notin \mathscr{E} \text{ and } i \neq j, \\
&ii) \quad W\mathbf{1}_N = \mathbf{1}_N, \rho\left(W - \frac{\mathbf{1}_N\mathbf{1}_N^T}{N}\right) < 1, \\
&iii) \quad W = W^T,
\end{aligned}
$$

where $\rho(\cdot)$ is the spectral radius and $\mathbf{1}_N \in \mathbb{R}^N$ is the column vector of all ones.

The first assumption restricts nodes to only communicate with their immediate neighbors, while *iii)* encodes the assumption that the underlying graph is undirected (or that all links are bidirectional) and that the same weight is used in the consensus iterations for both directions of the same link. Finally, the second assumption ensures asymptotic average consensus:

Lemma 5.1. *If the weight matrix $W \in \mathbb{R}^{N\times N}$ is symmetric, then the limit*

$$\lim_{k\to\infty} W^k = \frac{\mathbf{1}_N\mathbf{1}_N^T}{N} \tag{5.28}$$

holds if and only if Assumption 5.1 ii) holds for W.

Proof. See, *e.g.*, [68, Theorem 1]. ∎

There are many ways of selecting W to satisfy the above conditions using either centralized or decentralized information, see *e.g.* the chapter by Garin and Schenato in this book or references [50, 8]. We will only present one example, namely the Metropolis-Hastings scheme [23, 8]:

Lemma 5.2 (Metropolis-Hastings). *If the graph $\mathscr{G}$ is connected, then W fulfills Assumption 5.1 if the elements of W are set to*

$$[W]_{vw} = \begin{cases} \min\{d_v^{-1}, d_w^{-1}\} & \text{if } (v,w) \in \mathscr{E} \text{ and } v \neq w \\ \sum_{(v,w)\in\mathscr{E}} \max\{0, d_v^{-1} - d_w^{-1}\} & \text{if } v = w \\ 0 & \text{otherwise}. \end{cases} \tag{5.29}$$

where d_v denotes the degree (number of neighbors) of node v.

Note that the iteration (5.26) is simply a method for distributed computation of a network-wide average. To make use of the algorithm for networked optimization, we must first compute the local quantity that should be averaged across nodes and then execute one or more iterations of the consensus algorithm. As an example, consider the consensus-subgradient method from [44] and further developed in [27]. The basic idea of this algorithm is to interpret the gradient of the objective function

$$\frac{\partial}{\partial\theta} v(\theta) = N\frac{1}{N}\sum_v v_v'(\theta)$$

as (N times) the average of the gradients of the individual node objective functions. Hence, for smooth objectives with $\Theta = \mathbb{R}^n$ it is natural to consider the algorithm

$$\theta_v^{(t+1)} = \sum_{w \in \mathscr{V}} [W^\varphi]_{vw} \left(\theta_w^{(t)} - \alpha^{(t)} \nu_w'(\theta_w^{(t)}) \right)$$

Here, W^φ denotes that the consensus iteration is performed φ times. Hence, every node v maintains a vector of θ_v of local variables, and computes a desired next iterate by accounting only for the local gradient. Nodes then perform consensus iterations to agree on the next iterate and execute this one. Clearly, if all nodes start with the equal initial values, $\theta_v(0) = \theta_w(0)$ for all $(v,w) \in \mathscr{E}$ and we let $\varphi \to \infty$ we recover the classical gradient iteration. More surprisingly, the method still works when nodes start with different initial values and perform a finite number of iterations, even when the objective functions are not differentiable and θ is subject to (convex) constraints. The constrained non-smooth version of algorithm reads

$$\theta_v^{(t+1)} = P_\Theta \left[\sum_w [W^\varphi]_{vw} \left(\theta_w^{(t)} - \alpha^{(t)} g_w(\theta_w^{(t)}) \right) \right], \tag{5.30}$$

where $\theta_v^{(0)} \in \Theta$ for all $v \in \mathscr{V}$, $g_v(\theta_v^{(t)}) \in \partial f_v(\theta_v^{(t)})$, $\varphi \in \mathbb{N}$, and $[W^\varphi]_{vw}$ denotes the element of W^φ in the v-th row and w-th column. Hence, using this scheme, agents maintain their local variable vector $\theta_v^{(t)}$ and compute a desired next iterate $\theta_v^{(t)} - \alpha^{(t)} g_v(\theta_w^{(t)})$. They then announce this desired iterate to their neighbors and update their own decision taking the neighbors desires into account. More specifically, (5.30) implies that each agent runs φ number of consensus iterations with its neighbors defined by (5.27) to determine its next iterate. The total number of iterations in the algorithm up to step t is therefore $t\varphi$. To give the flavor of the method, we give the following result

Theorem 5.2. *Consider Problem (5.23) with $\Theta = \mathbb{R}^n$ and assume that $\|g_v\|_2 \le L$ for all $g_v \in \partial \nu_v$ and all $v \in \mathscr{V}$. Let $\{\tilde{\theta}_v(t)\}_{t=0}^\infty$ be the Cesàro averages*

$$\tilde{\theta}_v^{(t)} = \frac{\sum_{k=0}^t \alpha^{(k)} \theta_v^{(k)}}{\sum_{k=0}^t \alpha^{(k)}}$$

of the iterates $\theta_v^{(t)}$ produced by (5.30) with a consensus matrix satisfying Assumption 5.1. Define

$$\gamma = \rho \left(W - \frac{\mathbf{1}\mathbf{1}^T}{N} \right)$$

and $\beta^{(0)}$ such that $\|\theta_v^{(0)} - |\mathscr{V}|^{-1} \sum_{w \in \mathscr{V}} \theta_w^{(0)}\| \le \beta^{(0)}$. Then, under diminishing stepsizes satisfying $\alpha^{(t+1)}/\alpha^{(t)} > \mu$, if

$$\varphi \ge \frac{\log(\mu\beta^{(0)}) - \log(4\sqrt{|\mathscr{V}|}n(\beta^{(0)} + \alpha^{(0)}L^2)}{\log(\gamma)}$$

we have that

$$\lim_{t\to\infty} v(\tilde{\theta}_v^{(t)}) = v^\star, \quad \forall v \in \mathcal{V}.$$

More extensive analysis results for several variants of the method can be found in [27].

5.4.3 Networked Incremental Subgradient Methods

The use of classical incremental subgradient methods in a networked setting would suggest a message-passing solution where in each iteration, one node receive a token that contains the most recent iterate from another node in the network, computes the next iterate by taking a step in the negative subgradient of its own local objective function, and then passes the token to another node. A drawback with the incremental subgradient method is that the analysis assumes that the token is passed around in a ring (*i.e.* it performs cyclic rounds in which each component is updated once).

