Optimization Algorithms as Robust Feedback Controllers

Adrian Hauswirth, Saverio Bolognani, Gabriela Hug, and Florian Dörfler

Department of Information Technology and Electrical Engineering, ETH Zürich, Switzerland

Abstract

Mathematical optimization is one of the cornerstones of modern engineering research and practice. Yet, throughout all application domains, mathematical optimization is, for the most part, considered to be a numerical discipline. Optimization problems are formulated to be solved numerically with specific algorithms running on microprocessors. An emerging alternative is to view optimization algorithms as dynamical systems. While this new perspective is insightful in itself, liberating optimization methods from specific numerical and algorithmic aspects opens up new possibilities to endow complex real-world systems with sophisticated self-optimizing behavior. Towards this goal, it is necessary to understand how numerical optimization algorithms can be converted into feedback controllers to enable robust "closed-loop optimization". In this article, we review several research streams that have been pursued in this direction, including extremum seeking and pertinent methods from model predictive and process control. However, our primary focus lies on recent methods under the name of "feedback-based optimization". This research stream studies control designs that directly implement optimization algorithms in closed loop with physical systems. Such ideas are finding widespread application in the design and retrofit of control protocols for communication networks and electricity grids. In addition to an overview over continuous-time dynamical systems for optimization, our particular emphasis in this survey lies on closed-loop stability as well as the enforcement of physical and operational constraints in closed-loop implementations. We further illustrate these methods in the context of classical problems, namely congestion control in communication networks and optimal frequency control in electricity grids, and we highlight one potential future application in the form of autonomous reserve dispatch in power systems.

Keywords: feedback-based optimization, autonomous systems, nonlinear control, nonconvex optimization

Contents

1. Introduction

Most advances in mathematical optimization in the past decades have been geared towards numerical implementations of iterative algorithms. The common viewpoint is that an optimization problem can be formulated, transformed, reduced, and relaxed, but ultimately the necessary steps to solve the problem rely purely on numerical linear algebra, which can be implemented and run on microprocessors. This offline computed solution is then used to reach and realize some decision.

This paradigm of optimization as a computational problem is almost synonymous with the field of management science and operations research (Bertsimas & Freund, 2004; Hillier & Lieberman, 2001), which has flourished ever since the inception of linear programming in the mid-20th century. Today, this kind of offline optimization is applied in various disciplines ranging from econometrics over statistical and machine learning (Bishop, 2009) to optimal control (Bertsekas, 2017).

From a control perspective, solving an optimization problem offline (with known data) and implementing its output as a decision is a *feedforward* approach. In contrast, in this article, we consider *feedback* approaches to constrained nonlinear optimization to drive a physical system towards an optimal steady state. This type of *closed-loop optimization* has been pursued mainly for four reasons:

(i) to increase the robustness against inaccurate problem data and time-varying disturbances,

(ii) to reduce model-dependence, i.e., to make the optimization procedure model-free,

(iii) to minimize computational effort, and

(iv) to eliminate exogenous setpoints and reference signals.

These reasons resonate well with the general feedback and feedforward paradigms advocated in control textbooks (Doyle et al., 2009; Franklin et al., 2010), and we further dwell on them hereafter.

Robustness is key in optimization. More often than not, practical problems lack precise data. Parameters and states based on measurements and statistical inference are inherently inaccurate, and so is the solution of an optimization problem based on such data. The earliest attempts at addressing this issue have resulted in sensitivity analysis for optimization problems (Shapiro, 1988) which asks how a solution changes as problem parameters vary. From a more practical perspective, robust optimization (Ben-Tal et al., 2009) and stochastic programming (Bonnans, 2019) offer ways to incorporate uncertainty in the problem. However, these approaches are inherently conservative and can entail massive computational cost because they need to take into account the full set of possible problem instances. In contrast, using feedback, one needs to react only to the actual realization of the underlying disturbance process.

More radical than improving robustness is the idea of rendering optimization schemes *model-free*: A physical system defines a set of constraints (either algebraic or dynamic and often time-varying). Therefore, it is only natural to probe a system for information and learn its behavior from measurements instead of building a static model from first principles or previously acquired data.

Rather than learning its behavior online, a physical system can also be used to enforce constraints directly: The laws of nature couple inputs and outputs of a plant. Hence, for any input, the physical system will produce an output that satisfies this input-output relation. Similarly, saturation effects will naturally guarantee that states and inputs do not exceed their limits. Especially if a plant can be safely operated at the saturation limit, this type of natural constraint satisfaction can be used to lighten the computational load of a closed-loop optimization scheme because the physical system acts as a "constraint enforcer".

Finally, compared to standard feedback loops, closed-loop optimization setups can run without external setpoints or references. Instead, an economic objective can be directly optimized as long as a cost function can be specified and evaluated. This feature is particularly powerful in combination with the inherent constraint enforcement: Whereas in classical control setups pre-computed setpoints have to be feasible (e.g., lie within actuator limits), convergence to a feasible and optimal state is a defining aspect of closed-loop optimization.

The boundary between feedforward and feedback optimization, however, is not always clear cut. For instance, model predictive control (MPC) uses feedback to achieve robustness but relies on an accurate model for the formulation and solution of an optimal control problem at every iteration. On the other hand, some of the control schemes presented in this article require the computationally cheap solution of a simple quadratic program (QP) to compute a feasible descent direction at every iteration.

Given this potential ambiguity, the focus of this article are the implementation of optimization algorithms in closed loop, rather than "real-time numerical optimization" to control a physical process. Nevertheless, we give an overview over various other approaches in Section 2.

For the sake of a concise presentation, we restrict ourselves to continuous-time systems. Even though, analogous discrete-time results exist or can be derived similarly to the continuous-time case.

1.1. Illustrative Examples and Prototypical Problem Setup

The following idealized yet general example of a gradient system interconnected with a physical plant illustrates the central idea of this article and serves as a starting point for subsequent comparisons and variations.

Example 1.1. Consider a dynamic nonlinear plant

$$\dot{\zeta} = f(\zeta, u) \qquad y = g(\zeta) + d\,, \tag{1}$$

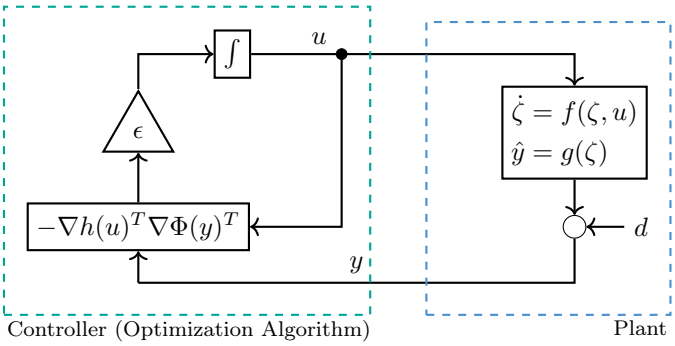


Figure 1: Simple feedback-based gradient flow

where ζ, u and y are the state, input, and output, and d denotes an additive disturbance. The vector field $f(\cdot,\cdot)$ and the map $g(\cdot)$ describe the process and output measurement, respectively.

