Distributed Consensus-Based Optimization

Advanced Topics in Control 2018: Distributed Systems & Control

Florian Dörfler

We have already seen in previous lectures that optimization problems defined over a network of agents with individual cost functions lead to solutions that can be computed via distributed average consensus; see the least-square estimation (Sections 1.5.1 and 8.4.2), graph partitioning (Section 6.4), and least-energy network routing (Exercise E6.13). In this lecture, we formalize the previous examples and outline the framework of distributed consensus-based optimization.

In Section 1, we review constrained convex optimization problems and present a suite of algorithms solve them. We present the algorithms from a control-theoretic perspective and primarily in a continuous-time setting to avoid tangential discussions on step-size selection. In Section 2, we consider optimization problems with separable objective functions, each of which encodes the private cost of an agent. We show how consensus constraints and the previously introduced algorithms can be used to coordinate the agents in a distributed fashion to reach a common optimizer.

1 Review of Optimization Theory

In the following we provide a brief review of some elements of convex analysis optimization algorithms. We refer to [18, 3, 2, 15] for further details and extensions. Our presentation is inspired by [11, 23, 22, 8, 6, 7, 10, 9, 19] treating optimization algorithms from a control-theoretic perspective.

Notation: In the literature there are multiple notations and conventions for the gradient of a function $f : \mathbb{R}^n \to \mathbb{R}$. To avoid any ambiguities, we define $\frac{\partial f}{\partial x} : \mathbb{R}^n \to \mathbb{R}^n$ as the *column vector* of partial derivatives. Accordingly, $\frac{\partial f(x)}{\partial x} \in \mathbb{R}^n$ is the evaluation of the gradient at $x \in \mathbb{R}^n$. We will interchangeably use $\frac{\partial f(x)}{\partial x}$ and $\frac{\partial f}{\partial x}(x)$ to highlight the evaluation of the gradient $\frac{\partial f}{\partial x}$ at $x \in \mathbb{R}^n$.

1.1 Unconstrained Optimization

Consider an unconstrained optimization problem of the form

$$\text{minimize}_{x \in \mathbb{R}^n} \; f(x)\,, \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is the *objective function* to be optimized. A point $x^\star \in \mathbb{R}^n$ is said to be a (global) *minimizer* of the optimization problem (1) if

$$f(x^\star) \leq f(x) \qquad \text{for all } x \in \mathbb{R}^n\,,$$

and $f(x^\star)$ is a (global) *minimum*. A (global) minimizer and minimum is said to be *strict* if

$$f(x^\star) < f(x) \qquad \text{for all } x \in \mathbb{R}^n \setminus \{x^\star\}\,,$$

in which case we may speak of *the* global minimizer and minimum. These concepts can also be defined *locally* in a neighborhood of $x^\star$ though we will not do so in this lecture.

While optimization theory has been developed for quite general classes of functions f, we will put some regularity properties on the function f – most importantly convexity:

- A function $f : \mathbb{R}^n \to \mathbb{R}$ is said be *convex* if $f(\alpha x + \beta y) \leq \alpha f(x) + \beta f(y)$ for all x and y in $\mathbb{R}^n$ and for all convex combination coefficients α and β, i.e., coefficients satisfying $\alpha, \beta \geq 0$ and $\alpha + \beta = 1$. A function is said to be *strictly convex* if the previous inequality holds strictly. Intuitively, a function f is convex if if the line segment between any two points on the graph of the function $\{(x, y) \in \mathbb{R}^n \times \mathbb{R} \,\big|\, y = f(x)\}$ lies above or on the graph; see Figure 1(a).

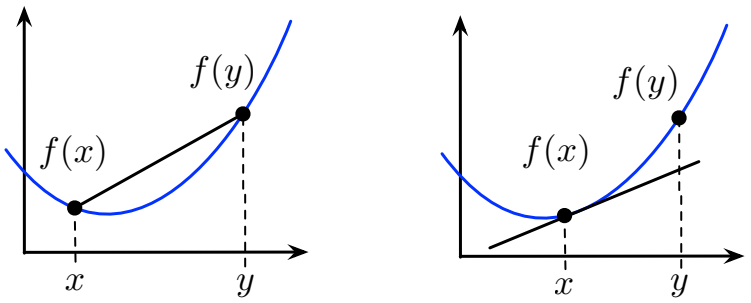
- A function $f : \mathbb{R}^n \to \mathbb{R}$ is said be *concave* if $-f$ is convex. All properties of convex functions and minima carry over analogously to concave functions and maxima.

- If f is continuously differentiable, then f is convex if and only if

$$f(y) \geq f(x) + (y - x)^\top \frac{\partial f(x)}{\partial x} \qquad \text{for all } x, y \in \mathbb{R}^n \,. \tag{2}$$

  If the above inequality (2) holds strictly for $x \neq y$, then f is strictly convex. Intuitively, condition (2) means that the first-order Taylor approximation globally underestimates the function f, and the function lies above all of its tangents; see Figure 1(b) for an illustration.

- If f is twice continuously differentiable, then f is convex if and only if the symmetric Hessian matrix $\text{Hess} f(x) = \frac{\partial^2 f(x)}{\partial x^2} \in \mathbb{R}^{n \times n}$ with (i, j) element $\frac{\partial^2 f(x)}{\partial x_i \partial x_j}$ is positive semidefinite for all $x \in \mathbb{R}^n$. If $\text{Hess} f(x)$ is positive definite for all $x \in \mathbb{R}^n$, then f is strictly convex. The converse is not necessarily true, see the strictly convex function $f(x) = x^4$.



(a) illustration of convexity (b) underestimator property

Figure 1: Figure 1(a) illustrates the definition of convexity (i.e., the function is below any line segment), and Figure 1(b) illustrates the underestimator property (2) (i.e., the function is above all of its tangents).

The global minimum of a continuously differentiable and strictly convex function is easily found:

Lemma 1.1 (Algebraic characterization of minimizer)**.** *Consider the optimization problem* (1) *and assume that f is continuously differentiable and strictly convex. Then $x^\star \in \mathbb{R}^n$ is the global minimizer if and only if*

$$\frac{\partial f}{\partial x}(x^\star) = \mathbb{0}_n \,. \tag{3}$$

Proof. By the underestimator property (2), we have $f(y) > f(x) + \frac{\partial f(x)}{\partial x}^\top (y - x)$ for all distinct $x, y \in \mathbb{R}^n$. If x satisfies $\frac{\partial f(x)}{\partial x} = \mathbb{0}_n$, then $f(y) > f(x)$ for all $y \in \mathbb{R}^n \setminus \{x\}$. Hence, $x = x^\star$ is the global minimizer. □

Another useful property of strictly convex functions is that their gradients are one-to-one mappings thus giving hope to uniquely solve equations involving $\frac{\partial f(x)}{\partial x}$; see Exercise 1.

1.2 Gradient Descent Algorithm

Recall that the gradient $\frac{\partial f}{\partial x}$ of the function f points orthogonally to any level set, $\{x \in \mathbb{R}^n \mid f(x) = c\}$ for any $c \in \text{image}(f)$, in the direction of steepest ascent; see Figure 2(a). Hence, the minimizer of a strictly convex function can be found via the *negative gradient flow*

$$\dot{x} = -\frac{\partial f(x)}{\partial x}^\top . \tag{4}$$

A block-diagram of the negative gradient flow (4) with an additional gain can be found in Figure 2(b). The convergence of the negative gradient flow can be established under mild assumptions.

Theorem 1.2 (Convergence of negative gradient flow)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable and strictly convex. Then each solution $t \mapsto x(t)$ of the negative gradient flow (4) converges to the global minimizer: $\lim_{t\to+\infty} x(t) = x^\star$.*

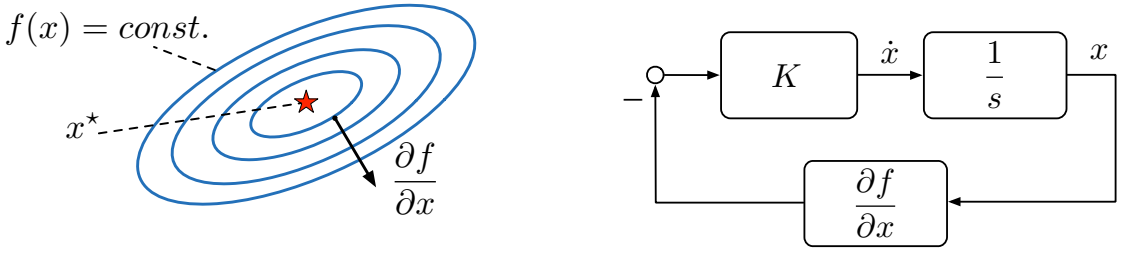
Proof. Consider the function $t \mapsto V(x(t)) = \frac{1}{2}\|x(t) - x^\star\|^2$ measuring the Euclidean distance of $x(t)$ to the optimizer $x^\star$. The derivative of the function $V(x)$ along its negative gradient flow is

$$\dot{V}(x) = -(x - x^\star)^\top \frac{\partial f(x)}{\partial x}^\top \leq f(x^\star) - f(x) \leq 0\,,$$

where the first inequality follows from evaluating the underestimator property (2) for $y = x^\star$:

$$f(x^\star) \geq f(x) + \frac{\partial f(x)}{\partial x}^\top (x^\star - x) \quad \Leftrightarrow \quad -(x - x^\star)^\top \frac{\partial f(x)}{\partial x}^\top \leq f(x^\star) - f(x) \leq 0 \qquad \text{for all } x \in \mathbb{R}^n\,.$$

Due to strict convexity of f, we have that $\dot{V}(x) = 0$ if and only if $x = x^\star$. The statement follows from the LaSalle Invariance Principle (see Theorem A.1), where as a set W we adopt, for any initial condition $x_0 \in \mathbb{R}^n$, the compact and forward invariant sublevel set $\{x \in \mathbb{R}^n \mid V(x) \leq V(x_0)\}$. □



(a) geometry of unconstrained optimization (b) block-diagram of negative gradient flow (5)

Figure 2: Figure 2(a) illustrates the geometry of an unconstrained, strictly convex, and smooth optimization problem. Figure 2(b) shows the weighted negative gradient flow (5) in a block-diagram.

Remark 1.3 (Convergence conditions and convergence rates)**.** *An alternative proof method under even less stringent assumptions (not exploiting convexity) can be found in Theorem 14.13.*

The convergence rate of the negative gradient flow is studied in Exercise 2, and it can be made arbitrary fast by multiplying a positive definite gain matrix $K \in \mathbb{R}^{n\times n}$ as follows (see Exercise 3)

$$\dot{x} = -K \frac{\partial f(x)}{\partial x}^{\top}. \tag{5}$$

Of course, in a digital computer the negative gradient flow (4) is implemented in discrete time.

$$x^{+} = x - \varepsilon \frac{\partial f(x)}{\partial x}^{\top}, \tag{6}$$

where the step-size $\varepsilon > 0$ has to be chosen sufficiently small (see Exercise 4). Hence, the discrete-time convergence rate is inherently limited by a sufficiently small step-size needed for convergence. This fundamental dilemma can be circumvented by time-varying step-sizes or acceleration methods incorporating higher-order dynamics, e.g., an Euler-discretized implementation of the dynamics

$$\ddot{x} + a\,\dot{x} + b \frac{\partial f(x)}{\partial x}^{\top} = 0, \tag{7}$$

where $a, b > 0$ are possibly time-varying gains; see Exercise 5. Since we do not want to further digress in step-size selection methods, we will henceforth study only continuous-time algorithms. We refer to the Appendix C for a more in-depth treatment of some of these concepts. □

1.3 Equality-Constrained Optimization

In the following, we consider the *equality-constrained optimization problem*

$$\text{minimize}_{x\in\mathbb{R}^n}\ f(x) \tag{8a}$$
$$\text{subject to } g(x) = Ax - b = \mathbb{0}_m, \tag{8b}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable objective function, and the constraint equation is specified by the matrix $A \in \mathbb{R}^{m\times n}$ and the vector $b \in \mathbb{R}^m$. We will generally assume that $\operatorname{rank}(A) < n$ so that the constraint equation leaves some room for optimizing x.

The constrained problem (8) can be reduced to an unconstrained one by modifying it to

$$\text{minimize}_{x\in\mathbb{R}^n}\ f(x) + \mathcal{I}(\|g(x)\|)$$

where $\mathcal{I}(\cdot)$ is the indicator function that evaluates to 0 whenever $\|g(x)\| = 0$ and ∞ otherwise. Whereas this approach formally leads to an unconstrained problem, it is impractical for the methods developed in Sections 1.1 and 1.2 requiring differentiability of the objective function. Alternatively, the constraint $g(x) = \mathbb{0}_m$ can be enforced through a *soft penalty* of the form

$$\text{minimize}_{x\in\mathbb{R}^n}\ f(x) + \frac{\rho}{2}\|g(x)\|^2, \tag{9}$$

where $\rho > 0$. Clearly, this soft penalty method does generally not perfectly enforce the constraint $g(x) = \mathbb{0}_m$ unless $\rho \to \infty$, but we will frequently return to this idea in the following subsections. In the following, we first resort to a geometrically inspired approach to constrained optimization.