An alternative approach, tailored to the networked setting, was proposed and analyzed in [28]. Here, nodes pass their updated iterate to a *random neighbor* and the need for organizing the nodes into an underlying logical ring is removed. The algorithm is shown to converge as long as each component of the objective function is updated with equal probability. In other words, the token should perform an unbiased random walk on the graph. To design such a random walk, consider a Markov chain in which each state is associated to a node in the underlying network. To ensure that the token is passed only between neighboring nodes, the transition matrix W must fulfill the sparsity constraint that $W_{vw} = 0$ if $(v,w) \notin \mathcal{E}$. Furthermore, the analysis requires that the Markov chain is irreducible, aperiodic, and its stationary distribution is uniform. It turns out that matrices W constructed using Lemma 5.2 satisfy these assumptions. Now, the algorithm takes the form

$$\theta^{(t+1)} = P_\Theta\left\{\theta^{(t)} - \alpha g_w\right\}, \tag{5.31}$$

where $w^{(t)}$ is the state of Markov Chain in iteration t, α is a fixed step-length parameter and $g_w \in \partial f_w(\theta_w^{(t)})$. The following result establishes its convergence

Proposition 5.9. *Let* $\{\theta^{(t)}\}_{t=0}^\infty$ *be generated by* (5.31)*. With probability 1*

$$\begin{cases} \liminf_{t\to\infty} v\left(\theta^{(t)}\right) = v^\star, & \text{if } f^\star = -\infty \\ \liminf_{t\to\infty} v\left(\theta^{(t)}\right) \leq v^\star + \frac{\alpha L^2 K}{2}, & \text{if } v^\star > -\infty. \end{cases}$$

where K is an upper bound on the second moment of the recurrence time for all states in the Markov Chain.

Several variations and extensions of the basic method are given in [28], including stronger results (lim sup rather than lim inf) for the running average of the iterate that

a specific node sees during the course of the algorithm. The same paper also compares the predicted convergence rates for the networked incremental subgradient method with the provable convergence for the classical (deterministic or randomized) incremental subgradient method on several classes of graphs. Extensions to time-varying graph topologies and diminishing step-sizes can be found in [55].

5.5 Game Theory in Distributed Optimization

Game theory is typically used to model and counteract selfish behaviors in distributed systems. However, there are several reasons for us to introduce game theory in this chapter. First, networked optimization usually consists of multiple agents who can observe and react to their environment. Game theory offers a powerful tool set to analyze interactions between such intelligent entities. Second, although network components or agents would like to cooperate, it might be impractical or impossible to exchange the information required to implement any of the distributed optimization techniques described so far. It might then be better for agents to optimize their local or private objective and react to limited network information. In these cases, we use non-cooperation to capture limited information. The third reason is that game theory provides a way to predict, analyze or even to improve the outcome of a non-cooperative interaction, *e.g.* the notation of equilibrium.

5.5.1 Basics of Game Theory

We will mainly focus on the discussion of non-cooperative game model with complete information[1] and finite number of players. Formally, a normal or strategic form of a game Ξ is given by $\Xi = \left\{\mathscr{V}, \{\mathscr{S}_v\}_{v\in\mathscr{V}}, \{w_v\}_{v\in\mathscr{V}}\right\}$, where $\mathscr{V}$ is the set of agents or players; $\mathscr{S}_v$ is the set of strategies[2] of player v and w_v is its payoff function[3], which is a function of the strategy x_v chosen by player v and the strategies chosen by other players, denoted as x_{-v}. We will use $\mathscr{S} = \Pi_{v=1}^{|\mathscr{V}|}\mathscr{S}_v$ to denote the Cartesian product of sets and use $x = [x_v, x_{-v}] = \left[x_1, x_2, \cdots, x_{|\mathscr{V}|}\right]$, $\forall v \in \mathscr{V}$ to refer to a vector. Based on this model, the Nash equilibrium steady-state of a game can be defined. The definition is based on the concept of a best response correspondence.

Definition 5.2. Let $\Xi = \left\{\mathscr{V}, \{\mathscr{S}_v\}_{v\in\mathscr{V}}, \{w_v\}_{v\in\mathscr{V}}\right\}$ be a strategic game. For any $x_{-v} \in \mathscr{S}_{-v}$ we define the best response correspondence $BR_v(x_{-v})$ as,

[1] The complete information means that every player knows the payoff of the others.

[2] When a game is presented in normal form or strategic form, it is presumed that each player acts simultaneously or, at least, without knowing the actions of the other. If players have some information about the choices of other players, the game is usually presented in extensive form.

[3] Hereafter, we consider each player chooses strategy to maximize its payoff. The game can be defined accordingly if each play chooses strategies to minimize its own cost function.

$$BR_v(x_{-v}) = \left\{x_v \in \mathscr{S}_v | w_v(x_v, x_{-v}) \geq w_v\left(x'_v, x_{-v}\right) \text{ for all } x'_v \in \mathscr{S}_v\right\}.$$

A strategy profile x^* is a *Nash equilibrium* if and only if $x_v^* \in BR_v\left(x_{-v}^*\right)$ for all $v \in \mathscr{V}$.

The above definition is referred to pure Nash equilibrium. As seen in the above definition, the pure Nash equilibrium consists of selecting one element from each player's possible action sets and is also stable for each player to deviate from the equilibrium. A mixed strategy consists of selecting multiple elements from each player's action set and run the series of actions with some probability. The associated equilibrium is then called a mixed Nash equilibrium. In the following we will assume the strategic game Ξ is static, *i.e.* it is played in one-shot with complete information of payoffs, with a finite number of players and finite strategy sets. As shown later, although best response and gradient methods are usually adopted to update players' strategies to optimize the *current* payoff in reaction to other players' strategies, the game is still called a static game. If players choose their strategies to maximize their payoff functions averaged over the whole game duration, the game is called a repeated game.

5.5.2 Properties of Nash Equilibria

5.5.2.1 Existence of Nash Equilibria

The strategy tuples corresponding to Nash equilibria are consistent predictions of the outcome of a game. The first question after defining a game is whether there exists an equilibrium. Generally, proving the existence of an equilibrium involves proving the existence of a solution to a fixed-point problem [7]. However, a number of sufficient conditions for the existence of Nash equilibria have been developed for games with particular structure of strategy sets and payoff functions.

Theorem 5.3 (Debreu, Glicksberg, Fan [20]). *Let $\Xi = \left\{\mathscr{V}, \{\mathscr{S}_v\}_{v\in\mathscr{V}}, \{w_v\}_{v\in\mathscr{V}}\right\}$ be a strategic game, where $\mathscr{V}$ is a finite set. If $\forall v \in \mathscr{V}$, $\mathscr{S}_v$ is a non-empty compact and convex subset of a finite-dimensional Euclidean space; $w_v(x)$ is a continuous function in the profile of strategies x and quasi-concave in x_v; then the game has at least one pure Nash equilibrium.*

The power control game in [22] has been shown to be a quasi-concave game with a compact convex strategy set and hence has at least one pure Nash equilibrium. The multiple access game in [71] has a concave payoff function, which is a special case of the above theorem and thus it has a pure Nash equilibrium.