We assume that, for any fixed u and d, the plant is asymptotically stable with fast-decaying transients such that, for every u, there exists a unique steady state $\hat{h}(u)$ such that $0 = f(\hat{h}(u), u)$. Consequently, there also exists a *steady-state map* $h(u) := g(\hat{h}(u))$. We assume h to be continuously differentiable in u.

We wish to minimize a cost $\Phi(y)$ which is a function of the plant output y. Given h and d, we may equivalently minimize the *reduced* cost $\tilde{\Phi}(u) := \Phi(h(u) + d)$ instead. For this purpose, we consider a simple gradient flow

$$\dot{u} = -\nabla \tilde{\Phi}(u)^T = -\nabla h(u)^T \nabla \Phi(h(u) + d)^T \,, \qquad (2)$$

where $\nabla h(u)$ is due to the chain rule applied to $\Phi(h(u)+d)$.

The gradient flow (2) is a closed system. However, recognizing $h(u) + d$ as the measurable output y, (2) can be easily transformed into an open system and interconnected with the plant (1) as shown in Figure 1. This yields the closed-loop dynamics

$$\begin{aligned} \text{plant} \begin{cases} \dot{\zeta} &= f(\zeta, u) \\ y &= g(\zeta) + d \end{cases} \\ \text{controller} \begin{cases} \dot{u} &= -\epsilon \nabla h(u)^T \nabla \Phi(y)^T \,, \end{cases} \end{aligned} \qquad (3)$$

where $\epsilon > 0$ is a scalar control gain.

The systems (1), (2) and (3) can be understood from a *singular perturbation* viewpoint (Khalil, 2002, Chap. 11): As $\epsilon \to 0^+$, the plant behavior is replaced by the algebraic map h, and the remaining dynamics (2) are the "slow" *reduced system*. Conversely, on a fast timescale, on which u and d can be assumed to be constant, the plant dynamics (1) are referred to as the "fast" *boundary-layer system*.

Thanks to the integral control structure of (3), it can be easily seen that any equilibrium point $(\zeta^\star, u^\star)$ of (3) is a steady-state of the plant and satisfies $\nabla \tilde{\Phi}(u^\star)^T = \nabla h(u^\star)^T \nabla \tilde{\Phi}(h(u^\star) + d)^T = 0$. Therefore, $u^\star$ is a critical point of $\tilde{\Phi}$ (and a minimizer if $\tilde{\Phi}$ is convex).

Crucially, the controller in (3) does not require explicit knowledge of h (nor of f, g). Instead, only the cost function gradient $\nabla \Phi(y)$ as well as steady-state input-output sensitivities $\nabla h(u)$ are required. Moreover, the additive disturbance d does not need to be known or explicitly estimated and is fully rejected, i.e., an equilibrium is a critical point of $\Phi(h(u) + d)$, independently of the value of d. ■

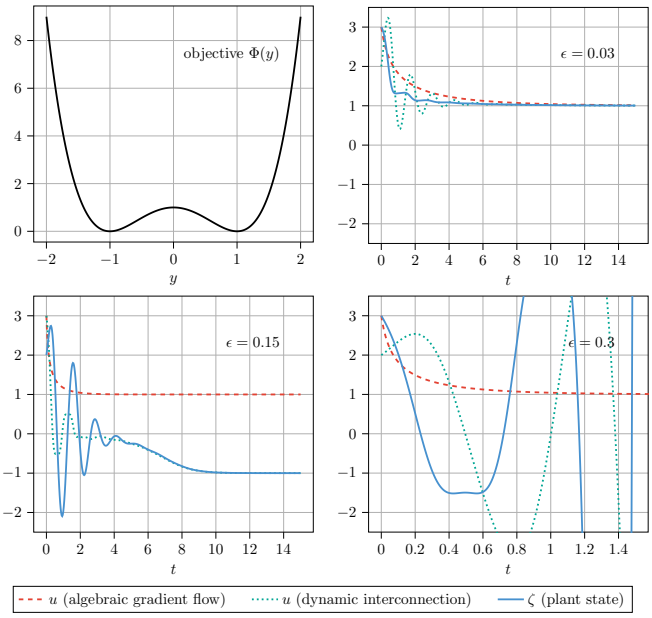


Figure 2: Illustrations for Example 1.2 (top left: objective function; remaining panels: system trajectories for different control gains ϵ)

One of the fundamental points left open by Example 1.1 is *closed-loop stability*. The idea that plant dynamics in (3) need to be fast-decaying is indeed crucial as the following numerical example shows.

Example 1.2. Consider the objective $\Phi(y) = (y^2 - 1)^2$ which is illustrated in the top left panel of Figure 2 and has two isolated minima $\{-1, 1\}$. Consider a single-input-single-output second-order plant governed by

$$\ddot{\zeta} + a\dot{\zeta} + b(\zeta - u) = 0$$

with $a = 2$ and $b = 25$ and $y = \zeta$. The plant is asymptotically stable, under-damped, and, at steady state, we have $y = \zeta = u$. Hence, the controller in (3) takes the form

$$\dot{u} = -\epsilon \nabla \Phi(y)^T = -4\epsilon y(y^2 - 1) \,.$$

Figure 2 shows trajectories of the closed-loop system (3) for the same initial condition, but different values of the gain ϵ, and comparing it to the "algebraic" gradient flow (2) given by $\dot{u} = -\epsilon \nabla \Phi(h(u))^T = -4\epsilon u(u^2 - 1)$.

We observe, that for the given initial condition the algebraic gradient trajectory converges to the minimizer at 1. In contrast, the trajectories of the closed-loop system (3) converge to either one of the two minimizers or diverge, depending on ϵ. In other words, closed-loop stability of (3) is not guaranteed, and even if it is, convergence may not be to the same minimizer as for (2). ■

The issue of stability is central to many closed-loop optimization setups and work on this topic, including quantitative stability requirements, will be discussed in more detail in Section 4.2.

Another key challenge of many real-world problems (and another main topic of this article) is the handling of *input and output constraints*. Because we consider physical systems, these constraints may be of different nature. For instance, *thermal limits* of physical components are often fairly benign since they can be violated temporarily, but should be satisfied in the long run. These constraints are not enforced automatically and need to be addressed with proper control design. Another type of constraints are imposed by *input saturation* of a plant due to limited actuator capabilities or due to the actions of a low-level controller. These constraints are enforced by the physical system irrespective of the controller. However, in order to achieve desirable closed-loop dynamics, the control design has to account for them. Finally, certain constraints may constitute *hard physical limits* whose violation would trigger the immediate instability, collapse, or destruction of the entire system. Thus, these constraints need to be satisfied at all times.