1.4 Duality theory

The geometry of the constrained optimization problem (8) is illustrated in Figure 3. Figure 3 suggests that x is a strict minimizer provided that the constraint $g(x) = \mathbb{0}_m$ is met, and the level sets of the objective function and the constraint equation are tangent (or equivalently, their normal vectors $\frac{\partial f(x)}{\partial x}$ and $\frac{\partial g(x)}{\partial x}$ are linearly dependent). These two conditions can be formalized as follows: if $x \in \mathbb{R}^n$ is a minimizer of (8), then there is $\lambda \in \mathbb{R}^m$ so that (x, λ) satisfy the following conditions:

$$\text{(i) feasibility condition: } \mathbb{0}_m = g(x) = Ax - b\,, \tag{10a}$$

$$\text{(ii) tangency condition: } \mathbb{0}_n = \frac{\partial f(x)}{\partial x} + \frac{\partial g(x)}{\partial x}^\top \lambda\,. \tag{10b}$$

The conditions (10), termed *KKT conditions* (after Karush, Kuhn, and Tucker), are necessary conditions for optimality of x. The variable $\lambda \in \mathbb{R}^m$ is termed a *Lagrange multiplier* or *dual variable*. Analogously, x is termed the *primal variable*. Additionally, if f is convex, then the KKT conditions (10) are sufficient for optimality and their solutions $(x^\star, \lambda^\star)$ specify all minimizers and optimal dual variables. If f is strictly convex and A has full rank, then the KKT conditions admit only a single solution $(x^\star, \lambda^\star)$, namely the strict global minimizer and associated dual variable.

1.5 Linearly-constrained quadratic programs

For the following developments, it is instructive to consider, as a running example, the linearly-constrained quadratic (LQ) program

$$\text{minimize}_{x\in\mathbb{R}^n}\ f(x) = \frac{1}{2}(x - \underline{x})^\top P(x - \underline{x}) \tag{11a}$$

$$\text{subject to } g(x) = Ax - b = \mathbb{0}_m\,, \tag{11b}$$

where $\underline{x} \in \mathbb{R}^n$ and $P \in \mathbb{R}^{n\times n}$ is a positive semidefinite matrix. Since $\mathrm{Hess} f(x) = P$, the LQ program is convex (respectively strictly convex) if P is positive semidefinite (respectively, positive definite). In either case, the KKT conditions (10) are

$$\underbrace{\begin{bmatrix} -P & -A^\top \\ A & \mathbb{0} \end{bmatrix}}_{=\mathcal{A}} \begin{bmatrix} x \\ \lambda \end{bmatrix} = \begin{bmatrix} -P\underline{x} \\ b \end{bmatrix} \tag{12}$$

The matrix $\mathcal{A}$ is a so-called *saddle matrix* [1]. The reasons for this terminology will become clear in the next section. The following lemma summarizes some properties of saddle-matrices for the general case when P is positive semidefinite. We leave the proof for the reader; see Exercise 12.

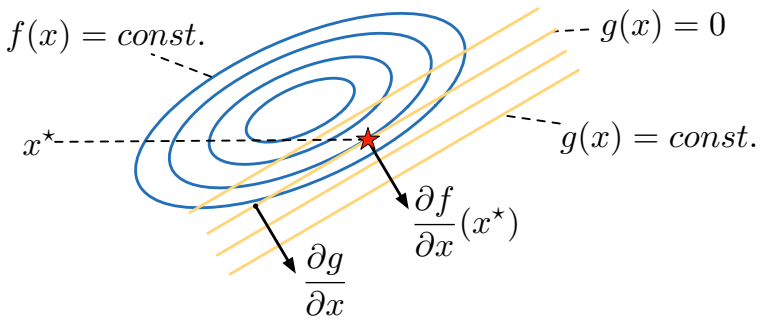


Figure 3: Geometry of a strictly convex, smooth, and equality-constrained optimization problem

Lemma 1.4 (Spectrum of saddle matrices). *Consider a positive semidefinite matrix $P \in \mathbb{R}^{n\times n}$ and a not necessarily square matrix $A \in \mathbb{R}^{m\times n}$ with $n \geq m$ forming the composite saddle-matrix*

$$\mathcal{A} = \begin{bmatrix} -P & -A^\top \\ A & \mathbb{0} \end{bmatrix}.$$

The saddle matrix $\mathcal{A}$ has the following properties:

1) *all eigenvalues are in the closed left half-plane:* $\text{spec}(\mathcal{A}) = \{\lambda \in \mathbb{C} \mid \Re(\lambda) \leq 0\}$. *Moreover, all eigenvalues on the imaginary axis have equal algebraic and geometric multiplicity;*

2) *if kernel$(P) \cap$ image$(A^\top) \subseteq \{\mathbb{0}_n\}$, then $\mathcal{A}$ has no eigenvalues on the imaginary axis except for 0; and*

3) *if P is positive definite and A has full rank, then $\mathcal{A}$ has no eigenvalues on the imaginary axis.*

Observe that as a consequence of Lemma 1.4, if P is positive definite and A has full rank m, then the KKT equations deliver a unique solution $(x^\star, \lambda^\star)$.

1.6 Lagrangian function

It is convenient to consider the *Lagrangian function* $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ for the problem (8):

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^\top g(x) = f(x) + \lambda^\top (Ax - b). \tag{13}$$

Observe that the Lagrangian function is (strictly) convex in x whenever f is (strictly) convex and linear (and thus concave) in λ. In this case, the Lagrangian function admits a set of *saddle points* $(x^\star, \lambda^\star) \in \mathbb{R}^n \times \mathbb{R}^m$ satisfying

$$\mathcal{L}(x^\star, \lambda) \leq \mathcal{L}(x^\star, \lambda^\star) \leq \mathcal{L}(x, \lambda^\star) \qquad \text{for all } (x, \lambda) \in \mathbb{R}^n \times \mathbb{R}^m. \tag{14}$$

Since $\mathcal{L}(x, \lambda)$ is differentiable, the saddle points in (14) can be obtained as solutions to the equations

$$\mathbb{0}_n = \frac{\partial \mathcal{L}(x,\lambda)}{\partial x} = \frac{\partial f(x)}{\partial x} + \frac{\partial g(x)}{\partial x}^\top \lambda \tag{15a}$$

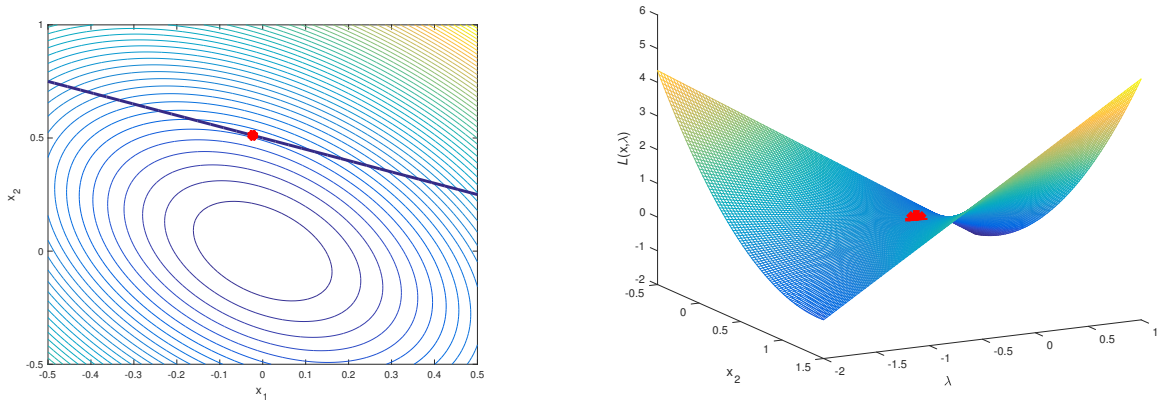
$$\mathbb{0}_m = \frac{\partial \mathcal{L}(x,\lambda)}{\partial \lambda} = g(x) = Ax - b. \tag{15b}$$

Observe that the saddle-point equations (15) coincide with the KKT conditions (10) (or (12) in the LQ case), and thus also reveal the minimizers and the optimal dual variables. In the literature on optimization, the dual variable λ is often interpreted as the price of the constraint entering linearly in the Lagragian (13). For the subsequent algorithmic development we rely on the Lagrangian function (13) and develop algorithms that seek its stationary points (15).

Example 1 (LQ program). *Consider the LQ program (11) with $x \in \mathbb{R}^2$ and with the matrices*

$$P = \begin{bmatrix} 2.6 & 0.8 \\ 0.8 & 1.4 \end{bmatrix}, \; \underline{x} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \; A = \begin{bmatrix} 1 & 2 \end{bmatrix}, \; b = 1$$

describing the objective and the constraint. The matrix P is positive definite with eigenvalues $\{1, 3\}$. The KKT conditions (12) determine the optimizer as $x^\star = [-0.0233 \;\; 0.5116]^\top$ and the optimal multiplier as $\lambda^\star = -0.3488$. The geometry of the optimization problem depicting the level sets of the objective, the constraint equation, as well the optimizer are shown in Figure 4(a). The Lagrangian $\mathcal{L}(x, \lambda)$ projected on $x_1 = 0$ is shown in Figure 4(b) clearly depicting the saddle point. □



(a) geometry of the LQ program

(b) Lagrangian of the LQ program

Figure 4: Figure 4(a) illustrates the geometry of of the LQ program including the ellipsoidal level sets of the objective and the linear equality constraint. Figure 4(b) shows the Lagrangian $\mathcal{L}(x, \lambda)$ projected on $x_1 = 0$. Both figures show the optimizer $x^\star$ and associated multiplier $\lambda^\star$ as red dots.

1.7 Primal-Dual Saddle-Point Algorithm

Since the Lagrangian function is convex in x and concave in λ, we can compute its saddle points by following the so-called *saddle-point flow*, consisting of a positive and negative gradient flow:

$$\dot{x} = -\frac{\partial \mathcal{L}(x,\lambda)}{\partial x} = -\frac{\partial f(x)}{\partial x} - \frac{\partial g(x)}{\partial x}^\top \lambda = -\frac{\partial f(x)}{\partial x} - A^\top \lambda \tag{16a}$$

$$\dot{\lambda} = +\frac{\partial \mathcal{L}(x,\lambda)}{\partial \lambda} = g(x) = Ax - b. \tag{16b}$$

The convergence of the saddle-point flow (16) can now be proved analogously to the proof of the negative gradient flow (4) in Theorem 1.2. The proof is left for the reader in Exercise 6.

Theorem 1.5 (Convergence of saddle-point flow)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable and strictly convex. Then each solution $t \mapsto (x(t), \lambda(t))$ of the saddle-point flow (16) converges to the set of saddle points $(x^\star, \lambda^\star)$.*

Similarly to the weighted gradient flow (5), additional positive definite convergence gain matrices K_1 and K_2 can be added to improve the convergence of the saddle-point flow (16) (see Exercise 7):

$$\dot{x} = -K_1 \frac{\partial \mathcal{L}(x,\lambda)}{\partial x} = -K_1 \frac{\partial f(x)}{\partial x} - K_1 A^\top \lambda \tag{17a}$$

$$\dot{\lambda} = +K_2 \frac{\partial \mathcal{L}(x,\lambda)}{\partial \lambda} = K_2 (Ax - b). \tag{17b}$$

The resulting block-diagram is shown in Figure 5 and reveals the saddle-point flow (17) as a skew-symmetric interconnection of the primal gradient dynamics (5) (see Figure 2(b)) together with an integral controller for the dual dynamics penalizing the constraint violation.

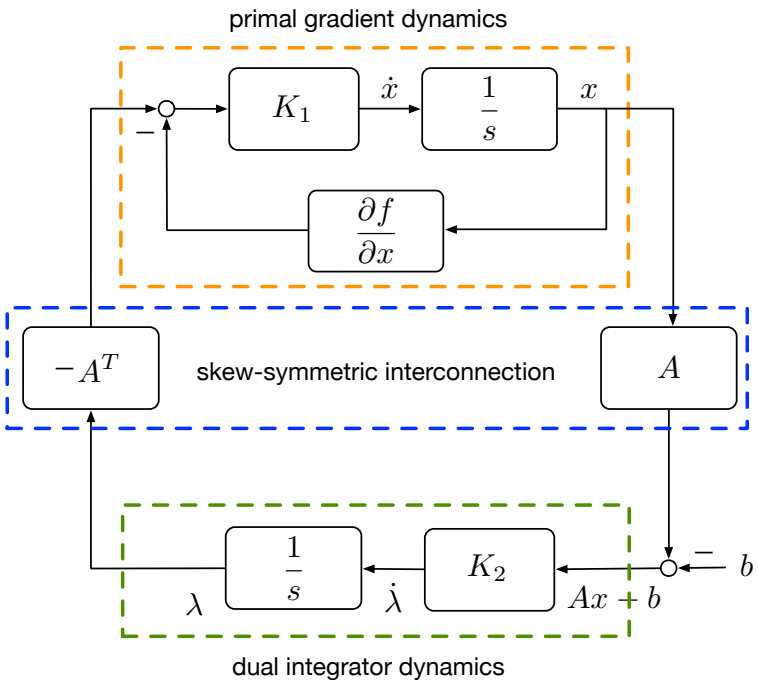


Figure 5: The block-diagram of the saddle-point flow (17) is composed of a skew-symmetric interconnection (through the matrices A and $-A^\top$) of the primal gradient dynamics (5) and an integrator for the dual dynamics integrating (and thus penalizing) the constraint violation: $\dot{\lambda} = Ax - b$.