If a game does not have quasi-concave payoffs, one may turn to supermodular games [66] and potential games [43] to argue about the existence of Nash equilibria. Let us first look at the definition of supermodular game.

Definition 5.3 ([66]). The strategic form game Ξ is called supermodular if: $\forall v \in \mathscr{V}$, $\mathscr{S}_v$ is a compact subset of $\mathbb{R}$; w_v is upper semi-continuous in x; $\forall v \in \mathscr{V}$, $\forall x_{-v} \succeq x'_{-v}$ the quantity $w_v(x) - w_v\left(x_v, x'_{-v}\right)$ is non-decreasing in x_v.

Intuitively, the definition means that the marginal payoff of increasing a player's strategy rises with increases in the other players' strategies. This implies that the best response of a player is a non-decreasing function of other players' strategies. Supermodular games are interesting for several reasons. First, many applied models satisfy the assumptions of supermodular games. Second, they have the remarkable property that many solution concepts yield the same predictions. Finally, they tend to be analytically appealing – they have nice comparative statical properties and behave well under various learning rules.

Theorem 5.4 ([66]). *If Ξ is an supermodular game, it has at least one pure Nash equilibrium.*

Applications of supermodular games include the pricing-based power control algorithm designed in [26] for solving a sum utility maximization problem. In [26], each wireless transmitter adapts transmission power and charges interference price to interfering transmitters based on best response update in an implicit supermodular game. Sum rate maximization, which is a special case in [26], has been solved by [13] using gradient algorithm based on dual decomposition. As opposed to gradient methods that might need a small stepsize to converge to the optimum at the price of slow convergece, the convergence of the best response algorithm in the fictitious game [26] is ensured by supermodular game theory without appealing to stepsize.

Another particular game possessing equilibrium is the potential game.

Definition 5.4 ([43]). A strategic game Ξ is called

1. an exact potential game if there exists a function $\Phi : \mathscr{S} \to \mathbb{R}$ such that for all $v \in \mathscr{V}$, and $(x_v, x_{-v}) \in \mathscr{S}$, $x'_v \in \mathscr{S}_v$:

$$w_v(x_v, x_{-v}) - w_v\left(x'_v, x_{-v}\right) = \Phi(x_v, x_{-v}) - \Phi\left(x'_v, x_{-v}\right). \tag{5.32}$$

   If the payoff function w_v is continuously differentiable, then (5.32) is equivalent to

$$\frac{\partial w_v(x_v, x_{-v})}{\partial x_v} = \frac{\partial \Phi(x_v, x_{-v})}{\partial x_v}, \ \forall v \in \mathscr{V}. \tag{5.33}$$

2. an ordinal potential game if there exists a function $\Phi : \mathscr{S} \to \mathbb{R}$ such that for all $v \in \mathscr{V}$, and $(x_v, x_{-v}) \in \mathscr{S}$, $x'_v \in \mathscr{S}_v$: $sgn(w_v(x_v, x_{-v}) - w_v(x'_v, x_{-v})) = sgn(\Phi(x_v, x_{-v}) - \Phi(x'_v, x_{-v})))$.

For definitions of min-max potential games and state-based potential games, please refer to [56] and [41], respectively.

Theorem 5.5. *If Ξ is a potential game with a finite number of players, compact strategy sets, and continuous payoffs, then it has at least one pure Nash equilibrium.*

One good example of potential game is the application to cooperative control, where multiple agents interact with each other to achieve a common target, *e.g.* the consensus problem [51] and vehicle formation problem [56]. There are two advantages when the cooperative control is modelled as a potential game. First, for

cooperative control each agent updates its strategy by evaluating its effects on the common objective function. This evaluation will require to observe the decisions of all agents [40]. In the potential game formulation, each player has local payoff function that captures the player's marginal contribution to the potential function, which is the common objective of cooperative control. The implementation overhead is reduced in the game setting. Thus, one critical point in applying potential game to cooperative control is to assign each player a reasonable payoff function that is aligned with the potential function. Secondly, with cooperative control problems formulated as a potential game, there are many learning algorithms, adaptive to a time-varying environment, available that are guaranteed convergence to Nash equilibria.

5.5.2.2 Uniqueness of Nash Equilibrium

After establishing the existence of a Nash equilibrium the next issue is to study its uniqueness. Uniqueness of Nash equilibria is both critical for predicting outcome of a game and important for convergence issues. A general result is given by Rosen [58] to ensure that a game has a unique Nash equilibrium. Another value of [58] is that it points out a way to select one equilibrium if there are multiple equilibria. See [36] and the references therein for more discussions. The sufficient conditions in [58] change the structure of a game and restrict the payoff and strategy sets. When the best response of a game can be explicitly expressed, there are some weaker conditions that guarantee uniqueness of Nash equilibrium. The first result is based on properties of contraction mapping.

Definition 5.5 ([67]). $M(\cdot) : X \to X$, where X is a subset of $\mathbb{R}^{|\mathcal{V}|}$, is a contraction mapping if there exists $\varepsilon \in (0,1)$ such that $\|M(x) - M(y)\| \leq \varepsilon \|x - y\|$[4], $\forall x,y \in X$.

Theorem 5.6. *Suppose that $M(\cdot) : X \to X$ is a contraction mapping and X be a closed subset of $\mathbb{R}^{|\mathcal{V}|}$. Then M has a unique fixed point x^* that is globally asymptotically stable.*

When the contraction mapping theorem is applied, the best response correspondence is presumed to be a mapping. In the context of game theory, if the best response mapping $BR(\cdot) : \mathscr{S} \to \mathscr{S}$ is a contraction mapping, the sequence $\left\{x^{(k)}\right\}$ generated by $x^{(k+1)} = BR\left(x^{(k)}\right)$ converges to a unique fixed point x^* from any initial strategy profiles $x(0) \in \mathscr{S}$. According to the definition of Nash equilibrium, this fixed point x^* is the unique Nash equilibrium. The application of contraction mapping theorem to study uniqueness of Nash equilibrium can be found in [69] for scalar strategies and in [63] for vector strategies.

In addition to contraction mapping, if the best response correspondences satisfy some nice properties there can be a unique intersection in the strategy space. One of

[4] Here $\|\cdot\|$ is some norm.

such functions or correspondences is called standard function [72], which was generalized by [64] to include type-II standard function. Together this type of function is called two-sided scalable function.