Hence, in real-world applications constraints on the plant inputs and outputs often take center stage and the general optimization problem that one wishes to solve in closed loop can be expressed as

$$\begin{aligned} \text{minimize} \quad & \Phi(u, y) && \text{(4a)} \\ \text{subject to} \quad & y = h(u, d) && \text{(4b)} \\ & u \in \mathcal{U} && \text{(4c)} \\ & (u, y) \in \mathcal{X}\,, && \text{(4d)} \end{aligned}$$

where h is the steady-state input-output map of a dynamic plant (like (1) in Example 1.1) and d is a disturbance, albeit not necessarily additive. Further, $\mathcal{U}$ denotes a set of admissible inputs, and $\mathcal{X}$ is a set of additional engineering constraints.

The particular structure and properties of $\Phi, h, \mathcal{U}$, and $\mathcal{X}$ largely depend on the specific problem setup, but a variety of "algorithmic strategies" from optimization can be cast into feedback control components to solve (4) in closed loop. This will be the overall topic of Sections 3 and 4 where we discuss optimization dynamics and their feedback realizations, respectively.

1.2. Key Application: Optimal Power Reserve Dispatch

Throughout this article, we provide small educational examples illustrating key concepts. In Section 4 we further present two classical applications that can be cast as closed-loop optimization problems: congestion avoidance in communication networks and secondary frequency control in power systems. These examples concern special cases of (4). One application where the full generality of (4) is required is the optimal real-time operation of future power systems with highly variable operating conditions.

Because of changing consumption patterns (e.g., caused by charging of electric vehicles) and increasing infeed from intermittent renewable generation (e.g., from solar and wind plants) future power systems will be faced with less predictable and more volatile conditions. These circumstances require new control and decision protocols to continue to operate power grids safely and efficiently. In a simplified and idealized fashion, this task of optimal real-time grid operation can be cast as tracking the solution of a (time-varying) AC optimal power flow (ACOPF) problem (Frank et al., 2012a,b; Huneault & Galiana, 1991) and has has been studied from an online closed-loop perspective in Dall'Anese & Simonetto (2018); Hauswirth et al. (2017); Tang et al. (2017), among others.

In this subsection, we provide a brief formal description of this problem and formulate the main challenges from an online optimization viewpoint. In Section 4 we then revisit this problem and present feedback-based optimization solutions from the literature.

1.2.1. Problem Description

Consider an AC power transmission network as an undirected graph with a set $\mathcal{N}$ of buses and a set $\mathcal{M}$ of transmission lines. We write $l \to k$ to denote a line from bus l to bus k, i.e., $(l, k) \in \mathcal{M}$. As decision variables, each bus $l \in \mathcal{N}$ has an associated voltage magnitude v_l, voltage angle θ_l, and active and reactive power generation $p_l^{\mathrm{G}}, q_l^{\mathrm{G}}$. By $p^{\mathrm{G}}, q^{\mathrm{G}}$, etc. we denote the vectors obtained from stacking all respective components p_l^{G} for all $l \in \mathcal{N}$.

For simplicity and without loss of generality, we consider the cost $\Phi(p^{\mathrm{G}}) = \sum_{l \in \mathcal{N}} \Phi_l(p_l^{\mathrm{G}})$ for active power as the minimization objective.

Hence, a basic ACOPF problem is given by

$$\begin{aligned} \underset{v, \theta, p^{\mathrm{G}}, q^{\mathrm{G}}}{\text{minimize}} \quad & \Phi(p^{\mathrm{G}}) && \text{(5a)} \\ \underset{\forall l \in \mathcal{N}}{\text{subject to}} \quad & 0 = p_l^{\mathrm{G}} - p_l^{\mathrm{L}} - \sum\nolimits_{l \to k} p_{lk}(v_l, v_k, \theta_l, \theta_k) && \text{(5b)} \\ & 0 = q_l^{\mathrm{G}} - q_l^{\mathrm{L}} - \sum\nolimits_{l \to k} q_{lk}(v_l, v_k, \theta_l, \theta_k) && \text{(5c)} \\ & \underline{p}_l \le p_l^{\mathrm{G}} \le \overline{p}_l && \text{(5d)} \\ & \underline{q}_l \le q_l^{\mathrm{G}} \le \overline{q}_l && \text{(5e)} \\ & \underline{v}_l \le v_l \le \overline{v}_l && \text{(5f)} \\ & i_{lk}^2(v_l, v_k, \theta_l, \theta_k) \le \bar{i}_{kl}^2 \qquad \forall (l, k) \in \mathcal{M}\,, && \text{(5g)} \end{aligned}$$

The inequalities (5d–f) denote box constraints on power generation and voltages at each bus, and (5g) limits the current that flows through the line from k to ℓ.

The functions $p_{lk}(\cdot), q_{lk}(\cdot)$ and $i_{lk}^2(\cdot)$ denote active and reactive power flow and squared current magnitude from bus l in direction of bus k, respectively, as illustrated in Figure 3. These nonlinear terms depend on line parameters and constitute the main source of complexity in ACOPF problems. Their explicit expressions (neglecting

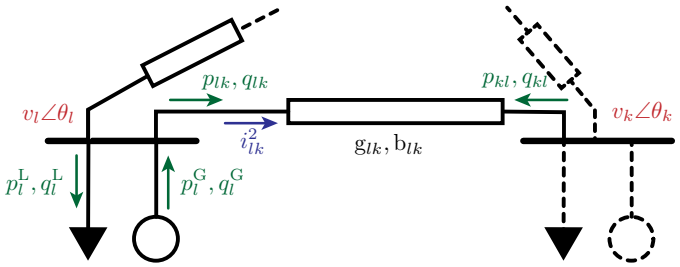


Figure 3: Physical AC power flow quantities

so-called *shunt* elements) are given by

$$p_{lk}(v_l, v_k, \theta_l, \theta_k) := v_l v_k \left(g_{lk} \cos(\theta_l - \theta_k) + b_{lk} \sin(\theta_l - \theta_k)\right)$$
$$q_{lk}(v_l, v_k, \theta_l, \theta_k) := v_l v_k \left(g_{lk} \sin(\theta_l - \theta_k) - b_{lk} \cos(\theta_l - \theta_k)\right)$$
$$i^2_{lk}(v_l, v_k, \theta_l, \theta_k) := (g^2_{lk} + b^2_{lk}) \left(v_k^2 + v_l^2 - 4 v_k v_l \cos(\theta_k - \theta_l)\right)$$

where g_{lk} and b_{lk} denote the *conductance* and *susceptance* of the transmission line connecting buses l and k. Note that $i^2_{lk} = i^2_{kl}$ holds (without shunt elements), but $p_{lk} = -p_{kl}$ and $p_{lk} = -p_{kl}$ are not generally true. For a more comprehensive introduction to AC power flow including alternative formulations the reader is referred to Frank & Rebennack (2016); Molzahn & Hiskens (2019).