1.8 Primal-Dual Algorithms with Augmented Lagrangians

To ensure convergence of saddle-point flow and to improve its transient performance, it is beneficial to re-consider the soft penalty method in (9) and define the *augmented Lagrangian function*

$$\mathcal{L}_{\text{aug}}(x,\lambda) = f(x) + \lambda^\top g(x) + \frac{\rho}{2}\,\|g(x)\|^2 = f(x) + \lambda^\top (Ax-b) + \frac{\rho}{2}\,(Ax-b)^\top (Ax-b)\,, \tag{18}$$

where $\rho > 0$. Note that the augmentation term $\rho\|g(x)\|^2$ evaluates to zero at optimality and merely serves the purpose to improve the algorithm performance by penalizing the constraint violation and thus adding (i) additional convexity as well as (ii) damping. Indeed, even if $f(x)$ was not strictly convex, the function $f(x) + \frac{\rho}{2}\,(Ax-b)^\top (Ax-b)$ may be strictly convex. Regarding additional damping, the saddle-point flow (17) applied to the augmented Lagrangian function (18) yields

$$\dot{x} = -K_1\,\frac{\partial}{\partial x}\mathcal{L}_{\text{aug}}(x,\lambda) = -K_1\,\frac{\partial f(x)}{\partial x} - K_1\,A^\top\lambda - \rho K_1\,A^\top (Ax-b) \tag{19a}$$

$$\dot{\lambda} = +K_2\,\frac{\partial}{\partial \lambda}\mathcal{L}_{\text{aug}}(x,\lambda) = K_2\,(Ax-b)\,. \tag{19b}$$

An interesting control-theoretic perspective on augmentation is revealed when analyzing the block-diagram of the augmented saddle-point flow (19); see Figure 6. In comparison to the regular saddle-point flow (17) shown in Figure 5 (where we understood the dual dynamics as integral control), the augmentation term $\frac{\rho}{2}\|g(x)\|^2$ induces an additional *proportional control* on the constraint violation. We leave the analysis of the augmented saddle-point flow (19) to the reader; see Exercise 8.

Remark 1.6 (Alternative augmentation)**.** *If $A = A^\top$ is a symmetric matrix and $b = \mathbb{0}_m$ so that $g(x) = Ax$, then another popular augmentation not affecting the saddle points is the following.*

$$\mathcal{L}_{aug,2}(x,\lambda) = f(x) + \lambda^\top g(x) + \frac{\rho}{2}\,x^\top g(x) = f(x) + \lambda^\top Ax + \frac{\rho}{2}\,x^\top Ax\,. \tag{20}$$

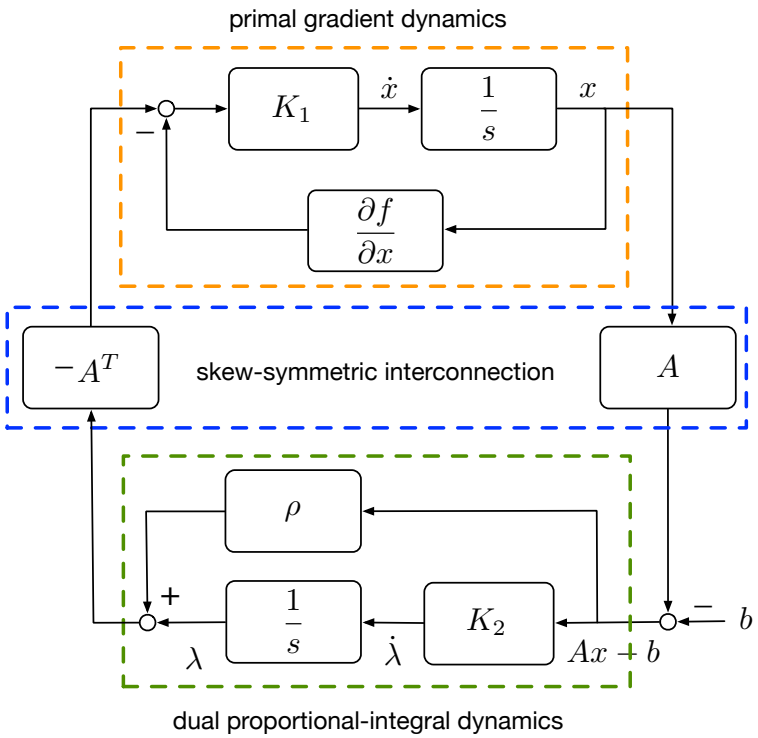


Figure 6: The block-diagram of the augmented saddle-point flow (19) is composed of a skew-symmetric interconnection (through the matrices A and $-A^\top$) of the primal gradient dynamics (5) and a proportional-integral control for the dual dynamics penalizing the constraint violation.

All previous comments concerning the first augmentation (18) *apply here as well. Note that a particular feature of the latter augmentation* (20) *is that* (20) *results in a sparse saddle-point flow (provided that A sparse) whereas the prior augmentation* (18) *gives a saddle-point flow involving the matrix $A^\top A$ which is typically more dense than A. We will revisit this theme in Section 2.2.*□

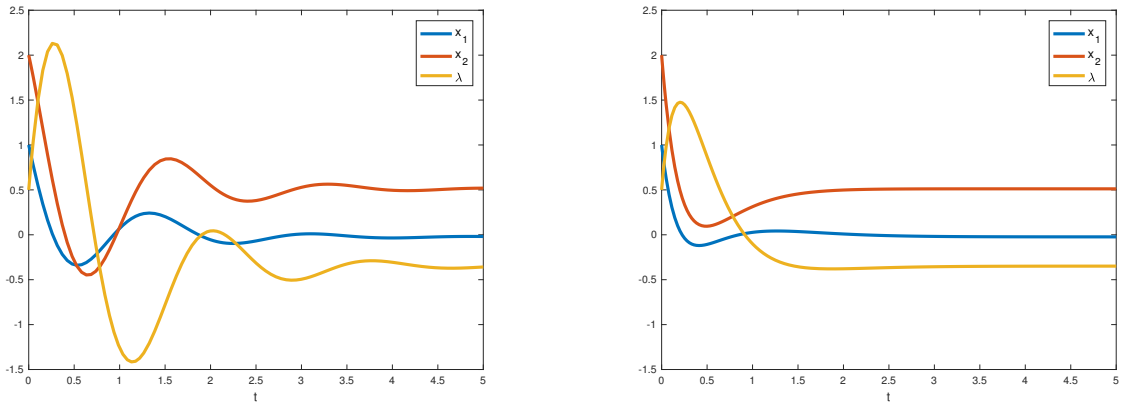
Example 2 ((Augmented) saddle-point flow of LQ program)**.** *For the LQ program* (11) *in Example 1 the saddle-point flow* (17) *is a linear system governed by the saddle matrix $\mathcal{A}$ in* (12)*:*

$$\begin{bmatrix} K_1^{-1}\dot{x} \\ K_2^{-1}\dot{\lambda} \end{bmatrix} = \underbrace{\begin{bmatrix} -P & -A^\top \\ A & \mathbb{0} \end{bmatrix}}_{=\mathcal{A}} \begin{bmatrix} x \\ \lambda \end{bmatrix} - \begin{bmatrix} -P\underline{x} \\ b \end{bmatrix} \tag{21}$$

A representative simulation is shown in Figure 7(a) for the gains $K_1 = I_2$ and $K_2 = 3$. The oscillations are due to the skew-symmetric off-diagonal elements of the saddle-matrix $\mathcal{A}$. A simulation of the augmented saddle-point flow (19) *is shown in Figure 7(b) with the gain $\rho = 1$. Observe the qualitative difference between the augmented and non-augmented flows. In particular, the augmentation induces an over-damped behavior and much faster convergence. Thus, due to their superior performance, augmented Lagrangians are generally preferred in saddle-point algorithms.* □

1.9 Dual Ascent Algorithm

Observe that the saddle-point flow (17) converges for any positive definite gain matrices K_1 and K_2. In the following, we consider the limit where the gain K_1 becomes arbitrary large (compared



(a) saddle-point flow (17)

(b) augmented saddle-point flow (19)

Figure 7: Trajectories of the saddle-point flow (17) and the augmented saddle-point flow (19).

to K_2) so that the primal dynamics (17a) converge instantaneously to their equilibria

$$\mathbb{0}_n = \frac{\partial \mathcal{L}(x,\lambda)}{\partial x} = \frac{\partial f(x)}{\partial x} + A^\top \lambda \,, \tag{22}$$

The equilibria (22) correspond to the tangency condition in the KKT equations (10). After solving the algebraic equations (22) for $x = x(\lambda)$ (recall from Exercise 1 that $\frac{\partial f(x)}{\partial x}$ is invertible for a strictly convex function), we are left with the dual gradient dynamics (17b). In summary, this scheme termed *dual gradient* or *dual ascent* may also be written as

$$x^\star = \operatorname{argmin}_{x\in\mathbb{R}^n} \mathcal{L}(x,\lambda) = \operatorname{argmin}_{x\in\mathbb{R}^n} \Big(f(x) + x^\top A^\top \lambda \Big) \tag{23a}$$

$$\dot\lambda = K \frac{\partial}{\partial \lambda} \mathcal{L}(x^\star,\lambda) = K \left(A x^\star(\lambda) - b \right), \tag{23b}$$

where $K \in \mathbb{R}^{m\times m}$ is positive definite, and we emphasized that the minimizer $x^\star = x^\star(\lambda)$ is a function of λ. If f is strictly convex, the minimizer $x^\star(\lambda)$ is obtained as the unique solution of (22).

The block-diagram of the dual ascent (23) in Figure 8 is composed of a primal minimization step interconnected with a dual integrator through a symmetric coupling. The dual ascent (23) converges under strict convexity and additional regularity assumptions on f and g. We refer to [2, Chapter 3] for a detailed analysis and consider the linearly-constrained quadratic program (11) here.

Theorem 1.7 (Convergence of dual ascent)**.** *Consider the LQ program* (11)*. Then each solution* $t \mapsto (x(t), \lambda(t))$ *of the dual ascent* (23) *converges to the set of saddle points* $(x^\star, \lambda^\star)$*.*

Proof. For the optimization problem (11), the minimizer $x^\star(\lambda)$ of $\mathcal{L}(x,\lambda)$ is found from (22) as

$$\mathbb{0}_n = P(x^\star(\lambda) - \underline{x}) + A^\top \lambda \quad \Leftrightarrow \quad x^\star(\lambda) = -P^{-1}A^\top \lambda + \underline{x} \,. \tag{24}$$

The set of associated optimal dual variables $\lambda^\star$ is found from the KKT conditions (10):

$$\mathbb{0}_m = A x^\star(\lambda) - b = -AP^{-1}A^\top \lambda^\star + A\underline{x} - b \,. \tag{25}$$

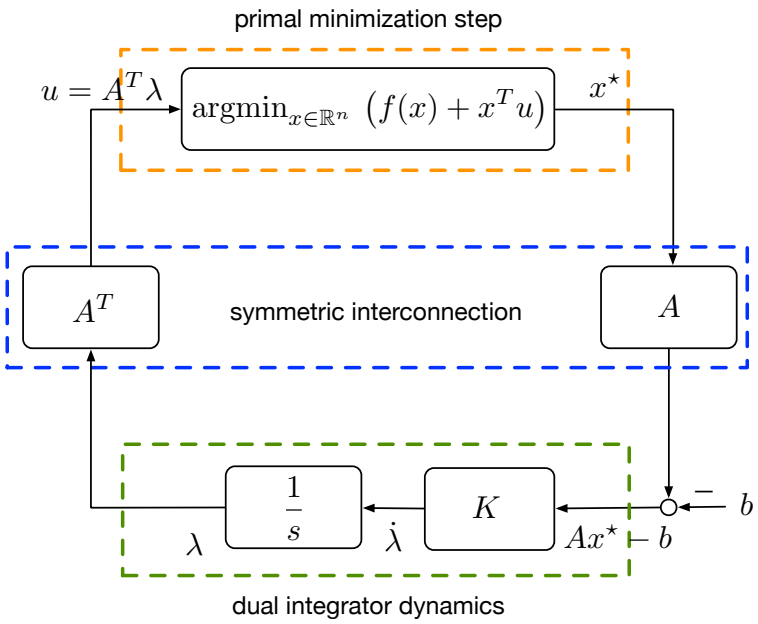


Figure 8: The block-diagram of the dual ascent algorithm (23) is composed of a symmetric interconnection (through the matrices A and $A^\top$) of the primal minimization step (23a) and an integrator for the dual dynamics integrating (and thus penalizing) the constraint violation: $\dot\lambda = Ax^\star - b$.

Observe that equation (25) admits a unique solution $\lambda^\star$ if and only if A has full rank. The associated dual gradient ascent dynamics (23b) can be written as

$$K^{-1}\dot\lambda = Ax^\star(\lambda) - b = -AP^{-1}A^\top\lambda + A\underline{x} - b = -AP^{-1}A^\top(\lambda - \lambda^\star)\,. \tag{26}$$

The matrix $AP^{-1}A^\top \in \mathbb{R}^{m\times m}$ is positive semidefinite. Consider the Lyapunov function

$$V(\lambda - \lambda^\star) = \frac{1}{2}(\lambda - \lambda^\star)^\top K^{-1}(\lambda - \lambda^\star)$$

with derivative along trajectories of (26) given by

$$\dot V(\lambda - \lambda^\star) = -(\lambda - \lambda^\star)^\top AP^{-1}A^\top(\lambda - \lambda^\star)\,.$$

If A has full rank m, then $AP^{-1}A^\top$ is positive definite, and the dual gradient ascent (26) is an asymptotically stable linear system. In this case, $\lambda(t)$ converges exponentially fast towards $\lambda^\star$, and $x(t) = x^\star(\lambda(t))$ converges exponentially towards $x^\star$. This concludes the proof if A has full rank. We leave the general proof in the rank-deficient case to the reader (see Exercise 9). □

We remark that the dual ascent (23) can also be defined for the augmented Lagrangian (18) (Exercise 10) as well as in discrete time (Exercise 11). An indicative simulation is shown in Figure 9.