Definition 5.6 ([64]). A vector function $I(\cdot): X \to X$ with X a subset of $\mathbb{R}$, is said to be two-sided scalable, if for all $\beta > 1$, $\forall x \in X$, $\forall x' \in X$, $(1/\beta)x \leq x' \leq \beta x$ implies

$$\frac{1}{\beta} I(x) < I\left(x'\right) < \beta I(x).$$

Theorem 5.7 ([64]). *If $I(\cdot): X \to X$ is two-sided scalable and a fixed point exists, then $I(\cdot)$ has a unique fixed point, which is globally asymptotically stable.*

Once again, if the best response mapping $BR(\cdot): \mathscr{S} \to \mathscr{S}$ of a strategic game is two sided scalable it converges to the unique Nash equilibrium when there is a Nash equilibrium of the game. Note that the above theorem can not guarantee the existence of Nash equilibrium. To study existence of Nash equilibrium one has to invoke Theorem 2 in [20] or turn to Brouwer's Fixed-Point Theorem in [7].

5.5.2.3 Efficiency of Nash Equilibrium

The Nash equilibrium discussed above provides a solution to multi-objective optimization problem where no agent can increase its performance through individual effort. Thus, it is an outcome of distributed decision making which could be less efficient than a possible scheme through cooperation between agents and/or as a result of centralized optimization. Equilibrium efficiency is also a criterion to select one from multiple equilibria, if there are more than one equilibrium.

A well known efficiency criterion is *Pareto optimality*. An outcome of a game is Pareto optimal if there is no other outcome that makes every player at least as well off and at least one player strictly better off. In other words, a Pareto-optimal outcome cannot be improved upon without hurting at least one player. It should be noted that a Nash equilibrium solving a social optimal problem such as those discussed in Example 7 is Pareto-optimal. However, in reality the competitive solution is far from social optimal or Pareto-optimal. One may expect to quantify the performance gap between the social optimal and Nash equilibria. The performance gap is known as the *price of anarchy [34]* , defined next.

Let us first represent the social performance achieved by all players at a given Nash equilibrium x as

$$SUM(x) = \sum_{v \in \mathscr{V}} \upsilon_v(x_v, x_{-v}),$$

where $\upsilon_v(x)$ denotes the utility (or payoff) of player v at equilibrium x. The social optimum OPT is defined to be the maximum $SUM(x)$ achieved by all players.

Definition 5.7. The price of anarchy of a game is the worst-case efficiency ratio among all pure strategy Nash equilibria,

$$PoA = \min_{x \in \mathscr{S}} \frac{SUM(x)}{OPT}$$

Example 5.10. Given the example considered in (5.14), in the case of price-anticipating users the problem turns into a game. Each user chooses b_i to maximize its payoff

$$w_i(b_i, b_{-i}) = \begin{cases} u_i\left(\frac{b_i}{\sum_{v=1}^{N} b_v} c\right) - b_i & \text{if } b_i > 0 \\ u_i(0) & \text{if } b_i = 0 \end{cases} \tag{5.34}$$

over non-negative b_i. Note that the payoff function in (5.34) maybe discontinuous at $b_i = 0$, if $\sum_{v \neq i} b_v = 0$. This discontinuity may preclude the existence of a Nash equilibrium [29]. The authors in [29] explore the effects of price-anticipation and prove that the price of anarchy is a 25% efficiency loss compared with the maximum possible aggregate utility in (5.13) or (5.14).

The reason for low efficiency of Nash equilibrium in a non-cooperative game is that each player aims to optimize its own performance without regarding the cost it imposes on others. Pricing (or taxation) has been proven to be an efficient way to improve efficiency. Here, we do not use price to generate revenue for the system but to use price to encourage players to use system resources more efficiently rather than the aggressive competition of the purely non-cooperative game. A pricing scheme is called incentive compatible if pricing enforces a Nash equilibrium that improves social welfare. An efficient price should reflect accurately the costs of usage of a resource and must take into account the individual player's effects on system. In [59], it has been shown that the utility in the energy-efficient power control game can be improved when some players deviate somewhat from the Nash equilibrium, *i.e.* the resulting equilibrium is not Pareto-optimal. To restrict interference (negative results of selfish behaviors), the authors use a usage-based pricing to improve the equilibrium utilities. However, this linear usage-based pricing in [59] is far from social optimum since it does not take into account individual player's effects on system performance. Maximizing the sum of coupled utilities in a non-cooperative environment usually involves some control message passing. To illustrate this, let us consider a social optimal problem,

$$\max_{x \in \mathscr{S}} \sum_{v \in \mathscr{V}} u_v(x), \tag{5.35}$$

where $u_v(x)$ is assumed to be differentiable. One of the necessary conditions for $x^* \in \mathscr{S}$ to be optimal for (5.35) is

$$\frac{\partial u_v(x^*)}{\partial x_v} + \sum_{m:m \neq v} \frac{\partial u_m(x^*)}{\partial x_v} = \lambda_v^*, \forall v \in \mathscr{V} \tag{5.36}$$

where λ_v^* is a Lagrange multiplier used to regulate the strategy selection within the strategy set of player v. In a non-cooperative game, where each player selfishly tries to optimize its own payoff rather than the common objective in (5.35), the condition

(5.36) can serve as a guideline to design an incentive compatible pricing scheme. Since each player solves

$$\max_{x_v \in \mathscr{S}_v} w_v(x), \ \forall v \in \mathscr{V}, \tag{5.37}$$

where $w_v(x) = u_v(x) - \kappa_v(x)$, the unit price $\kappa_v(x)$ that user v is charged for the common resource should be based on $\sum_{m:m\neq v} \frac{\partial u_m(x)}{\partial x_v}$. Specifically, $\kappa_v(x)$, $\forall v \in \mathscr{V}$ should be designed to guarantee that the Nash equilibrium for (5.37) also satisfies the first order necessary conditions of (5.35).

Applications of these techniques to communication systems can be found in, for example, [26, 39]. To bring the competitive players to solve a social utility maximization problem in (5.35), the authors in [26] duplicate the player sets into two sets. In the first set, each player updates its decision by best response to maximize its individual payoff in (5.37). In another set, each player charge prices according to the first order necessary condition (5.36). The existence and stability of a Nash equilibrium is guaranteed using supermodular game theory. In [39], the divisible resource allocation is addressed by proportional auction scheme. The efficiency of this auction is determined by the cost function, since it is related to the Lagrange multiplier, which is used to relax the global constraint in the corresponding social welfare maximization problem. To maximize the social welfare, the cost function is carefully designed by comparing the first order necessary conditions of local payoff function with those of social utility function. More discussion on pricing to improve equilibrium efficiency can be found in [42].

The intuition behind why pricing allows to improve the efficiency of Nash equilibria is that it introduces an (implicit) message passing between players such that the original interior equilibrium is driven to the Pareto frontier. In a word, the discrepancy of equilibrium states between the social optimum and selfish behavior is compensated by the price scheme.