ACOPF problems like (5) are well-studied and routinely solved (numerically) in practice. Yet, they remain computationally demanding, because of the nonlinear power flow equations (5b–c) which render the problem non-convex (although convex relaxations can yield global optimality certificates; Low 2014; Molzahn & Hiskens 2019).

1.2.2. Closed-Loop Optimization Modeling

In an online setting, the purpose of (5) is to find adjustments to the power flow and power injections p^G, q^G, given the actual power consumption p^L, q^L (as opposed to predictions used in offline day/hour-ahead calculations). The power consumption p^L, q^L and other parameters in (5) are generally time-varying. Hence, rather than solving (5) once for a given configuration, online optimization schemes need to track the ACOPF problem's solution across time. Moreover, (5) can also be considered when reacting to unforeseen contingencies such as line outages (which modify g_{kl} and b_{kl}) or generator outages (which modify $\overline{p}_l, \overline{q}_l$).

Further, note that (5b–c) are the steady-state equations of a complicated, interconnected nonlinear dynamics of transmission lines, generation units, and their low-level controllers. Although these dynamics can be assumed to be asymptotically stable, attempts at steering the physical system too aggressively towards a solution of (5) can destabilize them, similarly to Example 1.2.

In order to express (5) as a problem of the form (4), we need to identify inputs, outputs, and disturbances. For this purpose, for each generation unit, the active power output and either the reactive power or voltage magnitude are assumed to be controllable[1] and thus make up the control input u. The loads p^L, q^L define the disturbance d. All remaining quantities form the output y.

Consequently, (5) can be brought into the form (4) where the constraints (5d–g) are assigned to either $\mathcal{U}$ or $\mathcal{X}$ according to whether they apply only to controllable variables or not. Under normal operating conditions, the local existence and differentiability of the steady-state map $h(\cdot)$ derived from (5b–c) is guaranteed by the implicit function theorem (Bolognani & Dörfler, 2015).

From a practical perspective, "closing the loop" on the ACOPF problem (5) offers an opportunity to more closely integrate and combine different power system control tasks and thereby increase economic efficiency, resilience, and autonomy. More concretely, a controller tracking the solution of (5) makes complex, yet economically efficient re-dispatch decisions in response to unscheduled events to guarantee voltage and line flow limits are respected. Thus, (5) should be interpreted as a "residual" optimization problem around a pre-planned generation schedule and constraints (5d–e) do not necessarily quantify the full production capacity of a generation unit, but rather the amount of dispatchable reserves.

From a theoretical viewpoint, robustly tracking solutions of (5) presents important challenges:

(i) A large number of physical and engineering constraints of different nature have to be satisfied. Some of these limits, like the generation constraints (5b–c), are physical constraints that are strictly enforced by lower-level controllers or through saturation. Other constraints, such as the line flow limits (5g), are thermal limits that can be violated temporarily. Yet other constraints are hard limits that must not be violated at any time. This mainly concerns dynamic and voltage stability limits which we will not dwell on in this article. Simply note that the voltage constraints (5f) can be understood as guarding against voltage instability (Van Cutsem & Vournas, 1998).

(ii) In general, only an approximate model in the form of the steady-state AC power flow equations (5b–c) can be employed. A dynamical model for a large-scale power system is in practice not available, as parts of the system are owned by different stakeholders, models are proprietary, operating conditions (e.g., which generation units are online) change over time, and internal states are typically inaccessible. Nevertheless, the power system dynamics may be assumed to be asymptotically stable under normal operating conditions. But even then, steady-state models have significant parameter uncertainty and are affected by disturbances.

(iii) The nonlinear nature, especially under critical operating conditions, of the power flow equations (5b–c)

[1] One distinguishes between so-called PQ- and PV-buses. See Hauswirth et al. (2017) for a more detailed discussion including the role of the *slack bus* in numerical simulations).

call for methods that work in the absence of convexity and for ill-conditioned problems.

To address these challenges, we will show how the conceptually simple Example 1.1 has given rise to a multitude of variations that enable us to solve and track solutions of the general problem (4) (and thus of the ACOPF problem (5), which is revisited in Example 4.7).

1.3. Organization

The remainder of this article is structured as follows: In Section 2 we discuss several existing approaches which can be considered *closed-loop optimization* techniques, each of them with specific use cases, advantages, and disadvantages. Section 3 reviews various optimization algorithms, that are suitable for closed-loop implementations, from the perspective of dynamical systems. This allows us, in Section 4, to review recent work on constrained feedback-based optimization and to illustrate different control designs that implement the previously discussed optimization dynamics in closed loop. We pay particular attention to different mechanisms to enforce constraints and to closed-loop stability conditions. In this section, we also present the main application examples from communication networks and power systems. Finally, Section 5 concludes the article, highlights the unresolved problems in this domain, and presents worthwhile and exciting avenues for future research.

2. Existing Approaches

Although the key idea studied in this article is the interconnection of optimization algorithms with physical systems (as exemplified in Example 1.1), various other approaches can be understood as closed-loop optimization. In the following, we therefore review works from different areas that follow the same spirit, even though they have emerged independently from each other and follow different philosophies. Each of these methods is particularly suited for specific types of problems, but none comes without drawbacks.

(i) *Extremum seeking*, an outgrow of adaptive control, puts an emphasis on being completely *model-free* and probing the system with the help of a perturbation signal. However, the approach is limited to systems with low-dimensional inputs and methods for enforcing constraints are limited.

(ii) Emerging from process engineering, the primary merit of *modifier adaptation* is to mitigate the effects of model bias when solving successive optimization problems. By itself, this method does not reduce the computational requirements compared to feedforward optimization. Furthermore, an auxiliary method to estimate input-output sensitivities is required.

(iii) *Real-time iteration* schemes, which are rooted in model predictive control, aim at solving classical receding horizon problems with limited resources. The focus is the stabilization of a plant under state and input constraints. For this purpose, a full model of the plant dynamics is generally required and computational burden scales with the planning horizon.

Remark 2.1. Many more approaches can be interpreted as optimization in conjunction with feedback control. Several topics such as *iterative feedback tuning* (Hjalmarsson, 2002) or *iterative learning control* (Bristow et al., 2006) are concerned with the optimal tuning of controllers by either requiring a sequence of experiments or repetitive operation. Although these methods use measurement data to optimize subsequent control actions, because of their "episodic" nature, they are related only remotely to the key idea in this article, the feedback interconnection of optimization dynamics with physical systems.