2 Distributed Optimization

2.1 Distributed Optimization Setup

Consider a network of n agents that can perform local computation and communicate with another. The communication architecture is modeled by an undirected and connected graph with n nodes

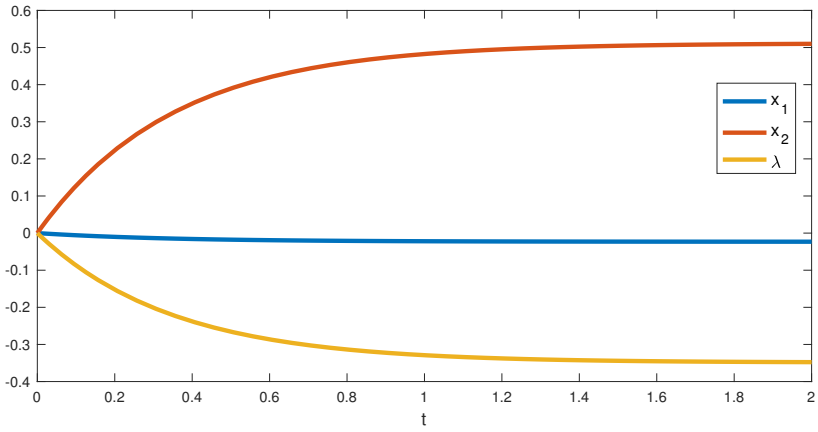


Figure 9: Dual ascent (23) applied to the LQ program from Example 1 with $K = 1$.

and m edges. The objective of the multi-agent network is to solve the optimization problem

$$\text{minimize}_{x\in\mathbb{R}} \; f(x) = \sum_{i=1}^{n} f_i(x), \tag{27}$$

where $f_i : \mathbb{R} \to \mathbb{R}$ is a *private* cost function known only to agent $i \in \{1, \dots, n\}$. We assume that each f_i is strictly convex and twice continuously differentiable. Note that each agent's cost is a function of the global variable x which is why the agents need to coordinate to solve the problem (27).

To find a distributed approach for solving the optimization problem (27), we associate a local estimate $y_i \in \mathbb{R}$ of the global variable $x \in \mathbb{R}$ to every agent and solve the *equivalent problem*

$$\text{minimize}_{y\in\mathbb{R}^n} \; \tilde{f}(y) = \sum_{i=1}^{n} f_i(y_i) \tag{28a}$$

$$\text{subject to } \mathcal{C}(y)\,, \tag{28b}$$

where $\mathcal{C}(y)$ is a *consensus constraint* that assures $y_i = y_j$ for all $i, j \in \{1, \dots, n\}$, that is, the local estimates of all processors coincide. Different choices for the consensus constraint are, for example,

$$\text{(i) Laplacian constraint:} \quad \mathbb{0}_n = Ly\,, \tag{29a}$$

$$\text{(ii) incidence constraint:} \quad \mathbb{0}_m = B^\top y\,, \tag{29b}$$

where $L \in \mathbb{R}^{n\times n}$ is a weighted Laplacian matrix and $B \in \mathbb{R}^{n\times n}$ is an incidence matrix associated with the communication graph among the processors. The different choices of consensus constraints $\mathcal{C}(y)$ are useful in different algorithmic applications to constrained optimization, and they give rise to fundamentally different computation and communication architectures as we will see below.

Example 3 (Quadratic distributed optimization)**.** *As a running example for distributed optimization, we consider the quadratic cost functions $f_i(x) = (x - \underline{x}_i)^\top P_i (x - \underline{x}_i)$, where $P_i > 0$ and $\underline{x}_i \in \mathbb{R}$ is the minimizer of the cost $f_i(x)$. Thus, the cost can be reformulated up to an additive constant as*

$$f(x) = \frac{1}{2}\sum_{i=1}^{n} (x - \underline{x}_i)^\top P_i (x - \underline{x}_i) = \frac{1}{2}\sum_{i=1}^{n} (x - x_{\text{w-avg}})^\top P_i (x - x_{\text{w-avg}}) + \mathcal{O}(x^0)\,, \tag{30}$$

where the weighted average $x_{\text{w-avg}} = (\sum_{i=1}^{n} P_i)^{-1} \sum_{i=1}^{n} P_i \underline{x}_i$ is the global minimizer of $f(x)$; see Exercise 14. In the following, we will frequently revisit the quadratic objective $f(x)$ either in its original form $\frac{1}{2}\sum_{i=1}^{n}(x - \underline{x}_i)^\top P_i (x - \underline{x}_i)$ or as $\frac{1}{2}\sum_{i=1}^{n}(x - x_{\text{w-avg}})^\top P_i (x - x_{\text{w-avg}})$. □

2.2 Distributed Primal-Dual Saddle-Point Algorithm

We begin our analysis of the the distributed optimization problem (28) by means of a primal-dual saddle-point algorithm from Section 1.7. We consider here the Laplacian-based consensus constraint (29a) and refer to Exercise 16 for an analysis of the incidence-based constraint (29b):

$$\text{minimize}_{y\in\mathbb{R}^n} \ \tilde{f}(y) = \sum\nolimits_{i=1}^{n} f_i(y_i) \tag{31a}$$

$$\text{subject to } Ly = \mathbb{0}_n \,. \tag{31b}$$

For problem (31) an augmented Lagrangian function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ of the form (20) is

$$\mathcal{L}(y, \lambda) = \tilde{f}(y) + \lambda^\top L y + \frac{1}{2} y^\top L y \,, \tag{32}$$

and the associated saddle-point flow (16) reads as

$$\dot{y} = -\frac{\partial \mathcal{L}(y,\lambda)}{\partial y} = -\frac{\partial}{\partial y}\tilde{f}(y) - Ly - L\lambda \,, \tag{33a}$$

$$\dot{\lambda} = +\frac{\partial \mathcal{L}(y,\lambda)}{\partial \lambda} = Ly \,, \tag{33b}$$

In components, for processor $i \in \{1, \dots, n\}$ the saddle-point flow (33) reads as

$$\dot{y}_i = -\frac{\partial}{\partial y_i} f_i(y_i) - \sum\nolimits_{j=1}^{n} a_{ij}(y_i - y_j) - \sum\nolimits_{j=1}^{n} a_{ij}(\lambda_i - \lambda_j),$$
$$\dot{\lambda}_i = \sum\nolimits_{j=1}^{n} a_{ij}(y_i - y_j) \,,$$

where $a_{ij} \geq 0$ are the (possibly non-binary) weights of the communication graph among the agents. The block-diagram of the saddle-point flow (33) is shown in Figure 10 including further gains for the augmentation, primal, and dual dynamics. The saddle-point dynamics (33) can be implemented in a distributed processor network using only local knowledge of $f_i(y_i)$, local computation, nearest-neighbor communication, and after time-discretizing the dynamics (33) (see Exercise 15). The reasons for preferring the augmentation (20) over the standard choice (18) is that the augmentation (18) results in the term $L^\top L y$ in (33a) (instead of Ly) hindering a distributed implementation.

The saddle points of the augmented Lagrangian (32) and the equilibria of the saddle-point flow (33) are characterized by the following lemma whose proof is left to the reader in Exercise 13.

Lemma 2.1 (Properties of saddle points)**.** *Let $L = L^\top \in \mathbb{R}^{n\times n}$ be a symmetric Laplacian associated to an undirected, connected, and weighted graph. Consider the augmented Lagrangian function (32), where each f_i is strictly convex and twice continuously differentiable for all $i \in \{1, \dots, n\}$. Then*

1. *if $(y^\star, \lambda^\star) \in \mathbb{R}^n \times \mathbb{R}^n$ is a saddle point of $\mathcal{L}$, then so is $(y^\star, \lambda^\star + \alpha \mathbb{1}_n)$ for any $\alpha \in \mathbb{R}$;*
2. *if $(y^\star, \lambda^\star) \in \mathbb{R}^n \times \mathbb{R}^n$ is a saddle point of $\mathcal{L}$, then $y^\star = x^\star \mathbb{1}_n$ where $x^\star \in \mathbb{R}$ is a solution of the original optimization problem (27); and*
3. *if $x^\star \in \mathbb{R}$ is a solution of the original optimization problem (27), then there are $\lambda^\star \in \mathbb{R}^n$ and $y^\star = x^\star \mathbb{1}_n$ satisfying $L\lambda^\star + \frac{\partial}{\partial y}\tilde{f}(y^\star) = \mathbb{0}_n$ so that $(y^\star, \lambda^\star)$ is a saddle point of $\mathcal{L}$.*

The following result characterizes the convergence of the distributed saddle point flow (33).

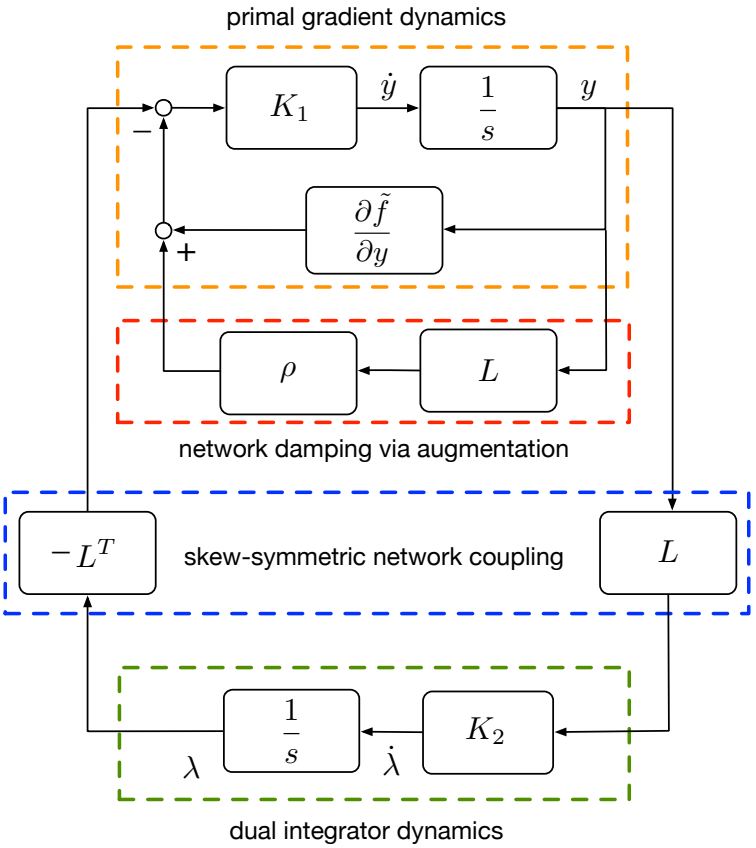


Figure 10: Block-diagram of the distributed saddle-point flow (33).

Theorem 2.2 (Convergence of distributed saddle-point flow)**.** *Let $\tilde{f} : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable and strictly convex. Then each solution $t \mapsto (y(t), \lambda(t))$ of the distributed saddle-point flow (33) converges to a saddle point $(y^\star, \lambda^\star)$ uniquely characterized by*

1. *$y^\star = x^\star \mathbb{1}_n$ where $x^\star \in \mathbb{R}$ is the solution of the original optimization problem (27); and*
2. *$\lambda^\star = \bar{\lambda} + average(\lambda_0)\mathbb{1}_n$ where $\bar{\lambda} \perp \mathbb{1}_n$ satisfies $L\bar{\lambda} + \frac{\partial}{\partial y}\tilde{f}(y^\star) = \mathbb{0}_n$.*

Proof. The general convergence to the set of saddle points $(y^\star, \lambda^\star)$ follows from Theorem 1.5. From Lemma 2.1, the saddle points $(y^\star, \lambda^\star)$ are identified as $y^\star = x^\star \mathbb{1}_n$ where $x^\star \in \mathbb{R}$ is the solution of the original optimization problem (27) and $\lambda^\star = \bar{\lambda} + \alpha \mathbb{1}_n$ with $\alpha \in \mathbb{R}$ and $\bar{\lambda} \perp \mathbb{1}_n$ satisfying $L\bar{\lambda} + \frac{\partial}{\partial y}\tilde{f}(y^\star) = \mathbb{0}_n$. From the dual dynamics (33b), we conclude that $\mathbb{1}_n^\top \dot{\lambda} = 0$ and thus average$(\lambda(t))$ = average(λ_0) for all $t \geq 0$. It follows that the saddle-point dynamics (33) converge to a unique saddle point $(y^\star, \lambda^\star)$ satisfying $y^\star = x^\star \mathbb{1}_n$ and $\lambda^\star = \bar{\lambda} + \text{average}(\lambda_0)\mathbb{1}_n$. □

Example 4 (Distributed LQ saddle point optimization)**.** *As shown in [22, 8, 6, 4], the distributed saddle-point flow (33) is a versatile and robust setup that extends to directed graphs and non-differentiable objective functions. In the following, we restrict ourselves to the quadratic cost functions discussed in Example 3. In this case, the saddle-point flow (33) reduces to the linear system*

$$\begin{bmatrix} \dot{\tilde{y}} \\ \dot{\lambda} \end{bmatrix} = \underbrace{\begin{bmatrix} -P - L & -L \\ L & \mathbb{0}_{n\times n} \end{bmatrix}}_{=\mathcal{A}} \begin{bmatrix} \tilde{y} \\ \lambda \end{bmatrix}, \tag{34}$$

where $\tilde{y} = y - x_{\text{w-avg}}\mathbb{1}_n$ and $P = diag(\{P_i\}_{i\in\{1,\dots,n\}})$. In the following, we analyze the kernel of $\mathcal{A}$:

$$\begin{aligned} \begin{bmatrix} \mathbb{0}_n \\ \mathbb{0}_n \end{bmatrix} = \begin{bmatrix} -P - L & -L \\ L & \mathbb{0}_{n\times n} \end{bmatrix} \begin{bmatrix} \tilde{y} \\ \lambda \end{bmatrix} \quad &\implies \quad (P+L)\tilde{y} + L\lambda = \mathbb{0}_n \text{ and } L\tilde{y} = \mathbb{0}_n \quad \implies \quad \tilde{y} \in span(\mathbb{1}_n) \\ &\implies \quad \text{multiplying } (P+L)\tilde{y} + L\lambda = \mathbb{0}_n \text{ by } \tilde{y}^\top\text{: } \tilde{y}^\top P \tilde{y} = \mathbb{0}_n \\ &\implies \quad \tilde{y} = \mathbb{0}_n \text{ and } \lambda = \alpha\, \mathbb{1}_n\,, \text{ where } \alpha \in \mathbb{R}. \end{aligned}$$

Thus, the saddle matrix $\mathcal{A}$ has a zero eigenvalue with multiplicity 1, and the associated eigenvector $\begin{bmatrix} \mathbb{0}_n^\top & \mathbb{1}_n^\top \end{bmatrix}^\top$ corresponds to the set of saddle points. From Lemma 1.4, we conclude that the remaining eigenvalues must be asymptotically stable, and thus (34) converges to the set of saddle points. In the chosen error coordinates, we have that $\tilde{y}(t \to \infty) = \mathbb{0}_n$ and $\lambda(t \to \infty) = average(\lambda_0)\mathbb{1}_n$. □

2.3 Distributed Dual Ascent

In the following, we seek to apply the dual ascent (23) to distributed optimization.