5.6 Dynamics of Gradient Algorithms

When people propose a gradient algorithm to solve an optimization problem, one of the basic questions to be answered is whether the algorithm will converge to the desired equilibria. We have already provided several convergence results when the gradient is immediately and accurately available to the decision-makers. However, in practice, information is often delayed and sometimes distorted. To study such information limitations, it is often useful to study the properties of the corresponding differential equation under delays and perturbations. For this purpose, we next introduce the Lyapunov stability theory, which is widely used in control theory.

Let $x = 0$ be an equilibrium point for

$$\frac{dx(t)}{dt} = \vartheta(x(t)) \tag{5.38}$$

and $B \subset \mathbb{R}^n$ be a region containing 0. Let $V : B \to \mathbb{R}$ be a continuously differentiable function such that $V(x) > 0$, $\forall x \neq 0$ and $V(0) = 0$. There are the following conditions for various notions of stability.
(1) If $\frac{dV(x(t))}{dt} \leq 0$, $\forall x \in B$, then the equilibrium is stable and $V(x)$ is called a Lyapunov function.
(2) In addition, if $\frac{dV(x(t))}{dt} < 0$, $\forall x \in B \backslash \{0\}$, then the equilibrium is asymptotically stable.
(3) In addition to (1) and (2) above, if V is radially unbounded *i.e.* $V(x) \to \infty$ as $x \to \infty$ then the equilibrium is globally asymptotically stable.

Note that the above theorem also holds if the equilibrium is a nonzero $\hat{x}$. In this case, consider a system with state $y = x - \hat{x}$ and the results hold immediately.

5.6.1 Connection between Lyapunov Functions and Objective Functions

In this section we first start to consider

$$\min_{x \in \mathbb{R}^n} f(x), \tag{5.39}$$

where the function $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable and strictly convex. One of the simplest methods for solving (5.39) is the gradient descent algorithm (5.40), which attempts to maximize the decrease of the objective function in each iteration by updating the current iterate in the opposite direction of the gradient of f.

$$x^{(k+\Delta)} = x^{(k)} - \alpha \nabla f(x) \tag{5.40}$$

Let us use the gradient flow in (5.41) to approximate the sequence $\left\{x^{(k)}\right\}$ generated by (5.40).

$$\frac{dx}{dt} = \lim_{\Delta \to 0^+} \frac{x^{(k+\Delta)} - x^{(k)}}{\Delta} = -\alpha \nabla f(x). \tag{5.41}$$

When people study the dynamics of gradient method, Lyapunov functions are usually adopted to study the stability of stationary point (local or global optimum) to which the gradient algorithm may converge. However, there is no common method to construct a Lyapunov function, whose existence is only sufficient to guarantee the stability of a stationary point. A Lyapunov function can be regarded as the "energy" of a system. If one can prove that the energy along the considered dynamics is continuously decreasing, the system would finally settle down at the lowest-energy state. When we study the convergence of gradient method in an optimization setting, the convex cost function $f(x)$ will keep decreasing until it reaches a stationary point of the gradient dynamics, which coincides with the minimum of $f(x)$. Thus, one intuition of selecting Lyapunov function for $\dot{x} := \frac{dx}{dt} := g(x)$ is to associate it with

the objective function. More precisely, choose

$$V(x(t)) := f(x^*) - f(x(t)) \tag{5.42}$$

as the Lyapunov candidate, where x^* is one optimum solution of (5.39). It is straightforward to see that $V(x) \geq 0$, $\forall x \in X$ and $V(x) = 0$ iff $x = x^*$. Furthermore $V(x)$ is nondecreasing along the trajectories of (5.41) by showing

$$\frac{dV(x(t))}{dt} = \sum_{i=1}^{n} \frac{\partial V(x)}{\partial x_i} \cdot \frac{dx_i}{dt} = -\alpha \|\nabla f(x)\|^2 \leq 0 \tag{5.43}$$

with $\frac{dV(x(t))}{dt} = 0$ iff $x = x^*$. It follows that (5.42) is a Lyapunov function. According to Lyapunov stability theory, the unique minimum x^* is globally asymptotically stable.

For a concrete example, Kelly *et al.* [31] study the stability of dual-based flow control algorithm in communication networks, where the Lyapunov function coincides the concave objective function of dual network problem. For the stability of primal algorithm in network resource allocation [2], [11], the Lyapunov function is the same as the associated strictly concave objective function.

Similar like the direct connection between Lyapunov function and objective function in convex optimization problems, in some non-cooperative game, say potential game, its Lyapunov function is just the potential of the game. Let's consider a strategic non-cooperative game $\Xi = \left\{\mathscr{V}, \{\mathscr{S}_v\}_{v\in\mathscr{V}}, \{w_v\}_{v\in\mathscr{V}}\right\}$, where $\mathscr{V}$ is the set of $|\mathscr{V}|$ players; $\mathscr{S}_v$ is the set of strategies of player v and w_v is its payoff function.

Gradient based algorithm are usually adopted for each player to follow to reach a Nash equilibrium,

$$\frac{dx_v}{dt} = \alpha \frac{\partial w_v(x(t))}{\partial x_v}, \ \forall v \in \mathscr{V}, \tag{5.44}$$

where $\alpha > 0$ is the stepsize.

One interesting property of potential games is that the Lyapunov function for the gradient system (5.44) is just the (scaled version) of potential function $\Phi(x)$ given in (5.33). Let's consider the following candidate Lyapunov function [61]:

$$V(x) = \Phi_{\max} - \Phi(x),$$

where $\Phi(x^*) := \Phi_{\max}$ denotes the maximum value of the potential Φ over $\mathscr{S}$. It is easy to show the positiveness of $V(x)$, $\forall x \in \mathscr{S}$ except $x = x^*$. Following (5.33), $V(x(t))$ is non-decreasing along the trajectories of the system (5.44),

$$\frac{dV(x(t))}{dt} = -\nabla_x^T \Phi(x) \left(\frac{d}{dt}x(t)\right) = -\alpha \|\nabla_x \Phi(x)\|^2 \leq 0. \tag{5.45}$$

By LaSalle's invariance principle[5] and (5.45) the trajectories of (5.44) converge to the largest invariant set

[5] To be introduced afterwards.

$$\left\{x\in\mathscr{S}:\ \frac{dV\left(x\left(t\right)\right)}{dt}=0\right\}. \tag{5.46}$$

Since the set in (5.46) contains only the unique Nash equilibrium of the game Ξ if $\Phi\left(x\right)$ is strictly concave, the dynamics (5.44) converges to such an equilibrium asymptotically. In a potential game, although each player v selfishly maximizes its own payoff $w_v\left(x\right)$, it implicitly maximizes an imaginary objective, the potential function. Thus, $V\left(x\right)=\Phi_{\max}-\Phi\left(x\right)$ can be used to measure the "energy" accumulated along the trajectory (5.44).