For similar reasons, we do not review recent work on *reinforcement learning*, even though it has produced staggering results, especially in sequential decision making for games and in robotics. Even though techniques such as *policy gradient* admit a dynamical perspective, RL is generally (and traditionally) framed as optimal control over Markov decision processes (Bertsekas, 2017; Lewis & Liu, 2012). ■

2.1. Extremum Seeking

Arguably, one of the oldest control methods to steer a plant to an extremum of a function rather than tracking a setpoint is extremum seeking (ES) (see Ariyur & Krstic 2003; Tan et al. 2010 for historical accounts). Its popularity rose in the 1950's and 60's as part of adaptive control (Åström & Wittenmark, 2008) and later regained momentum with the rigorous theoretical results derived in Krstic & Wang (2000); Tan et al. (2006); Wittenmark & Urquhart (1995).

The main idea behind ES is to inject a *dither signal* to locally explore the objective function and "learn" its gradient. This dither signal is generally a sinusoidal, but other perturbations have been proposed (Teel & Popovic, 2001). Consequently, the objective can be optimized without recourse to any model information about the plant (and objective) and without any computation aside from the addition and multiplication of the dither signal. The following example illustrates this fact.

Example 2.1. We consider the same problem as in Example 1.1, i.e., the minimization of $\Phi(y)$ where y is the output of a plant of the form (1). In addition, we assume that the plant is single-input-single-output (and thus all signals are scalar). Instead of the gradient scheme in Figure 1, we apply the ES setup in Figure 4 where a sinusoid perturbs the signal u, yielding $\tilde{u}$, which is then fed to the plant.

As in Example 1.1, we replace the fast plant dynamics by the steady-state map $y = h(u) + d$, and we define the

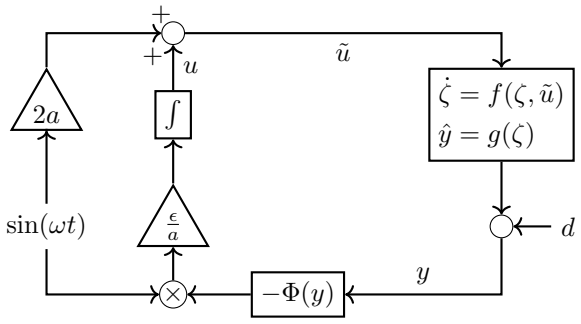


Figure 4: Simple extremum seeking to minimize $\Phi(h(u))$

reduced cost function $\tilde{\Phi}(u) := \Phi(h(u) + d)$. Consequently, the *reduced* ES dynamics of the system in Figure 4 can be expressed in terms of $\dot{u}$ as

$$\dot{u} = F(u,t) := -\tfrac{\epsilon}{a} \underbrace{\Phi\big(h\left(u + 2a\sin(\omega t)\right) + d\big)}_{\tilde{\Phi}(u+2a\sin(\omega t))} + d\big)\sin(\omega t)\,.$$

The *averaged* dynamics are obtained by integrating $F(u,t)$ from 0 to $T = \frac{2\pi}{\omega}$. Namely, using the Taylor expansion of $\tilde{\Phi}$ around u, the average control signal is

$$\begin{aligned}
&\tfrac{1}{T}\int_0^T F(u,t)dt = -\tfrac{\epsilon}{aT}\int_0^T \tilde{\Phi}\big(u + 2a\sin(\omega t)\big)\sin(\omega t)dt \\
&\approx -\tfrac{\epsilon}{aT}\int_0^T \big(\tilde{\Phi}\left(u\right) + 2a\sin(\omega t)\nabla\tilde{\Phi}(u)\big)\sin(\omega t)dt \\
&= -\tfrac{\epsilon}{a}\tfrac{\omega}{2\pi}2a\nabla\tilde{\Phi}(u)\int_0^T \sin(\omega t)^2 dt \;= -\epsilon\nabla\tilde{\Phi}(u)\,,
\end{aligned}$$

where we have neglected higher-order terms in ϵ. Thus, the ES scheme approximates the gradient flow (2) from Example 1.1. However, ES merely requires measurements of $\Phi(y)$ and neither an estimate of ∇h nor of $\nabla\Phi$.

In contrast to the design in Example 1.1, ES systems generally evolve on three (rather than two) different timescales: the plant dynamics (which have been ignored for this example), the frequency range of the probing signal, and the slow averaged optimization dynamics. ■

Classically, averaging theory and singular perturbation analysis (for dynamic plants) are used to render the insights from Example 2.1 rigorous (Guay & Zhang, 2003; Krstic & Wang, 2000; Tan et al., 2006). More recently, ES schemes have also been studied with the help of Lie bracket approximations which offer an alternative perspective (Dürr et al., 2017, 2013a; Grushkovskaya et al., 2018).

While original work only considered finding extrema (i.e. minima or maxima) of unconstrained problems, constraints have been incorporated by submanifold constraints (Dürr et al., 2014), barrier function (DeHaan & Guay, 2005), and saddle-point formulations (Dürr et al., 2013b), and ES has been studied for Nash-equilibrium seeking (Frihauf et al., 2012; Stankovic et al., 2012). Further, ES has been studied for stochastic (Coito et al., 2005; Stanković & Stipanović, 2010) and discrete-time setups (Feiling et al., 2018; Frihauf et al., 2013; Stankovic & Stipanovic, 2009), and hybrid extensions have been proposed (Poveda et al., 2017; Poveda & Teel, 2017).

ES has been applied in the automotive sector (Killingsworth et al., 2009), process engineering (Guay et al., 2004), formation flight and obstacle avoidance (Binetti et al., 2003; Montenbruck et al., 2014) and others. Further applications concern problems in renewable energy such as maximum power point tracking in photovoltaic (Ghaffari et al., 2015) or wind energy systems Ghaffari et al. (2014); Krstic et al. (2014), and Volt-VAR control in power systems (Arnold et al., 2016).

Despite strong theoretical guarantees and being model-free, ES has been confined to relatively low-dimensional, mostly unconstrained, systems. This is due to the fact that plants with multidimensional input require probing signals at different, carefully chosen, frequencies that do not interfere with each other.

2.2. Modifier Adaptation

In the context of *real-time optimization* in process engineering, the notion of *measurement-based optimization* (Chachuat et al., 2009; Francois & Bonvin, 2013) has been used to collect several approaches towards mitigating the effects of model bias in repetitive optimization applications. In the following, we showcase modifier adaptation (MA) methods by François et al. (2005); Gao & Engell (2005); Marchetti et al. (2009a,b).

Given a model of a physical system, assume that we can solve an optimal steady-state problem like (4) numerically. When implementing this solution by setting the appropriate inputs of the stable plant to their pre-computed optimal setpoints, the mismatch between the model estimate (used for computing an optimal state) and the actual plant will invariably lead to a system state that is suboptimal, and possibly violating constraints.

If the optimization of the optimal plant state is performed repeatedly, and at each step the solution is implemented on the physical system, MA provides a method to steadily reduce the discrepancy between model-based solution and physical plant by *modifying* the optimal steady state problem at every iteration by incorporating plant measurements from the previous iteration. MA does not directly "learn" or identify a better model of the plant. Instead, MA corrects the optimization problem by adding adaption terms to the cost and constraint functions. The following simple example demonstrates one possible adaptation mechanism.