As in Section 1.9, we restrict ourselves to the LQ setup and adopt the quadratic cost functions (30). Observe from the block-diagram in Figure 8 that the dual ascent (23) can be implemented in a distributed fashion as long as the constraint matrix (the A matrix in Figure 8) and its transpose are sparse and encode the information exchange encoded in the communication graph. We can thus resort to either the Laplacian or the incidence consensus constraints in (29). In the following, we adopt the incidence consensus constraint (29b) giving rise to the distributed optimization problem

$$\text{minimize}_{y\in\mathbb{R}^n} \; \tilde{f}(y) = \frac{1}{2}\sum\nolimits_{i=1}^{n} (y_i - \underline{x}_i)^\top P_i (y_i - \underline{x}_i) \tag{35a}$$

$$\text{subject to } B^\top y = \mathbb{0}_m\,. \tag{35b}$$

In this case, the primal minimization step (24) reduces to solving the equation

$$\mathbb{0}_n = P(y^\star(\lambda) - \underline{x}) + B\lambda \quad \Leftrightarrow \quad y^\star(\lambda) = -P^{-1}B\lambda + \underline{x}\,, \tag{36}$$

where $P = \mathrm{diag}(\{P_i\}_{i\in\{1,\dots,n\}})$ and $\underline{x} = [\underline{x}_1 \ \dots \ \underline{x}_n]^\top$. The minimization (36) can be performed locally at each agent provided that it has access to the multipliers λ associated to the adjacent communication links. The multipliers are updated via the dual gradient step (26), that is,

$$K^{-1}\dot{\lambda} = B^\top y^\star(\lambda) = -B^\top P^{-1}B\lambda + B^\top \underline{x} \tag{37}$$

by simply taking differences of the values $y^\star(\lambda)$ at the neighboring agent. In short, the incidence-based dual ascent (36)-(37) can be implemented in a distributed fashion amongst the agents provided that every communication link has an associated multiplier; see Figure 11. This alternation of updates at agents and communication links is a general feature of the incidence constraint (29b) and also occurs for incidence-based saddle-point flows (see Exercise 16). In a real-world distributed computing setting, the nodal computation could be performed by individual processors whereas the edge computations are performed by routers linking these computers.

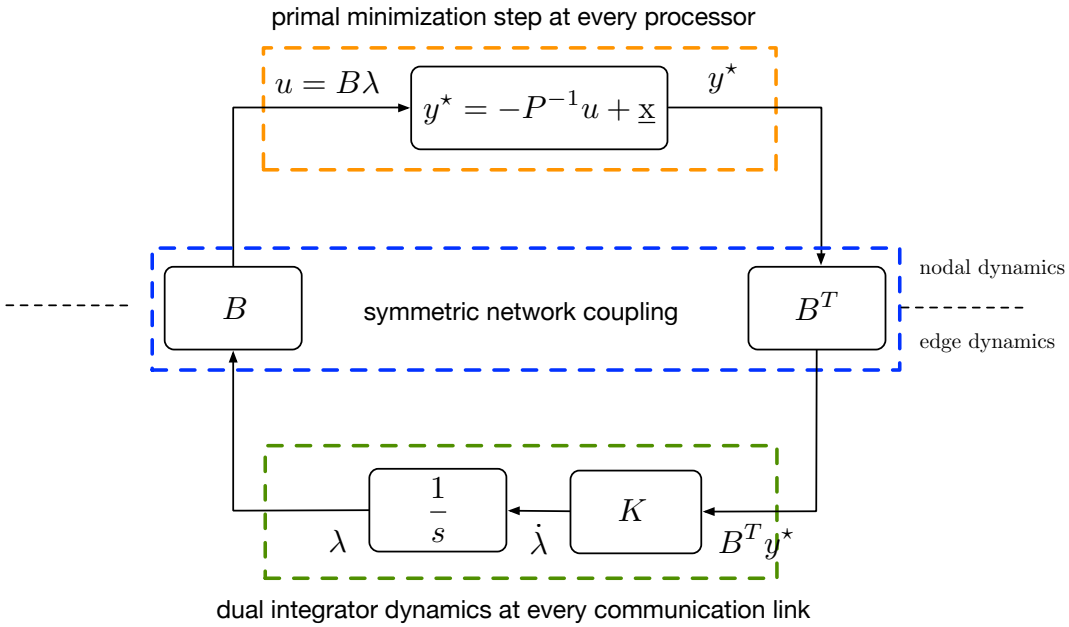


Figure 11: Block-diagram of the incidence-based dual ascent (36)-(37).

Theorem 2.3 (Convergence of distributed dual ascent)**.** *Consider the LQ program (35). Then each solution $t \mapsto (y(t), \lambda(t))$ of the distributed dual ascent (36)-(37) converges to a saddle point $(y^\star, \lambda^\star)$ uniquely characterized by*

1. *$y^\star = x_{\text{w-avg}}\mathbb{1}_n$ where $x_{\text{w-avg}} = (\sum_{i=1}^n P_i)^{-1}\sum_{i=1}^n P_i\underline{x}_i$ is the solution of the original optimization problem with the quadratic cost (30); and*
2. *$\lambda^\star = \bar{\lambda} + \tilde{\lambda}$, where $\bar{\lambda} \in \mathrm{image}(B^\top)$ satisfies $\mathbb{0}_n = B\bar{\lambda} + P(y^\star - \underline{x})$ and $\tilde{\lambda} \in \mathrm{kernel}(B)$.*

Proof. The closed-loop dual ascent (36)-(37) reads as

$$K^{-1}\dot{\lambda} = -B^\top P^{-1}B\lambda + B^\top \underline{x} = -L_{\text{edge}}(\lambda - \lambda^\star)\,, \tag{38}$$

where $L_{\text{edge}} = B^\top P^{-1} B$ is an edge Laplacian matrix (see Exercise E8.2), and we used the fact that the optimal dual variable $\lambda^\star$ satisfies the KKT conditions (10):

$$\mathbb{0}_m = B^\top y^\star(\lambda) = -B^\top P^{-1} B \lambda^\star + B^\top \underline{x}. \tag{39}$$

By Theorem 1.7, the final update equation (38) converges exponentially to an optimal dual variable $\lambda^\star$, and the primal minimizer $y^\star(\lambda(t))$ from (36) converges exponentially to the global minimizer.

From the KKT condition (39), we have that $y^\star = c \cdot \mathbb{1}_n$ for some $c \in \mathbb{R}$. When we multiply the other KKT condition (36) from the left by $\mathbb{1}_n^\top$, we obtain $0 = \sum_{i=1}^n P_i\, c - \sum_{i=1}^n P_i \underline{x}_i$, that is, $c = x_{\text{w-avg}}$, and $y^\star = x_{\text{w-avg}} \mathbb{1}_n$. Concerning the optimal dual variable $\lambda^\star$, the KKT condition (36) shows that $\lambda^\star = \bar{\lambda} + \tilde{\lambda}$, where $\tilde{\lambda} \in \text{kernel}(B)$ and $\bar{\lambda} \in \text{image}(B^\top)$ satisfies $\mathbb{0}_n = B\bar{\lambda} + P(y^\star - \underline{x})$. To further investigate $\tilde{\lambda} \in \text{kernel}(B)$, consider any vector $v \in \text{kernel}(B)$ and multiply (38) by $v^\top K$ to obtain $v^\top \dot{\lambda} = 0$, that is, $v^\top \lambda(t) = v^\top \tilde{\lambda}(t) = v^\top \tilde{\lambda}(0) = v^\top \lambda(0)$ for all $t \geq 0$. Thus, any vector $\tilde{\lambda}(0) \in \text{kernel}(B)$ remains constant under the dynamics (38). Hence, the final saddle point $(y^\star, \lambda^\star)$ is uniquely characterized by $\mathbb{0}_n = B\bar{\lambda} + P(y^\star - \underline{x})$ and the initial condition $\tilde{\lambda}(0) \in \text{kernel}(B)$. □

We refer to Exercises 17 and 18 for a Laplacian-based and discrete-time implementations.

2.4 Distributed Gradient Descent

Recall the block-diagram of the distributed saddle-point algorithm (33) in Figure 10. In the following, consider the case of taking the dual gain K_2 in Figure 10 to zero. As a result, Figure 10 reduces to Figure 12, and the primal update reads as a distributed gradient flow with Laplacian coupling:

$$K_1 \dot{y} = -\frac{\partial \tilde{f}(y)}{\partial y} - \rho L y\,. \tag{40}$$

Inspired from the soft-penalty method (9), one may also view the distributed gradient flow (40)

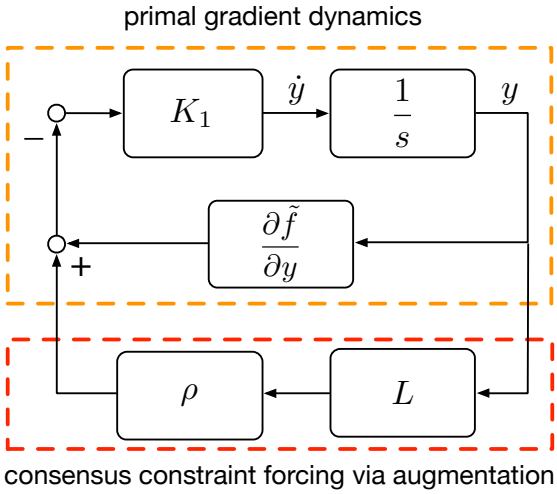


Figure 12: This figure shows the block-diagram of the distributed gradient flow (40). It is instructive to compare this block diagram to that of the distributed saddle-point flow (33) in Figure 10

with Laplacian coupling as the regular gradient flow of the optimization problem

$$\text{minimize}_{y \in \mathbb{R}^n} \ \sum_{i=1}^n \tilde{f}_i(y_i) + \frac{\rho}{2}\, y^\top L y\,, \tag{41}$$

where the soft-penalty $\frac{\rho}{2}\, y^\top L y$ is tasked with enforcing a Laplacian consensus-constraint. Observe that the minimizer of (41) is generally different from that of the distributed optimization problem (31). Nevertheless, it can be shown that the algorithm (40) converges to the correct solution of the distributed optimization problem (31) provided that the gains are time-varying. This idea has been first proposed in [14] in the discrete-time case dating back to earlier work in [21], and variations of the continuous flow (40) have later been studied in continuous time [13, 12, 10].

We follow ideas in [10, 13] and present an independent analysis. Consider the time-varying flow

$$\dot{y} = -\alpha(t)\, \frac{\partial \tilde{f}(y)}{\partial y} - L y\,, \tag{42}$$

where the time-varying positive gain $\alpha : \mathbb{R}_{\geq} \to \mathbb{R}_{>0}$ satisfies the persistence condition

$$\alpha(t) > 0\,, \quad \int_0^\infty \alpha(\tau)\, d\tau = \infty\,, \text{ and } \quad \lim_{t\to\infty} \alpha(t) = 0\,, \tag{43}$$

that is, over time the Laplacian coupling in the distributed gradient flow (42) dominates the gradient descent with a decaying yet non-integrable gain $\alpha(t)$, e.g., $\alpha(t) = (t+0.1)^{-1}$.