By the discussions above, the convexity (concavity) plays an essential role in finding a candidate Lyapunov function. The connection between Lyapunov function and objective function may be invalidated in some cases. For example, the integration of gradient dynamics with respect to the variables does not result in a common objective function or the potential function in a non-cooperative game is hard to find. One may seek Lyapunov function by a more general method, the Krasovskii's method.

5.6.2 Krasovskii's Method

Let us use a saddle point problem to illustrate Krasovskii's method: find x^* and λ^* such that

$$L\left(x^*,\lambda\right)\leq L\left(x^*,\lambda^*\right)\leq L\left(x,\lambda^*\right),\ \forall x\in X,\ \forall\lambda\in M,$$

where $L:X\times M\rightarrow\mathbb{R}$ is a strictly convex-concave function. X and M are closed convex sets in $\mathbb{R}^n$ and $\mathbb{R}^m$. The saddle point function can for example be a Lagrangian

$$L\left(x,\lambda\right)=f_0\left(x\right)+\lambda^T f\left(x\right)$$

of a convex programming problem

$$x^*\in\arg\min\left\{f_0\left(x\right)|f_i\left(x\right)\leq 0,i=1,\ldots,m,\ \ x\in X\right\}.$$

Assuming that the function $L\left(x,\lambda\right)$ is differentiable, the sufficient and necessary conditions to be a saddle point are

$$x^*=P_X\left(x^*-\alpha\nabla_x L\left(x^*,\lambda^*\right)\right), \tag{5.47a}$$

$$\lambda^*=P_M\left(\lambda^*+\alpha\nabla_\lambda L\left(x^*,\lambda^*\right)\right) \tag{5.47b}$$

where $P_X\left(\cdot\right)$ and $P_M\left(\cdot\right)$ are the projection on sets X and M, respectively. The difference between the left and right side of of (5.47), which is equal to zero at the point x^*, λ^* and non-zero at an arbitrary point x, λ, specifies a mapping of the set $\mathbb{R}^n\times\mathbb{R}^m$ into itself. The resultant space can be viewed as a vector field with the fixed point x^*, λ^*. If $X=\mathbb{R}^n$, $M=\mathbb{R}^m$, this problem is written as the system of

$$\frac{dx}{dt} = -\alpha \nabla_x L(x,\lambda) \tag{5.48a}$$

$$\frac{d\lambda}{dt} = \alpha \nabla_\lambda L(x,\lambda) \tag{5.48b}$$

where α is the step-size. It can be checked that (5.48) admits a unique solution.

Obviously, $L(x,\lambda)$ is not a Lyapunov function of (5.48) since (5.48) accounts for the decrease in one variable x and increase in another variable λ. One alternative Lyapunov function is constructed by Krasovskii's method,

$$V(z) = \dot{z}^T A \dot{z}, \tag{5.49}$$

where $z = [x,\lambda]^T$, $A = \frac{1}{2} diag\left(\alpha^{-1}\right)$. $V(z)$ is positive except the equilibrium $z^* = [x^*,\lambda^*]^T$. It can be computed that the time-derivative of (5.49) along the trajectories of (5.48) results in

$$\dot{V}(z) = -\dot{z}^T \begin{bmatrix} \frac{\partial^2 L}{\partial x^2} & 0 \\ 0 & -\frac{\partial^2 L}{\partial \lambda^2} \end{bmatrix} \dot{z} \leq 0. \tag{5.50}$$

$\dot{V}(z) = 0$ happens only at the equilibrium point $z^* = [x^*,\lambda^*]^T$. Hence, the trajectory of dynamics (5.48) will tend to the saddle point of (x^*,λ^*) asymptotically.

Krasovskii's method is quite general and it can also be used to construct a Lyapunov function for the gradient based algorithms (5.41) and (5.44). Recently it has been used for studying stability of gradient algorithms for network resource allocation based on convex optimization [19]. Since the time-derivative of Lyapunov function (5.49) contains the second order global information, it can be used to prove the global asymptotic stability of the unique Nash equilibrium in a non-cooperative game [58], [1], [70]. However, imposing conditions on the Jacobian of pseudo-gradient [58]-[70] may be sometimes conservative since it ignores local structure of each player's payoff. By exploiting the local structure of payoff in [17], the same stability of gradient algorithm can be proved by a quadratic Lyapunov function without imposing global conditions proposed by [1].

5.6.3 Non-strictly Convex Problem

So far, we only discussed the stability of gradient algorithm for strictly concave/convex problem. For non-strictly convex/concave objective function, there may exist multiple equilibria. Then the corresponding gradient algorithm is no more guaranteed to converge to one of the equilibria asymptotically, see [12], [38]. We will invoke LaSalle's invariance principle to determine the convergence of gradient algorithms to multiple equilibria.

Theorem 5.8 (LaSalle's invariance principle [32]). *Consider the differential equation in* (5.38). *Let* $V : B \to \mathbb{R}$ *be a radially unbounded, continuoustly differentiable, positive definite function such that* $\dot{V}(x) \leq 0$ *for all* $x \in B$. *Let* $\mathscr{E}$ *be the set of points in*

B such that $\dot{V}(x) = 0$. *Let* $\mathscr{M}$ *be the largest invariant set*[6] *in* $\mathscr{E}$. *Then, every solution of* (5.38) *starting in B tends to* $\mathscr{M}$ *as* $t \to \infty$.

LaSalle's invariance principle implies that any equilibrium point of a gradient algorithm is in invariant set $\mathscr{M}$. Also the domain of attraction[7] of an equilibrium point is also an invariant set. However, we do not know whether all elements in invariant set are preferred in terms of stability and optimality. The key is to ensure that the invariant set includes the global optimal equilibria exclusively. Basically, the invariant set also gives us some insight to construct a Lyapunov function. Recall that the invariant set should include all global optimum for the gradient algorithm to converge to. One possible way of constructing Lyapunov function is to connect the time-derivative of Lyapunov function with KKT conditions, *i.e.* any equilibrium satisfying KKT conditions should be in the largest invariant set $\mathscr{M}$. After that, one may argue that the trajectories of gradient algorithm converge to the desired equilibrium in $\mathscr{M}$. They may also converge to non-equilibria in $\mathscr{M}$ if there are any. To achieve optimality, one usually establish conditions to exclude those undesired points in the set $\mathscr{M}$. This is done by studying the dynamics or properties of a reduced system, which is obtained by substituting the elements in $\mathscr{M}$ into original gradient systems.