Example 2.2. Consider the same setup as in Example 1.1. Namely, we wish to minimize the function $\tilde{\Phi}(u) := \Phi(h(u) + d)$ where $y = h(u) + d$ is the steady-state input-output map of a plant with fast-decaying dynamics. However, only an approximate model $\tilde{h}$ of h and an estimate $\tilde{d}$ of d is available.

Therefore, instead of minimizing $\tilde{\Phi}$, we repeatedly solve

$$\text{minimize}_u \quad \tilde{\Phi}(\tilde{h}(u) + \tilde{d}) + \lambda_k u\,, \tag{6}$$

where λ_k is a *modifier* at iteration k that is adapted at every iteration based on the outcome of the previous iteration $u_k^\star$. In particular, λ is updated according to

$$\lambda_{k+1} = \nabla\tilde{\Phi}(u_k^\star) - \nabla\tilde{\Phi}'(u_k^\star)\,,$$

where $\nabla\tilde{\Phi}(u_k^\star)$ needs to be estimated, and $\nabla\tilde{\Phi}'(u_k^\star) := \nabla(\Phi(\tilde{h}(u_k^\star) + \tilde{d})$ is model-based. The particular structure of $\tilde{\Phi}$, however, lets us write

$$\nabla\tilde{\Phi}(u_k^\star) = \nabla\Phi(h(u_k^\star) + d)\nabla h(u_k^\star)\,, \tag{7}$$

where $h(u_k^\star)+d$ is the measured output of the plant. Thus, essentially, only $\nabla h(u_k^\star)$ needs to be estimated. If the scheme converges to some $u^\star$, we can easily verify that $u^\star$ is a critical point of $\tilde{\Phi}(u)$. ■

Example 2.2 is simplified to the point that a comparison with Example 1.1 is easily possible. However, MA methods are easily applied to constrained problems where modifiers on constraints are introduced analogously (Costello et al., 2014; Faulwasser et al., 2018; Grégory et al., 2014).

Clearly, the tricky part about MA is the estimation of the (true) plant sensitivities $\nabla\tilde{\Phi}(u_k^\star)$. This can be achieved with finite differences (Mansour & Ellis, 2003). Moreover, MA does not reduce the computation burden nor does it aim to reduce the amount of model information required.

2.3. Model Predictive Control with Incomplete Optimization and Real-Time Iterations

Historically, model predictive control (MPC) is an approach to control and stabilize a plant that is subject to input and state constraints. This is achieved by numerically solving an optimal control problem with a finite receding horizon at every sampling time (or, every few sampling instants), but implementing only the first (respectively, few) input(s) of the computed optimal policy before solving the next problem with shifted horizon and based on an updated state measurement.

The high computational requirements have long restricted the application of MPC to relatively slow and low-dimensional plants in process engineering. For standard (linear) MPC, this issue has led to *explicit* MPC (Alessio & Bemporad, 2009; Bemporad et al., 2002) which exploits multi-parametric optimization (Tondel et al., 2003) to solve the receding horizon problem ahead of time and implement the controller as a simple lookup table.

More interesting from our perspective are real-time iterations (RTIs) for *nonlinear MPC* (Bock et al., 2000; Diehl et al., 2002). These methods have emerged as an approximation of *multiple shooting methods* (Diehl et al., 2006) and have been proposed for various applications in process engineering (Diehl et al., 2003), robotics (Diehl et al., 2006), and for airborne kites (Diehl et al., 2005a).

The main idea of RTIs is to solve the optimal control problem only approximately at every iteration by performing only a single iteration of the underlying optimization algorithm (which is usually a sequential quadratic programming (SQP) scheme; Nocedal & Wright 2006; Quirynen et al. 2018; Zavala & Biegler 2009). The first input of the approximate control policy is implemented and the optimization problem for the next sampling period is warm-started at a shifted version of the previous (approximate) solution. In other words, RTIs are a special case of MPC methods with incomplete optimization (Graichen & Kugi, 2010; Grüne & Pannek, 2010; Liao-McPherson et al., 2020). Example 2.3 below illustrates this procedure.

The underlying idea of RTIs is that the approximation error committed by performing only a single optimization iteration is offset by savings in computation time. In particular, because the receding horizon problem is solved more often, it changes less between samples. This feature allows one to prove stability and convergence of RTI schemes (Diehl et al., 2005a, 2007, 2005b; Zanelli et al., 2020).

However, although they interleave optimization iterations with physical dynamics, RTIs have been developed for stabilization and require an exogenous setpoint as well as a dynamic model of the plant. This is particularly reflected in the assumptions on the state cost function, which are, roughly speaking, required to be quadratic functions centered at the origin (see also Remark 2.2 further below on economic MPC).

Example 2.3. We consider a discrete-time plant $\zeta^+ = f(\zeta, u)$ for which the origin is a steady state, i.e., $0 = f(0, 0)$. For simplicity, we do not model any constraints, although RTIs can incorporate them naturally.

Consider the receding horizon problem at time l

$$\begin{aligned} \underset{r_{1:K-1}, s_{1:K}}{\text{minimize}} \quad & \sum_{k=1}^{K-1} \left[\begin{smallmatrix} s_k \\ r_k \end{smallmatrix}\right]^T Q \left[\begin{smallmatrix} s_k \\ r_k \end{smallmatrix}\right] + s_K^T R s_K \\ \text{subject to} \quad & s_1 = \overline{\zeta}_l \\ & s_{k+1} = f(s_k, r_k) \quad \forall k \in \{1, \dots, K-1\} \end{aligned} \tag{8}$$

where K denotes the horizon length, Q, R are positive definite stage and terminal cost matrices, and $\overline{\zeta}_l$ denotes the measured plant state at time l. Let $(\hat{r}^l, \hat{s}^l)$ denote the solution of (8) for the sampling instant l. Then, the feedback law at l is given by $u[l] = \hat{r}_1^l$. In other words, upon solving (8), the first control of the optimal policy is implemented at l. This is the key mechanism behind standard MPC.

RTI schemes approximate the solution of (8) by performing only a single iteration of an SQP method. Namely, let $z = (s, r, \lambda)$ and consider the Lagrangian of (8) at time

l defined as

$$L^l(z) := \sum_{k=1}^{K-1} \left[\begin{smallmatrix} s_k \\ r_k \end{smallmatrix}\right]^T Q \left[\begin{smallmatrix} s_k \\ r_k \end{smallmatrix}\right] + s_K^T R s_K + \lambda_1 (s_1 - \overline{\zeta}_l) + \sum_{k=1}^{K-1} \lambda_k \left(s_{k+1} - f(s_k, r_k)\right) .$$

An SQP iteration then takes the form

$$z^+ = z + \Delta z \tag{9}$$

where Δz solves the first-order condition

$$\nabla L^l(z) + \nabla^2_{zz} L^l(z) \Delta z = 0 .$$

where, in practice, the inverse of the Hessian $\nabla^2_{zz} L^l$ is often approximated.