Theorem 2.4 (Convergence of distributed gradient flow)**.** *Consider the distributed gradient flow (42) where $\alpha : \mathbb{R}_{\geq} \to \mathbb{R}_{>0}$ satisfies the persistence condition (43), and $\tilde{f} : \mathbb{R}^n \to \mathbb{R}$ is assumed to be twice continuously differentiable, strictly convex, and radially unbounded. Then each solution $t \mapsto y(t)$ of the distributed gradient flow (42) converges to the unique optimizer $y^\star = x^\star \mathbb{1}_n$.*

Proof. The proof consists of three subsequent analysis steps:

Step 1: boundedness of the state: Consider the Lyapunov function $V(y) = \frac{1}{2}\|y - y^\star\|_2^2 = \frac{1}{2}\|y - x^\star \mathbb{1}_n\|_2^2$ whose derivative along trajectories of (42) is

$$\begin{aligned}
\dot{V}(y) &= -\alpha(t)\, (y - y^\star)^\top \frac{\partial \tilde{f}(y)}{\partial y} - (y - y^\star)^\top L y \\
&= -\alpha(t)\, \Big(\tilde{f}(y) - \tilde{f}(y^\star)\Big) - (y - y^\star)^\top L\, (y - y^\star) \leq 0\,,
\end{aligned}$$

where we leveraged the underestimator property (2) and the fact that $L y^\star = L x^\star \mathbb{1}_n = \mathbb{0}_n$. Thus, the state y is bounded, and the distance to the optimizer is non-increasing: $\|y(t) - y^\star\| \leq \|y_0 - y^\star\|$.

Step 2: asymptotic consensus: Consider the change of coordinates

$$\begin{bmatrix} \delta \\ \xi \end{bmatrix} = \begin{bmatrix} U^\top \\ \mathbb{1}_n^\top / n \end{bmatrix} y \quad \Leftrightarrow \quad y = \begin{bmatrix} U & \mathbb{1}_n \end{bmatrix} \begin{bmatrix} \delta \\ \xi \end{bmatrix}\,,$$

where the matrix $U \in \mathbb{R}^{n\times(n-1)}$ collects the $n-1$ Laplacian eigenvectors orthogonal to $\mathbb{1}_n$ and with associated eigenvalues $\lambda_2, \dots, \lambda_n > 0$. In these coordinates, the dynamics (42) read as

$$\frac{d}{dt}\begin{bmatrix} \delta \\ \xi \end{bmatrix} = \begin{bmatrix} U^\top L U & \mathbb{0}_{n-1} \\ \mathbb{0}_{n-1}^\top & 0 \end{bmatrix} \begin{bmatrix} \delta \\ \xi \end{bmatrix} - \alpha(t) \begin{bmatrix} U^\top \frac{\partial \tilde{f}(y)}{\partial y} \\ \frac{1}{n}\mathbb{1}_n^\top \frac{\partial \tilde{f}(y)}{\partial y} \end{bmatrix}\,.$$

In the following, we focus on the first set of equations, which read as

$$\dot{\delta} = U^\top L U \delta - \alpha(t)\, U^\top \frac{\partial \tilde{f}(y)}{\partial y} = -\mathrm{diag}(\lambda_2, \dots, \lambda_n)\delta + \eta(t)\,,$$

where we defined $\eta(t) = -\alpha(t)\, U^\top \frac{\partial \tilde{f}(y)}{\partial y}$. Note that $\eta(t)$ is a bounded signal (since y is bounded) that is decaying to zero (since $\alpha(t)$ is decaying to zero). Hence, the δ-dynamics are an exponentially stable linear system subject to the bounded and convergent input $\eta(t)$. Thus, the δ-coordinate converges asymptotically to zero, and consensus $y(t \to \infty) \in \operatorname{span}(\mathbb{1}_n)$ is achieved asymptotically.

Step 3: asymptotic optimality: The remaining coordinate $\xi = \text{average}(y)$ has the dynamics

$$\dot{\xi} = -\frac{\alpha(t)}{n}\, \mathbb{1}_n^\top \frac{\partial \tilde{f}(y)}{\partial y} = -\frac{\alpha(t)}{n} \sum\nolimits_{i=1}^n \frac{\partial f_i}{\partial y}(y_i) = -\frac{\alpha(t)}{n} \sum\nolimits_{i=1}^n \frac{\partial f_i}{\partial y}\big((U\delta)_i + \xi\big)\,,$$

where we used $y_i = (U\delta)_i + \xi$. Consider the time-scaling $\tau = \int_0^t \alpha(\sigma)\, d\sigma$ so that $\frac{d\tau}{dt} = \alpha(t)$. Note that $\alpha(t) > 0$ assures that τ is strictly monotonically increasing in t, and the integral persistence condition (43) guarantees surjectiveness: $t \in [0, \infty]$ is mapped to $\tau \in [0, \infty]$. Hence, the change of time-scales is invertible. In the τ-time-scale, the ξ-dynamics read as

$$\frac{d}{d\tau}\xi = -\frac{1}{n} \sum\nolimits_{i=1}^n \frac{\partial f_i}{\partial y}((U\delta)_i + \xi)$$

Consider the Lyapunov function $W(\xi) = \frac{1}{2}(\xi - \xi^\star)^2$, where $\xi^\star = x^\star$ is the global minimizer of $f(x) = \sum_{i=1}^n f_i(x)$. The derivative of $W(x)$ is

$$\frac{d}{d\tau}W(\xi) = -(\xi - \xi^\star) \cdot \sum\nolimits_{i=1}^n \frac{\partial f_i}{\partial y}(\xi) + (\xi^\star - \xi) \cdot \sum\nolimits_{i=1}^n \left(\frac{\partial f_i}{\partial y}((U\delta)_i + \xi) - \frac{\partial f_i}{\partial y}(\xi) \right)$$

By the underestimator property (2), the first term of $\dot{W}(\xi)$ is upper-bounded by $f(\xi^\star) - f(\xi)$. Since the state is bounded (see Step 1), the second term can be upper-bounded as

$$(\xi^\star - \xi) \cdot \sum\nolimits_{i=1}^n \left(\frac{\partial f_i}{\partial y}((U\delta)_i + \xi) - \frac{\partial f_i}{\partial y}(\xi) \right) \leq |\xi^\star - \xi| \cdot \sum\nolimits_{i=1}^n L_i \cdot |(U\delta)_i|$$

where $L_i = \max_{\|y - y^\star\| \leq \|y_0 - y^\star\|} \operatorname{Hess} f_i(y)$ is the maximal Lipschitz constant of $\frac{\partial f_i}{\partial y}$ evaluated in the compact invariant set $\{y \in \mathbb{R}^n \mid \|y - y^\star\| \leq \|y_0 - y^\star\|\}$ calculated in Step 1. By Step 2, for each $\varepsilon \geq 0$, there is a τ_ε so that for all $\tau \geq \tau_\varepsilon$ we have $\sum_{i=1}^n L_i \cdot |(U\delta)_i| \leq \varepsilon$. In summary, for each $\tau \geq \tau_\varepsilon$

$$\frac{d}{d\tau}W(\xi) \leq f(\xi^\star) - f(\xi) + \varepsilon \cdot |\xi^\star - \xi| \leq 0\,.$$

Due to radial unboundedness and strict convexity of $\tilde{f}$ (and thus also of f), we have that there is $|\xi_\varepsilon|$ so that $\frac{d}{d\tau}W(\xi)$ is strictly negative for sufficiently large $|\xi - \xi^\star| \geq |\xi_\varepsilon|$ and thus $W(\xi(\tau \to \infty)) \leq W(\xi_\varepsilon)$. Moreover, $\xi_\varepsilon \to 0$ as $\varepsilon \to 0$. Hence, for each $\tilde{\varepsilon} \geq 0$, there is ε sufficiently small so that for $\tau \geq \tau_\varepsilon$, $W(\xi(\tau)) \leq \tilde{\varepsilon}$. Spoken differently, $W(\xi(\tau))$ converges to 0. Equivalently, $\xi(\tau)$ converges to the optimal value $\xi^\star$ as $\tau \to \infty$ and so does $\xi(t)$ as $t \to \infty$. □

3 Exercises

1. Exercise **(One-to-one property of the gradient of a strictly convex function)**: Show that if $f : \mathbb{R}^n \to \mathbb{R}$ is strictly convex and continuously differentiable, then the gradient $\frac{\partial f}{\partial x} : \mathbb{R}^n \to \mathbb{R}^n$ is a one-to-one mapping, that is, for all $x_1, x_2 \in \mathbb{R}^n$

$$\frac{\partial f}{\partial x}(x_1) = \frac{\partial f}{\partial x}(x_2)$$

implies $x_1 = x_2$. Hence, the function $\frac{\partial f(x)}{\partial x}$ is invertible on its range.

2. Exercise **(Convergence rate of gradient algorithms)**: Consider the gradient flow

$$\dot{x} = -\frac{\partial f(x)}{\partial x}\,.$$

Show the following convergence rate measured in terms of sub-optimality:

$$f(x(t)) - f(x^\star) \leq \frac{\|x_0 - x^\star\|^2}{2\,t}\,.$$

Hint: Consider the following function and its derivative to estimate the convergence rate:

$$\mathcal{E}(t) = t\,(f(x(t)) - f(x^\star)) + \frac{1}{2}\,\|x(t) - x^\star\|^2\,.$$

3. Exercise **(Gradient flows with gains)**: Consider the weighted negative gradient flow

$$\dot{x} = -K\,\frac{\partial f(x)}{\partial x}\,,$$

where $K \in \mathbb{R}^{n\times n}$ is a general positive definite matrix and f is a continuously differentiable and strictly convex function. Your tasks are as follows:

   (a) Show that the the weighted negative gradient flow converges to the global minimizer $x^\star$.

   (b) Show that the convergence estimate in Exercise 2 can be made arbitrary fast by picking K appropriately.

4. Exercise **(Convergence of discrete-time gradient descent)**: Consider the discrete-time negative gradient flow

$$x^+ = x - \varepsilon\,\frac{\partial f(x)}{\partial x}\,,$$

where f is a strictly convex and Lipschtiz-continuously differentiable function, that is, there is $L > 0$ so that

$$\left|\frac{\partial f}{\partial x}(y) - \frac{\partial f}{\partial x}(z)\right| \leq L|y - z| \qquad \text{for all } y, z \in \mathbb{R}^n\,.$$

Show that the time-discrete negative gradient flow converges to the strict global optimizer $x^\star$ of f if

$$0 < \varepsilon < \frac{2}{L}\,.$$

Hint: If you cannot prove the above convergence condition, then try to show it for the quadratic objective function $f(x) = x^\top P x$, where $P \in \mathbb{R}^{n\times n}$ is positive definite.

5. Exercise **(Acceleration of gradient algorithms)**: Consider the so-called *Nesterov-accelerated gradient flow* [20]
$$\ddot{x} + \frac{3}{t}\dot{x} = -\frac{\partial f(x)}{\partial x}\,.$$
Show the following convergence rate measured in terms of sub-optimality:
$$f(x(t)) - f(x^\star) \leq \frac{2\|x_0 - x^\star\|^2}{t^2}\,.$$
Hint: Consider the following function and its derivative to estimate the convergence rate:
$$\mathcal{E}(t) = t^2\left(f(x(t)) - f(x^\star)\right) + 2\,\|x(t) + t\dot{x}(t)/2 - x^\star\|^2\,.$$

6. Exercise **(Convergence of saddle-point flow)**: Prove Theorem 1.5.

7. Exercise **(Saddle-point flow with gains)**: Prove that the saddle-point flow (17) with positive definite gain matrices K_1, K_2 converges to the set of saddle points.

8. Exercise **(Convergence of augmented saddle-point flow)**: Prove that the augmented saddle-point flow (19) with positive definite gain matrices K_1, K_2 converges to the set of saddle points.

9. Exercise **(Proof of Theorem 1.7 in rank- deficient case)**: Complete the proof of Theorem 1.7 in case that $A \in \mathbb{R}^{m\times n}$ does not have full rank.

10. Exercise **(Dual ascent with augmented Lagrangian)**: Consider the linearly-constrained quadratic problem (11) and its augmented Lagrangian as defined in (18). Show that the dual ascent (23) applied to to the augmented Lagrangian converges to the set of saddle points.

11. Exercise **(Dual ascent in discrete time)**: Consider the linearly-constrained quadratic problem (11), and the discrete-time dual ascent algorithm
$$x^\star = \operatorname{argmin}_{x\in\mathbb{R}^n} \mathcal{L}(x,\lambda) \tag{44a}$$
$$\lambda^+ = \lambda + \varepsilon\,\frac{\partial}{\partial\lambda}\mathcal{L}(x^\star,\lambda)\,, \tag{44b}$$
where $\varepsilon > 0$ is a sufficiently small step-size. Give conditions on the step-size $\varepsilon > 0$ so that the discrete-time dual ascent (44) converges.

12. Exercise **(Spectrum of saddle matrices)**: Prove Lemma 1.4.

13. Exercise **(Properties of saddle points)**: Prove Lemma 2.1.

14. Exercise **(Centralized formulation of sum-of-squares cost)**: Consider a distributed optimization problem with n agents, where the cost function $f_i(x)$ of each agent $i \in \{1,\dots,n\}$ is described by $f_i(x) = (x - \underline{x}_i)^\top P_i (x - \underline{x}_i)$, where $P_i > 0$ and $\underline{x}_i \in \mathbb{R}$ is the minimizer of the cost $f_i(x)$. Consider the joint sum-of-squares cost function
$$f_{\text{sos}}(x) = \sum\nolimits_{i=1}^{n} (x - \underline{x}_i)^\top P_i (x - \underline{x}_i)\,.$$
Calculate the global minimizer $x^\star$ of $f_{\text{sos}}(x)$, and show that the sum-of-squares cost $f_{\text{sos}}(x)$ is, up to constant terms, equivalent to the centralized cost function
$$f_{\text{centralized}}(x) = (x - x^\star)^\top \left(\sum\nolimits_{i=1}^{n} P_i\right)(x - x^\star)\,.$$

15. Exercise **(Discrete-time saddle-point algorithm for distributed optimization)**: As in Example 3, consider the distributed optimization problem (31) with the quadratic local cost functions from Exercise 14 and with the Laplacian constraint (29a) corresponding to a connected and undirected network:

$$\text{minimize}_{y\in\mathbb{R}^n}\ \tilde{f}(y) = \frac{1}{2}\sum\nolimits_{i=1}^{n}(y_i-\underline{x}_i)^\top P_i(y_i-\underline{x}_i) \tag{45a}$$

$$\text{subject to } Ly = \mathbb{0}_n\,, \tag{45b}$$

(a) Formulate the augmented Lagrangian $\mathcal{L}(y,\lambda)$ in (32) as well as the KKT conditions (10) for this problem. Show that a generic pair $(y^\star,\lambda^\star)$ is a solution of the KKT system if and only if $y^\star = x^\star\mathbb{1}_n$ and $\lambda^\star = \bar{\lambda} + \alpha\mathbb{1}_n$ where $x^\star\in\mathbb{R}$ is unique and $\alpha\in\mathbb{R}$ is not determined uniquely.