Recently, a primal-dual algorithm was proposed in [73] to solve a non-strictly concave problem, which comes from uplink resource allocation problem in OFDM networks. The authors first established some properties that the elements in the largest invariant set should satisfy. Based on those properties they reduced the original primal-dual gradient-based algorithm into a set of linear differential equations. By studying stability of the reduced linear system, they equivalently excluded the undesired elements in $\mathscr{M}$. It should be noted that LaSalle's invariance principle can also be used to argue the uniqueness of Lagrange multiplier to which a primal-dual algorithm converges, although the Lagrange multiplier satisfying KKT conditions can be multiple, see [16] for network resource allocation example.

5.7 Conclusions

We have attempted to provide a tutorial overview of distributed optimization and games for decision-making in networked systems. Starting with a review of first-order methods for convex optimization, we have discussed how distributed optimization mechanisms can be designed using mathematical decomposition, networked optimization, and game theory. While decomposition methods result in very efficient computations, they typically require central coordination of subsystems.

[6] A set $\mathscr{I}$ is an invariant set for a dynamic system $\frac{dx(t)}{dt} = \vartheta(x(t))$ if every trajectory $x(t)$ which starts from a point in $\mathscr{I}$ remains in $\mathscr{I}$ for all time. For example, any equilibrium point is an invarient set.

[7] Let $\varphi(t;x)$ be the solution of (5.38) that starts at initial state x at time $t = 0$. Then, the domain of attraction is defined as the set of all points x such that $\varphi(t;x)$ is defined for $t \geq 0$ and $\lim_{t\to\infty} \varphi(t;x) = 0$.

Networked optimization techniques, on the other hand, remove the central coordination and subsystems communicate and cooperate only with their nearest neighbors to find the global optimum. Finally, techniques from non-cooperative games allow to eliminate the overhead traffic for coordination of subsystems altogether, but will in general not be able to find the global optimum (unless a pricing mechanism is introduced that encourages entities to strive towards the common good).

Naturally, a book chapter like this has to be selective in scope. Many interesting and useful ideas and results had to be left out. This includes, for example, optimization of systems with stochastic parameters or noise, and analysis of decomposition and networked optimization under information delay. For the material that we have presented, our focus has been on basic concepts and ideas rather than the very latest extensions and generalizations. Although many of the key results are now almost half a century old, they have proven to be very useful in a wide range of applications, from resource allocation in communication systems to wide-area control of infrastructures. It is our firm belief that these techniques will play an increasingly important role in engineering, as the systems that we build become more and more networked and interconnected, and as the requirements on their performance and resource-efficiency continue to increase.

References

1. T. Alpcan, T. Basar, and S. Dey. A power control game based on outage probabilities for multicell wireless data networks. *IEEE transactions on wireless communications*, 5(4), 2006.
2. T. Alpcan, X. Fan, T. Basar, M. Arcak, and J.T. Wen. Power control for multicell CDMA wireless networks: A team optimization approach. *Wireless Networks*, 14(5):647–657, 2008.
3. K. J. Arrow and L. Hurwicz. *Essays in Economics and Econometrics*, chapter Decentralization and Computation in Resource Allocation. University of North Carolina Press, Rayleigh, North Carolina, 1960.
4. J. F. Benders. Partitioning procedures for solving mixed-variables programming. *Numerische Matematik*, 4:238–252, 1962.
5. D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.
6. D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and distributed computation: numerical methods*. Prentice-Hall, 1989.
7. K.C. Border. *Fixed point theorems with applications to economics and game theory*. Cambridge Univ Pr, 1989.
8. S. Boyd, P. Diaconis, and L. Xiao. Fastest mixing markov chain on a graph. *SIAM Review*, 46:667–689, 2004.
9. S. P. Boyd. Course material for ee364b, stanford university. Available via http://www.stanford.edu/class/ee364b/, 2007.
10. S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
11. L. Chen, SH Low, and J.C. Doyle. Joint congestion control and media access control design for wireless ad hoc networks. In *Proceedings of IEEE Infocom*, pages 2212–2222, 2005.
12. M. Chen, M. Ponec, S. Sengupta, J. Li, and P.A. Chou. Utility maximization in peer-to-peer systems. In *Proc. ACM Sigmetrics*, 2008.
13. Mung Chiang. Balancing transport and physical layers in wireless multihop networks: Jointly optimal congestion control and power control. *IEEE Journal on Selected Areas in Communications*, 23:104–116, 2005.

14. G. Cohen. Optimization by decomposition and coordination: a unified approach. *IEEE Transactions on Automatic Control*, 23(2):222–232, 1978.
15. G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8:101–111, 1960.
16. A. Eryilmaz and R. Srikant. Joint congestion control, routing and mac for stability and fairness in wireless networks. *IEEE Journal on Selected Areas in Communications*, 24:1514–1524, 2006.
17. X. Fan, T. Alpcan, M. Arcak, TJ Wen, and T. Basar. A passivity approach to game-theoretic CDMA power control. *Automatica*, 42(11):1837–1847, 2006.
18. H.E. Fawal, D. Georges, and G. Bornard. Optimal control of complex irrigation systems via decomposition-coordination and the use of augmented lagrangian. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 3874 – 3879, San Diego, CA, November 1998.
19. D. Feijer and F. Paganini. Krasovskiis Method in the Stability of Network Control. In *Proceedings of the 2009 conference on American Control Conference*, pages 3292–3297. Institute of Electrical and Electronics Engineers Inc., The, 2009.
20. D. Fudenberg and J. Tirole. *Game theory*. MIT Press, 1991.
21. A. M. Geoffrion. Elements of large-scale mathematical programming I–II. *Management Science*, 16:652–691, 1970.
22. David Goodman and Narayan Mandayam. Power control for wireless data. *IEEE PERSONAL COMMUNICATIONS*, 7:48–54, 2000.
23. W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109, 1970.
24. J.-B. Hiriart-Urruty and C. Lemaréchal. *Fundamentals of Convex Analysis*. Springer, 2001.
25. K. Holmberg. *Design models for hierarchical organizations: computation, information and decentralization*, chapter Primal and dual decomposition as organizational design: price and/or resource directive decomposition, pages 61–92. Kluwer Academic Publishers, 1995.
26. Jianwei Huang, Randall A. Berry, and Michael L. Honig. Distributed interference compensation for wireless networks. *IEEE Journal on Selected Areas in Communications*, 24:1074–1084, 2006.
27. B. Johansson, T. Keviczky, M. Johansson, and K.H. Johansson. Subgradient methods and consensus algorithms for solving convex optimization problems. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pages 4185 –4190, 2008.
28. Björn Johansson, Maben Rabi, and Mikael Johansson. A randomized incremental subgradient method for distributed optimization in networked systems. *SIAM Journal on Optimization*, 20(3):1157–1170, 2009.
29. Ramesh Johari, Shie Mannor, and John N. Tsitsiklis. Efficiency loss in a network resource allocation game: The case of elastic supply. *Mathematics of Operations Research*, 29:407–435, 2004.
30. F. Kelly. Charging and rate control for elastic traffic. *European transactions on Telecommunications*, 8(1):33–37, 1997.
31. F.P. Kelly, A.K. Maulloo, and D.K.H. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research society*, 49(3):237–252, 1998.
32. H.K. Khalil. Nonlinear systems. 3rd, 2002.
33. J. Koshal, A. Nedic, and U.V. Shanbhag. Distributed multiuser optimization: Algorithms and error analysis. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 4372 – 4377, 2009.
34. E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. *Computer Science Review*, 3(2):65–69, 2009.
35. T. Larsson, M. Patriksson, and A.-B. Strömberg. Ergodic primal convergence in dual subgradient schemes for convex programming. *Mathematical Programming*, 86:283–312, 1999.
36. S. Lasaulce, M. Debbah, E. Altman, E.N.S. de Cachan, and F. Cachan. Methodologies for analyzing equilibria in wireless games. *IEEE Signal Processing Magazine*, 26(5):41–52, 2009.