Crucially, the receding horizon problem (8) at the next sample $l+1$ is warm-started with the shifted approximate of the previous sample. This procedure leads, under additional assumptions, to local stability of the scheme. ■

Example 2.3 elucidates several differences with respect to the optimal steady-state control problem (4) we wish to solve: First, the purpose of RTIs is primarily to drive a plant to the steady state at the origin, not seeking out an operating point with minimal cost (see Remark 2.2 below). Further, a full dynamic model f of the plant dynamics is required, and finally, the computational burden of solving the SQP iteration scales not only with the system dimension but also with the prediction horizon. Conversely, compared to Example 1.1, while stabilizing the system RTI also seeks to minimize a quadratic running stage cost.

Remark 2.2. Traditionally, linear and nonlinear MPC have been considered with the goal of stabilizing a plant and to track a precomputed setpoint or trajectory. The modern variation of *economic MPC* (Ellis et al., 2014; Faulwasser et al., 2018; Rawlings et al., 2012) studies the effects of incorporating an economic objective directly and thus not requiring an exogenous setpoint. This idea is very much in line with the topic of this article.

Economic MPC, however, still requires the solution of an optimal control problem at every iteration. Thus, it is computationally very expensive. Moreover, the optimal solution is not a-priori guaranteed to be a steady-state of the plant. This makes the stability analysis of economic MPC more involved and, in particular, still requires a full model of the plant dynamics. ■

3. Optimization Algorithms as Dynamical Systems

In recent years, renewed attention has been paid to the fact that many numerical optimization algorithms can be interpreted as dynamical systems. This perspective is essential to bridge the gap between algorithms and their implementation as feedback systems. In this section, we therefore give a tutorial-like overview of different *optimization dynamics* and related tools. Feedback aspects will be treated in the subsequent section.

3.1. Gradient Methods

Arguably as old as differential calculus itself are methods that seek out (local) minima of a function by following a descent direction. The most prototypical class of methods are gradient (or "steepest descent") schemes. Gradient methods exist in a wide variety of forms and contexts. Apart from $\mathbb{R}^n$, they can be defined on non-Euclidean manifolds (Absil et al., 2008; Brockett, 1988; Helmke & Moore, 1996) and on infinite-dimensional spaces (Ambrosio et al., 2005). In the absence of smoothness they can also be generalized to subgradient methods (Beck, 2017; Clarke, 1990). On top of that, steepest descent methods are generally available as continuous-time flows, discrete-time algorithms, or stochastic processes (Borkar, 2008). For ease of exposition and in line with the rest of this article, we will limit ourselves to continuous-time gradient flows and its variations.

At least for continuous-time negative gradient flows, convergence to minimizers appears plausible, if not tautological. However, closer inspection reveals important technical details, summarized in the following theorem.

Theorem 1. *Let $\Phi : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable with locally Lipschitz derivative $\nabla\Phi$ such that, for some $c \in \mathbb{R}$, the sublevel set $\Phi^{-1}(c) = \{x \in \mathbb{R}^n \mid \Phi(x) \leq c\}$ is compact. Then, the following statements hold for the gradient flow $\dot{x} = -\nabla\Phi(x)^T$:*

(i) Trajectories starting in $\Phi^{-1}(c)$ converge to the set of critical points, i.e., points $x^\star$ such that $\nabla\Phi(x^\star) = 0$.

(ii) If Φ is analytic or convex, then every solution starting in $\Phi^{-1}(c)$ converges to a single point.

(iii) Every asymptotically stable equilibrium is a strict and isolated minimizer, and every local minimizer is stable. If Φ is analytic or convex, then the set of all (strict) minimizers is equivalent to the set of (asymptotically) stable equilibria.

(iv) If Φ is twice continuously differentiable, the stability of an critical points is partially governed by its Hessian: If $\nabla^2\Phi(x^\star)$ is positive definite, $x^\star$ is a local minimizer and locally exponentially stable. If $\nabla^2\Phi(x^\star)$ has at least one negative eigenvalue, $x^\star$ is unstable.

Point (i) in Theorem 1 is a simple consequence of LaSalle invariance (Khalil, 2002). Namely, it is immediate that Φ is non-increasing along gradient trajectories. Compactness of $\Phi^{-1}(c)$ is generally required to preclude unbounded trajectories that escape to the horizon. Further, compactness of $\Phi^{-1}(c)$ together with the continuity of Φ guarantees lower boundedness and thus the existence of a minimizer.

In general, however, trajectories may not converge to a single point but to the entire set of critical points (Palis & De Melo, 1982). This pathological behavior is ruled out in (ii) if Φ is analytic, which guarantees the finite length of trajectories due to Lojasiewicz's inequality (Absil et al., 2005), or if Φ is convex.

In the absence of local Lipschitz continuity of $\nabla\Phi$, uniqueness of trajectories is not guaranteed and a distinction between *weak* and *strong equilibria* has to be made (Cortés, 2008; Hauswirth et al., 2020a).

As indicated in (iii), the stability of equilibria is related to their optimality, but an equivalence between the two is given only under additional assumptions. See Absil & Kurdyka (2006) for a proof and counterexamples. Finally, (iv) follows since, if Φ is smooth enough, the stability of an equilibrium $x^\star$ can be analyzed by investigating the linearized dynamics $\dot{x} = -\nabla^2\Phi(x^\star)\cdot x$

Example 3.1. A straightforward degree of freedom for gradient flows is the use of a *metric* $Q(\cdot)$ that maps every point $x \in \mathbb{R}^n$ to a square symmetric positive-definite matrix $Q(x)$. Under minor technical conditions (e.g., that Q has a uniformly bounded condition number; Hauswirth et al. 2020a) trajectories of the generalized gradient flow

$$\dot{x} = -Q(x)\nabla\Phi(x)^T \tag{10}$$

converge to the set of critical points of Φ. Namely, the use of Q, does neither alter the equilibrium points nor qualitative attractivity and stability, but only the trajectories. This feature is illustrated in Figure 5 which shows gradient trajectories for the same nonconvex potential function but for different metrics.

If Q is constant, (10) is equivalent to a Euclidean gradient flow in linearly transformed coordinates. Namely, if $Q = V^T\Lambda V$ is the eigenvalue decomposition of Q, we can define the coordinate transformation $y := \sqrt{\Lambda}^{-1}Vx$ where $\sqrt{\Lambda}$ denotes the diagonal matrix of square root eigenvalues. Then, it can be easily shown that (10) is equivalent to $\dot{y} = -\nabla\hat{\Phi}(y)^T$ with $\hat{\Phi}(y) := \Phi(V^T\sqrt{\Lambda}y)$.