(b) Consider a discrete-time distributed saddle-point algorithm obtained by an Euler discretization of the continuous-time dynamics (33) with sufficiently small step-size $\tau > 0$:

$$y^+ = y - \tau\frac{\partial\mathcal{L}(y,\lambda)}{\partial y}\,, \tag{46a}$$

$$\lambda^+ = \lambda + \tau\frac{\partial\mathcal{L}(y,\lambda)}{\partial\lambda}\,. \tag{46b}$$

Argue why the algorithm (46) can be implemented in a distributed fashion. Show that, if the algorithm (46) converges, then it converges to a solution $(y^\star,\lambda^\star)$ of Problem (45).

(c) Define the error vector of the algorithm (46) by

$$e(t) = \begin{bmatrix} y(t)-y^\star \\ \lambda(t)-\lambda^\star \end{bmatrix}.$$

Find the error dynamics, that is, the matrix G such that $e^+ = Ge$.

(d) Show that, for $\tau > 0$ small enough, if μ is an eigenvalue of G then either $\mu = 1$ or $|\mu| < 1$ and that $\begin{bmatrix}\mathbb{0}_n \\ \alpha\mathbb{1}_n\end{bmatrix}$ is the only eigenvector relative to the eigenvalue $\mu = 1$. Find an expression for α. Use these results to study the convergence properties of the distributed algorithm (46). Will $y(t)\to y^\star$ as $t\to\infty$?

16. Exercise **(Primal-Dual Consensus Optimization with Incidence Constraint)**:

As in Example 3, consider the distributed optimization problem (28) with the quadratic local cost functions from Exercise 14 and with the incidence constraint (29b) corresponding to a connected and undirected network:

$$\text{minimize}_{y\in\mathbb{R}^n}\ \tilde{f}(y) = \frac{1}{2}\sum\nolimits_{i=1}^{n}(y_i-\underline{x}_i)^\top P_i(y_i-\underline{x}_i) \tag{47a}$$

$$\text{subject to } B^\top y = \mathbb{0}_m\,. \tag{47b}$$

(a) Formulate the Lagrangian $\mathcal{L}(y,\lambda)$ as well as the KKT conditions (10) for this problem. Show that a generic pair $(y^\star,\lambda^\star)$ is a solution of the KKT system if and only if $y^\star = x^\star\mathbb{1}_n$ and $\lambda^\star = \lambda_\mathrm{p} + \lambda_\mathrm{h}$ where $\lambda_\mathrm{h}\in\text{kernel}(B)$, and $x^\star\in\mathbb{R}$ and $\lambda_\mathrm{h}\in\mathbb{R}^m$ are unique. Try to further characterize $x^\star\in\mathbb{R}$ and $\lambda_\mathrm{h}\in\mathbb{R}^m$.

(b) Consider the following continuous-time primal-dual algorithm to solve the problem (47):

$$\dot{y} = -\frac{\partial \mathcal{L}(y,\lambda)}{\partial y} \tag{48a}$$

$$\dot{\lambda} = +\frac{\partial \mathcal{L}(y,\lambda)}{\partial \lambda}\,. \tag{48b}$$

Explain how the algorithm (48) can be implemented in a distributed fashion by means of a block-diagram indicating which variables need to be communicated and which computations can be performed locally.

(c) In the following, we analyze the algorithm (48). Write the algorithm (48) in error coordinates $(y - y^\star, \lambda - \lambda^\star)$, where $y^\star$ is the unique minimizer of (48) and $\lambda^\star$ is the associated optimal Lagrange multiplier. Show for the particular case of an *acyclic* graph that the equilibria are uniquely determined and all trajectories converge to $y = y^\star$ and $\lambda = \lambda^\star$.

17. Exercise **(Distributed Dual Ascent Consensus Optimization with Laplacian Constraint)**: As in Example 3, consider the distributed optimization problem (31) with the quadratic local cost functions from Exercise 14 and with the Laplacian constraint (29a) corresponding to a connected and undirected network:

$$\text{minimize}_{y\in\mathbb{R}^n}\ \tilde{f}(y) = \frac{1}{2}\sum\nolimits_{i=1}^{n}(y_i - \underline{x}_i)^\top P_i (y_i - \underline{x}_i) \tag{49a}$$

$$\text{subject to } Ly = \mathbb{0}_n\,. \tag{49b}$$

(a) Formulate the Lagrangian $\mathcal{L}(y,\lambda)$ as well as the KKT conditions (10) for this problem. Analyze the set of saddle points.

(b) Consider the dual ascent (23) algorithm for this problem, argue why it can be implemented in a distributed fashion, and show that this algorithm converges to the set of saddle points.

18. Exercise **(Discrete-Time Dual Ascent Consensus Optimization with Incidence Constraint)**: Consider the distributed optimization problem (35) with the incidence consensus constraint and the dual ascent algorithm (23) with $K = I$ to solve it. In the following, we consider a discrete-time implementation by replacing the continuous-time dual gradient step (23b) by its Euler discretization (44b) with a step-size $\varepsilon > 0$.

(a) Draw a block-diagram of the proposed algorithm.

(b) Give a condition on the step-size ε when this algorithm converges for the particular case of an *acyclic* graph.

19. Exercise **(Broadcast scheme)**: As in Example 3, consider the distributed optimization problem (28) with the quadratic local cost functions from Exercise 14. A clever student proposes a distributed optimization algorithm based on a broadcast scheme. Each agent has an associated primal variable y_i and a dual variable λ_i. The update equations are

$$y^\star = -P^{-1}\lambda + \underline{x} \tag{50a}$$

$$k\dot{\lambda} = \Pi y^\star\,, \tag{50b}$$

where $\Pi = I_n - \frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top$ and $k > 0$. Hence, first an entirely local primal update (50a) is performed. Next, the local variables $y^\star$ are broadcast so that every agent knows all the $y_i^\star$ and performs a global averaging of the primal variables (50b).

(a) Draw the block-diagram of the update equations (50).

(b) Argue why the algorithm (50) converges to the strict global minimizer $y^\star$ and associated dual variable $\lambda^\star$.

20. Exercise **(Exponential convergence of Newton's method)**: Consider the algebraic equation (51) with $F : \mathbb{R}^n \to \mathbb{R}^n$ continuously differentiable, and let Assumptions B.1 hold. Derive the exponential convergence rate (54) of the Newton flow (53).

21. Exercise **(Newton's method applied to constrained optimization)**: Consider the constrained optimization problem (8). Apply the Newton flow (53) to find the roots of the associated KKT conditions (10). Under which conditions does the Newton flow (53) converge?

22. Exercise **(Derivation of heavy ball method)**: Consider the second-order continuous-time gradient flow (7). Show that its Euler discretization is obtained as in equation (58).

23. Exercise **(Convergence-rate of heavy ball method)**: Prove Proposition C.2.

Appendix

A LaSalle Invariance Principle

Theorem A.1 (LaSalle Invariance Principle)**.** *Consider a dynamical system (X, f) with differentiable f. Assume there exist*

1. *a compact set $W \subset X$ that is invariant for (X, f), and*
2. *a continuously-differentiable function $V : X \to \mathbb{R}$ satisfying $\frac{\partial V(x)}{\partial x} f(x) \leq 0$ for all $x \in X$.*

Then each solution $t \mapsto x(t)$ starting in W, that is, $x(0) \in W$, converges to the largest invariant set contained in

$$\big\{x \in W \mid \frac{\partial V(x)}{\partial x} f(x) = 0\big\}.$$

Note: If the set S is composed of multiple disconnected components and $t \mapsto x(t)$ approaches S, then it must approach one of its disconnected components. Specifically, if the set S is composed of a finite number of points, then $t \mapsto x(t)$ must converge to one of the points.

B Newton's Method and Applications in Optimization

Acknowledgment: part of the results in this section originated from discussions with C. de Persis.

An alternative method to find a minimizer of an optimization problem is to directly attack the algebraic conditions for optimality: (3) in the unconstrained case and the KKT conditions (10) in the constrained case. A general purpose method to solve a nonlinear algebraic equation of the form

$$F(x) = \mathbb{0}_n \tag{51}$$

is the well-known *Newton's method* or *Newton iteration*. In this section, we will derive Newton's method in continuous time, analyze its convergence rate, and apply it to optimization problems. We will assume throughout that $F : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable.

B.1 Derivation of the Newton iteration and the Newton flow

Newton's method can be motivated as follows: assume that we have an initial guess x_0 for a root of the equation (51). If this guess is incorrect, then a better guess x_1 can be obtained from a first-order Taylor expansion of (51) at the initial guess x_0:

$$\mathbb{0}_n = F(x_0) + \frac{\partial F(x_0)}{\partial x}(x_1 - x_0)\,.$$

If the Jacobian of F is nonsingular, then the latter equation can be solved for x_1 as

$$x_1 = x_0 - \frac{\partial F(x_0)}{\partial x}^{-1} F(x_0)$$

By iterating this idea, we arrive at the *Newton iteration*

$$x^+ = x - \frac{\partial F(x)}{\partial x}^{-1} F(x)\,. \tag{52}$$

In continuous time, the Newton iteration (52) then reads as the *Newton flow*

$$\dot{x} = -k \left(\frac{\partial F(x)}{\partial x} \right)^{-1} F(x) \,, \tag{53}$$

where $k > 0$ is a positive gain.

B.2 Exponential convergence of the Newton flow

In the following, we study the convergence properties of the Newton flow (53) under the following two assumptions:

Assumption B.1. *Consider the algebraic equation (51) with $F : \mathbb{R}^n \to \mathbb{R}^n$ continuously differentiable. Assume that*

A1) there exists an isolated solution $x = x^\star \in \mathbb{R}^n$ of (51): $F(x^\star) = \mathbb{0}_n$; and

A2) there exists $\mathbb{B}_r(x^\star)$, a ball or radius r and center $x^\star$, over which the Jacobian of F is invertible and satisfies

$$\max_{x \in \mathbb{B}_r(x^\star)} \left\| \left(\frac{\partial F}{\partial x} \right)^{-1} \right\| = m(r).$$

for some positive constant $m(r)$.

Under this assumption, we are able to establish exponential convergence of the *Newton flow* [16].

Theorem B.2 (Exponential convergence of the Newton flow)**.** *Consider the algebraic equation (51) with $F : \mathbb{R}^n \to \mathbb{R}^n$ continuously differentiable. The following statements hold:*

1. *If Assumption B.1.A1) holds, then all solutions $x(t)$ of the Newton flow (53) that start from an initial condition $x_0 \in \mathbb{R}^n$ sufficiently close to the solution $x^\star$ converge to $x^\star$.*

2. *If additionally, Assumption B.1.A2) holds, then these solutions satisfy*

$$\|x(t) - x^\star\| \leq k m(r) L \|x_0 - x^\star\| \mathrm{e}^{-kt} \,, \quad \forall t \geq 0 \,, \tag{54}$$

where $L = \max_{x \in \mathbb{B}_r(x^\star)} \|F(x) - F(x^\star)\|$ is the maximum Lipschitz constant of $F(x)$ in $\mathbb{B}_r(x^\star)$.

Proof. Asymptotic stability: We first show that $x^\star$ is an asymptotically stable equilibrium of (53). Consider the Krasovskii-type Lyapunov function

$$V(x) = \frac{1}{2} F(x)^\top F(x).$$

Under Assumption B.1.A1), the Lyapunov function is positive definite in a suitable open subset of $\mathbb{B}_r(x^\star)$. Moreover, its derivative along the Newton flow (53) equals

$$\dot{V}(x) = \frac{\partial V}{\partial x}^\top \dot{x} = F(x)^\top \frac{\partial F}{\partial x} \left(-k \left(\frac{\partial F}{\partial x} \right)^{-1} F(x) \right) = -k \|F(x)\|^2 = -2kV(x) \,,$$

where the right-hand side is nonpositive and equal to zero in a suitable subset of $\mathbb{B}_r(x^\star)$ if and only if $x = x^\star$. By Lyapunov's theorem, $x^\star$ is a locally asymptotically stable equilibrium.