37. L. S. Lasdon. *Optimization Theory for Large Systems*. Macmillan Co., New York, N. Y., 1970.
38. X. Lin and N.B. Shroff. Utility maximization for communication networks with multipath routing. *IEEE Transactions on Automatic Control*, 51(5), 2006.
39. R. Maheswaran and T. Basar. Efficient signal proportional allocation (ESPA) mechanisms: Decentralized social welfare maximization for divisible resources. *IEEE Journal on Selected Areas in Communications*, 24(5), 2006.
40. J.R. Marden, G. Arslan, and J.S. Shamma. Cooperative control and potential games. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 39(6):1393–1407, 2009.
41. J.R. Marden and A. Wierman. Overcoming limitations of game-theoretic distributed control. In *48th IEEE Conference on Decision and Control*, 2009.
42. R.R. Mazumdar, C.A. Courcoubetis, N. Duffield, G. Kesidis, A. Odlyzko, R. Srikant, J. Walrand, and P. Cosman. Guest Editorial Price-Based Access Control and Economics of Networking. *IEEE Journal on Selected Areas in Communications*, 24(5), 2006.
43. D. Monderer and L.S. Shapley. Potential games. *Games and economic behavior*, 14:124–143, 1996.
44. A. Nedic and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *Automatic Control, IEEE Transactions on*, 54(1):48 –61, jan. 2009.
45. Angelia Nedic and Dimitri P. Bertsekas. Incremental subgradient methods for nondifferentiable optimization. *SIAM J. on Optimization*, 12(1):109–138, 2001.
46. Angelia Nedić and Asuman Ozdaglar. Approximate primal solutions and rate analysis for dual subgradient methods. *SIAM Journal on Optimization*, 19(4):1757–1780, 2008.
47. Y. Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$. *Doklady AN SSSR (translated as Soviet Math. Docl.)*, 269:543–547, 1983.
48. Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 1995.
49. Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Netherlands, 2003.
50. R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multiagent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.
51. R. Olfati-Saber, J.A. Fax, and R.M. Murray. Consensus and cooperation in networked multiagent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.
52. A. Olshevsky and J. N. Tsitsiklis. Convergence rates in distributed consensus and averaging. In *Proceedings of IEEE CDC*, 2006.
53. B. Polyak. *Introduction to Optimization*. Optimization Software, 1987.
54. M. Rabbat and R. Nowak. Distributed optimization in sensor networks. In *Information Processing in Sensor Networks, 2004. IPSN 2004. Third International Symposium on*, pages 20 – 27, 2004.
55. S. Sundhar Ram, Angelia Nedic, and Venugopal V. Veeravalli. Incremental stochastic subgradient algorithms for convex optimization. *SIAM Journal on Optimization*, 20(2):691–717, 2009.
56. A. Rantzer. Using game theory for distributed control engineering. *Department of Automatic Control, Lund University, Sweden, Tech. Rep. ISRN LUTFD2/TFRT–7620–SE, July*, 2008.
57. R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1996.
58. JB Rosen. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica: Journal of the Econometric Society*, 33(3):520–534, 1965.
59. C.U. Saraydar, N.B. Mandayam, and D.J. Goodman. Efficient power control via pricing in wireless data networks. *IEEE transactions on Communications*, 50(2):291–303, 2002.
60. I.D. Schizas, A. Ribeiro, and G.B. Giannakis. Consensus in ad hoc wsns with noisy links part i: Distributed estimation of deterministic signals. *Signal Processing, IEEE Transactions on*, 56(1):350 –364, jan. 2008.
61. G. Scutari, S. Barbarossa, and DP Palomar. Potential games: A framework for vector power control problems with coupled constraints. In *2006 IEEE International Conference on Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings*, volume 4, 2006.
62. N. Z. Shor. *Minimization methods for non-differentiable functions*. Springer-Verlag, 1985.

63. K.W. Shum, K.K. Leung, and C.W. Sung. Convergence of iterative waterfilling algorithm for Gaussian interference channels. *IEEE Journal on Selected Areas in Communications*, 25(6):1091–1100, 2007.
64. C.W. Sung and K.K. Leung. A generalized framework for distributed power control in wireless networks. *IEEE Transactions on Information Theory*, 51(7):2625, 2005.
65. Y. Takahara. Multilevel approach to dynamic optimization. Technical Report SRC-50-C-64-18, Systems Research Center, Case Western Reserve University, 1964.
66. D.M. Topkis. *Supermodularity and complementarity*. Princeton Univ Pr, 1998.
67. R. Walter. *Principles of mathematical analysis*. McGraw-Hill, 1976.
68. L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53(1):65–78, 2004.
69. B. Yang, G. Feng, and X. Guan. Noncooperative random access game via pricing in ad hoc networks. In *2007 46th IEEE Conference on Decision and Control*, pages 5704–5709, 2007.
70. B. Yang, G. Feng, Y. Shen, C. Long, and X. Guan. Channel-Aware Access for Cognitive Radio Networks. *IEEE transactions on vehicular technology*, 58(7):3726–3737, 2009.
71. B. Yang, Y. Shen, M. Johansson, and X. Guan. Threshold-based Multichannel Access with Energy Constraint . In *ICC 2010-IEEE International Conference on Communications*, 2010.
72. R.D. Yates. A framework for uplink power control in cellular radio systems. *IEEE Journal on Selected Areas in Communications*, 13(7):1341–1347, 1995.
73. X. Zhang, L. Chen, J. Huang, M. Chen, and Y.P. Zhao. Distributed and optimal reduced primal-dual algorithm for uplink OFDM resource allocation . In *2009 Joint 48th IEEE Conference on Decision and Control (CDC) and 28th Chinese Control Conference (CCC 2009)*, 2009.