Exponential convergence: This proof is left as an exercise to the reader; see Exercise 20. □

B.3 The Newton flow in optimization

In the following, we apply the Newton flow (53) to solve the unconstrained optimization problem (1), that is, we seek the roots of equation (3) where $F(x) = \frac{\partial f(x)}{\partial x}$. We assume that the Hessian matrix of f, $\mathrm{Hess} f(x) = \frac{\partial^2 f(x)}{\partial x^2}$, exists and is nonsingular. In this case, the Newton flow (53) equals

$$\dot{x} = -k \left(\mathrm{Hess} f(x)\right)^{-1} \frac{\partial f(x)}{\partial x}\,, \tag{55}$$

Theorem B.3 (Newton's method in optimization)**.** *Consider the unconstrained optimization problem (1). Assume that f is strictly convex, twice continuously differentiable with a nonsingular Hessian matrix $\mathrm{Hess} f(x)$, and that there is an isolated minimizer $x^\star$. Then all solutions $x(t)$ of the Newton flow (55) that start sufficiently close to the minimizer $x^\star$ satisfy (54).*

Newton's algorithm has the desirable property of converging *exponentially* to the minimizer, but it requires the computation of the inverse of the Hessian function, which can be very expensive and hampers the possibility of a distributed implementation. The other drawback is that convergence is only local. The careful reader may have noticed that the Newton flow (55) is actually a gradient flow (5) with a *state-dependent gain matrix* $K = k \left(\mathrm{Hess} f(x)\right)^{-1}$. The advantage of this state-dependent gain is that the convergence rate is *uniform* in all directions, as illustrated by the following example.

Example 5 (Newton's method applied to LQ program)**.** *Consider the LQ program*

$$\mathit{minimize}_{x\in\mathbb{R}^2}\ f(x) = \frac{1}{2} x^\top A x - b^\top x\,, \tag{56}$$

where $A = \left[\begin{smallmatrix} 1 & \\ & 1000 \end{smallmatrix}\right]$ and $b = \left[\begin{smallmatrix} 1 \\ 2 \end{smallmatrix}\right]$. The problem (56) is strictly convex with minimizer given by $x^\star = A^{-1} b$. The weighted gradient method (5) with scalar gain $k > 0$ applied to this optimization problem (56) reads as $\dot{x} = -k\,(Ax - b) = -kA(x - x^\star)$. In error coordinates $\tilde{x} = x - x^\star$ we have

$$\dot{\tilde{x}} = -kA\tilde{x} = -k \begin{bmatrix} 1 & \\ & 1000 \end{bmatrix} \tilde{x}\,.$$

Hence, depending on the initial condition $x_0 \in \mathbb{R}^2$, the gradient method converges as $\exp(-kt)$ or as $\exp(-1000kt)$. Newton's method (55) applied to problem (56) yields, with $\mathrm{Hess} f(x) = A$,

$$\dot{x} = -kA^{-1}\,(Ax - b) = -kA^{-1}A(x - x^\star) = -k(x - x^\star)\,.$$

Thus, independent of initial conditions, the error coordinate $\tilde{x} = x - x^\star$ converges "uniformly" as $\exp(-kt)$. Thus, Newton's method can be understood as a pre-conditioning of the problem (56). □

The Newton flow (53) can also be used to solve the constrained optimization problem (8) by seeking the roots of the associated KKT conditions (10). We leave the detailed analysis as an exercise to the reader; see Exercise 21.

C Convergence Rates of Discrete-Time Gradient Methods

We focused thus far almost exclusively on continuous-time algorithms — all of which are based on the negative gradient flow (4). We have briefly touched upon its discrete-time implementation (5) in Remark 1.3 and in Exercise 4. We have already seen in the continuous-time case that the second-order implementation (7) involving a momentum term $\ddot{x}$ improves the convergence rate of the first-order gradient flow (4); see Exercises 2 and 5. Though in continuous time this distinction is somewhat arbitrary as the gradient flow can be sped up indefinitely by adding a gain as in (5); see also Exercise 3. In what follows, we take a closer on the convergence rates of different discrete-time algorithms. Our treatment is motivated particularly from the lecture notes [17] by B. Recht.

C.1 Discrete-time gradient descent and heavy ball method

Consider again the unconstrained optimization problem (1). An Euler discretization of the standard gradient descent method (4) with a time-varying step size $\gamma_k > 0$ yields the iteration

$$x_{k+1} = x_k - \gamma_k \frac{\partial f(x_k)}{\partial x}\,. \tag{57}$$

Consider also the second-order gradient flow (7) and its Euler discretization (see Exercise 22):

$$x_{k+1} = x_k - \frac{b}{a+1}\frac{\partial f(x_k)}{\partial x} + \frac{1}{a+1}(x_k - x_{k-1}) \tag{58}$$

With the shorthands $\alpha_k = \frac{b}{a+1}$ and $\beta_k = \frac{1}{a+1}$, we arrive at the so-called *heavy-ball method*

$$x_{k+1} = x_k - \alpha_k \frac{\partial f(x_k)}{\partial x} + \beta_k(x_k - x_{k-1})\,, \tag{59}$$

where we allow α_k, β_k to be time-varying positive sequences. Observe that (59) reads as the gradient method (57) plus the extra term $(x_k - x_{k-1})$ referred to as *momentum*. The iteration is termed heavy ball method as it describes the movement of a point mass in the potential field $\frac{\partial f}{\partial x}$.

C.2 Convergence rate analysis

In the following discussion, we will fine-tune the step-sizes γ_k, α_k, and β_k to optimize the respective convergence rates. We will also demonstrate that the heavy ball method (59) improves over the gradient method (57) in terms of its convergence rate. To keep our discussion crisp and concise we restrict the general optimization problem (1) to the unconstrained quadratic optimization problem

$$\text{minimize}_{x\in\mathbb{R}^n}\ f(x) = \frac{1}{2}x^\top A x - b^\top x + c\,, \tag{60}$$

where $A \in \mathbb{R}^{n\times n}$ is a positive definite matrix, $b \in \mathbb{R}^n$ is a vector, and $c \in \mathbb{R}$ is a constant. Observe that the problem (60) admits a unique minimizer given by $x^\star = A^{-1}b$.

We denote the maximum (respectively, the minimum) eigenvalue of A by λ_L (respectively, by λ_l), where "L" and "l" stand for the maximum and minimum Lipschitz constants of a map associated to the A-matrix. We will denote the *condition number* of A by $\kappa = \lambda_L/\lambda_l$.

We begin our analysis with the gradient method (57). Surprisingly, we find that the optimal step-size is a constant one, and the convergence rate is determined by the condition number κ.

Proposition C.1. *Consider the unconstrained quadratic optimization problem (60) and the gradient method (57) with a positive step-size sequence γ_k. The per-step convergence of the method is*

$$\|x_{k+1} - x^\star\| \le \max\{|1-\gamma_k\lambda_L|,\, |1-\gamma_k\lambda_l|\}\,\|(x_k - x^\star)\|\,. \tag{61}$$

The per-step convergence rate (61) is maximized by the constant step-size $\gamma_k = 2/(\lambda_L + \lambda_l)$ yielding

$$\|x_k - x^\star\| \le \left(\frac{\kappa-1}{\kappa+1}\right)^k \|x_0 - x^\star\|\,. \tag{62}$$

Proof. We can express the error $x_k - x^\star$ at the k^{th} iteration as

$$\begin{aligned}\|x_{k+1} - x^\star\| &= \left\| x_k - \gamma_k \frac{\partial f(x_k)}{\partial x} - x^\star \right\| = \|(I - \gamma_k A)(x_k - x^\star)\| \\ &\leq \|(I - \gamma_k A)\| \, \|(x_k - x^\star)\| \leq \max\{|1 - \gamma_k \lambda_L|, \, |1 - \gamma_k \lambda_l|\} \, \|(x_k - x^\star)\| \,,\end{aligned}$$

where we used $A\, x^\star = b$. The $\max\{\cdot\}$ in the convergence rate is illustrated in Figure 13. Note that a constant step-size policy $\gamma_k = 2/(\lambda_L + \lambda_l)$ is the lower bound on the maximum and yields

$$\|x_k - x^\star\| \leq \left(\frac{\lambda_L - \lambda_l}{\lambda_L + \lambda_l} \right)^k \|x_0 - x^\star\|$$

which equals (62) with the condition number $\kappa = \lambda_L / \lambda_l$. □

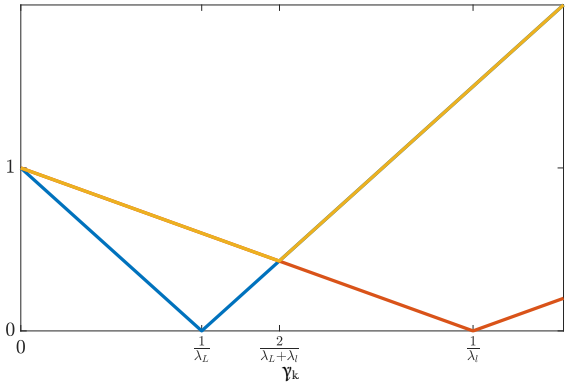


Figure 13: Optimal choice of the step-size γ_k.

For the heavy ball method (59), we specialize the discussion to time-invariant iteration constants. A similar proof as for Proposition C.1 accounting for the two errors $(x_{k+1} - x^\star, x_k - x^\star)$ yields the following result; see Exercise 23.

Proposition C.2. *Consider the unconstrained quadratic optimization problem (60) and theheavy ball method (59) with a positive time-invariant step sizes $\alpha_k = \alpha$ and $\beta_k = \beta$ for all $k \in \mathbb{N}$. For $\alpha = 4/(\sqrt{\lambda_L} + \sqrt{\lambda_l})^2$ and $\beta = \max\{|1 - \sqrt{\alpha\lambda_L}|, \, |1 - \sqrt{\alpha\lambda_l}|\}^2$ the method converges as*

$$\|x_k - x^\star\| \leq \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} + \epsilon_k \right)^k \|x_0 - x^\star\|. \,, \tag{63}$$

where $\lim_{k \to \infty} \epsilon_k = 0$.

We now compare the number of iterations required by both the the gradient method (57) and the heavy ball method (59) to arrive within a certain proximity of the optimizer, i.e., $\|x_k - x^\star\| \leq \mu \, \|x_0 - x^\star\|$ for $\mu > 0$. The rates (62) and (63) then yield the following number of iterations:

$$\begin{aligned}\text{gradient descent:} \quad & k_{gd} \geq \frac{\kappa}{2} \log(\frac{1}{\mu}) \\ \text{heavy ball:} \quad & k_{hb} \geq \frac{\sqrt{\kappa}}{2} \log(\frac{1}{\mu})\end{aligned}$$

Observe that the heavy ball method considerably speeds up the convergence. As an illustration, consider $\kappa = 100$ and observe that $k_{gd}/k_{hb} = 10$ for the same value of μ, i.e., the gradient method requires 10 times the number of iterations for similar convergence vis-à-vis the heavy ball method.

References

[1] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, 2005.

[2] D.P. Bertsekas. *Constrained optimization and Lagrange multiplier methods.* Academic Press, Inc., Halsted Press, 1981.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[4] A. Cherukuri, B. Gharesifard, and J. Cortes. Saddle-point dynamics: conditions for asymptotic stability of saddle points, 2015.

[5] A. Cherukuri, E. Mallada, S. Low, and J. Cortes. The role of convexity on saddle-point dynamics: Lyapunov function and robustness. *arXiv preprint arXiv:1608.08586*, 2016.

[6] G. Droge, H. Kawashima, and M. Egerstedt. Continuous-time proportional-integral distributed optimisation for networked systems. *Journal of Control and Decision*, 1(3):191–213, 2014.

[7] D. Feijer and F. Paganini. Stability of primal–dual gradient dynamics and applications to network optimization. *Automatica*, 46(12):1974–1981, 2010.

[8] B. Gharesifard and J. Cortes. Distributed continuous-time convex optimization on weight-balanced digraphs. *IEEE Transactions on Automatic Control*, 59(3):781–786, 2014.

[9] Takeshi Hatanaka, Nikhil Chopra, Takayuki Ishizaki, and Na Li. Passivity-based distributed optimization with communication delays using pi consensus algorithm. *arXiv preprint arXiv:1609.04666*, 2016.

[10] K. Kvaternik and L. Pavel. A continuous-time decentralized optimization scheme with positivity constraints. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pages 6801–6807. IEEE, 2012.

[11] Laurent Lessard, Benjamin Recht, and Andrew Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1):57–95, 2016.

[12] P. Lin, W. Ren, and J.A. Farrell. Distributed continuous-time optimization: nonuniform gradient gains, finite-time convergence, and convex constraint set. *IEEE Transactions on Automatic Control*, 2016.

[13] S. Liu, Z. Qiu, and L. Xie. Continuous-time distributed convex optimization with set constraints. *IFAC Proceedings Volumes*, 47(3):9762–9767, 2014.

[14] A. Nedić and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

[15] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

[16] Alexander G Ramm. Acceleration of convergence of a continuous analog of the newton method. *Applicable Analysis*, 81(4):1001–1004, 2002.

[17] B. Recht. Lyapunov analysis and the heavy ball method. CS726 Lecture Notes, 2012.

[18] R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[19] Guodong Shi, Karl Henrik Johansson, and Yiguang Hong. Reaching an optimal consensus: dynamical systems that compute intersections of convex sets. *IEEE Transactions on Automatic Control*, 58(3):610–622, 2013.

[20] W. Su, S. Boyd, and E. Candes. In *Advances in Neural Information Processing Systems*, pages 2510–2518, 2014.

[21] J. N. Tsitsiklis. *Problems in Decentralized Decision Making and Computation*. PhD thesis, Massachusetts Institute of Technology, November 1984.

[22] J. Wang and N. Elia. Control approach to distributed optimization. In *Allerton Conf. on Communications, Control and Computing*, pages 557–561, Monticello, IL, USA, 2010.

[23] Jing Wang and N. Elia. A control perspective for centralized and distributed convex optimization. In *IEEE Conf. on Decision and Control and European Control Conference*, pages 3800–3805, Orlando, FL, USA, December 2011.

[24] F. Zhang. *The Schur Complement and Its Applications*. Springer, 2005.