If Q is not constant, then, from differential geometric viewpoint, Q can be interpreted as a matrix representation of a non-Euclidean *Riemannian metric* on $\mathbb{R}^n$. Thus, Q endows $\mathbb{R}^n$ with a non-flat geometry (Lee, 1997).

As a special case of a non-constant metric, assume that Φ is twice differentiable and strongly convex. Then, Q can be chosen to be the inverse of the Hessian $\nabla^2\Phi$. This results in a continuous-time version of the classic *Newton method*, also referred to as "Newton Gradient Flows" (Jongen et al., 2001, Chap. 9.3). However, unlike the iterative Newton method, the continuous-time flow does not exhibit an inherently faster convergence rate compared to other gradient flows. For the inverted Hessian metric, convergence is merely *isotropic*, i.e., the same from all directions. This property counteracts ill-conditioning of the objective function as illustrated in Figure 6.

Finally, if Q is sparse, then the *sparsity* pattern induces algebraic structure that can often be exploited for the purpose of a distributed implementation. ■

Example 3.2. A relatively easy and widely applicable way to incorporate constraints into an optimization problem are the addition of penalty (or regularizing) or barrier terms to the objective. For a constraint of the form

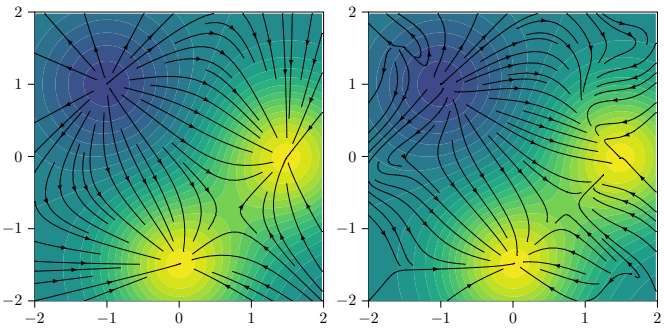


Figure 5: Gradient trajectories for a non-convex objective function. Trajectories under the Euclidean metric (left) and a generic variable metric (right) differ significantly. The critical points (i.e., the minima, maximum, and saddle-point) and their stability properties are unaffected by the choice of metric.

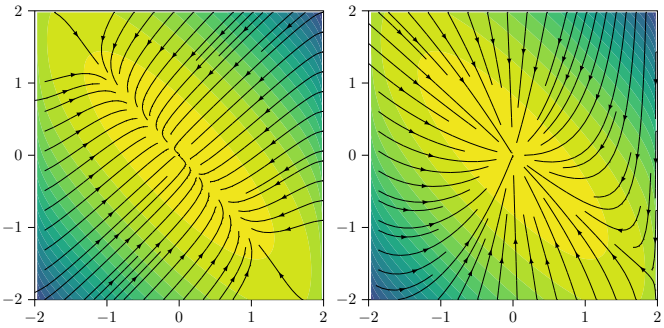


Figure 6: Gradient trajectories for strongly convex, but ill-conditioned objective. The trajectories under the Euclidean metric (left) quickly approach a subspace on which the objective is almost flat, and then converge only slowly to the global optimizer. Trajectories under the "Newton metric" (right), approach the global optimizer isotropically, unaffected by the ill-conditioning of the objective.

$g(x) \leq 0$ where $g : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable, a common penalty function is for example the squared 2-norm of the constraint violation vector, i.e., $\phi(x) = \frac{\rho}{2}\|\max\{g(x), 0\}\|^2$ where $\rho > 0$ denotes a scaling parameter. Many variations, including different norms on constraint violations are possible. The common feature of penalty function lies in the fact that they technically allow for constraint violations, i.e., minimizers of a penalty-augmented cost function $\Phi(x)+\phi(x)$ do not generally satisfy $g(x) \leq 0$. A notable exception are so-called *exact penalty methods* that transform a constrained optimization problem into an unconstrained one without changing the location of minimizers, albeit at the expense of smoothness or other technical drawbacks (Di Pillo & Grippo, 1989).

Barrier functions, on the other hand, can be used to apply constraints strictly, i.e., without allowing for any violation. For this purpose, a barrier function $\psi(\cdot)$ for the constraint $g(x) \leq 0$ needs to be such that for $x \to x^\star$ with $g(x^\star) = 0$ we have $\psi(x) \to \infty$. A common example satisfying this condition are negative log-barriers of the form $\psi(x) = -\frac{1}{\mu}\log(g(x))$ which are important for interior-point methods for constrained convex programming (Nesterov & Nemirovskii, 1994).

The effects of penalty and barrier terms on gradient flows is illustrated in the top panels of Figure 8, respectively. Both types of additional cost terms can enforce constraints only approximately. In general, steeper penalties or barriers (i.e., $\rho \to \infty$ or $\mu \to \infty$) will lead to more precise results. However, excessively steep augmentations can lead to stability issues when implemented numerically or in closed loop with a dynamical system (see Section 4.2). ■

3.2. Projected Gradient Flows

To enforce unilateral (i.e., inequality) constraints it is often possible to rely on projection mechanisms. In a computational context, this is particularly true if the projection onto a given constraint set is easy to evaluate numerically.

Consider the constrained optimization problem

$$\text{minimize} \quad \Phi(x) \qquad \text{subject to} \quad x \in \mathcal{X}\,, \tag{11}$$

where $\mathcal{X} \subset \mathbb{R}^n$ is closed convex and non-empty, and Φ is a convex cost function. The classical *projected gradient descent* to solve this problem takes the form

$$x^{k+1} = P_{\mathcal{X}}\left(x^k - \alpha^k \nabla\Phi(x^k)^T\right)\,. \tag{12}$$

where $P_{\mathcal{X}}(y) := \arg\min_{x\in\mathcal{X}} \|x-y\|$ denotes the Euclidean minimum norm projection onto $\mathcal{X}$, and $\{\alpha^k\}$ is a sequence of step sizes.

By choosing infinitesimally small step-sizes, the continuous-time limit of (12) is a *projected gradient flow*

$$\dot{x} = \Pi_{\mathcal{X}}\left[-\nabla\Phi(x)^T\right](x) \tag{13}$$

where $\Pi_{\mathcal{X}}[v](x)$ denotes the projection of the vector v onto the tangent cone $T_{\mathcal{X}}(x)$ of $\mathcal{X}$ at x, i.e., $\Pi_{\mathcal{X}}[v](x) = \arg\min_{w\in T_{\mathcal{X}}(x)} \|v-w\|$. As illustrated in Figure 7, the tangent cone is the set of all directions starting at x which point inward into the set $\mathcal{X}$. For convex sets, it takes the general form $T_{\mathcal{X}}(x) = \mathrm{cl}\{d \,|\, d = \alpha(y-x), y \in \mathcal{X}, \alpha \geq 0\}$. For a comprehensive discussion (including for non-convex sets) see Rockafellar & Wets (2009).

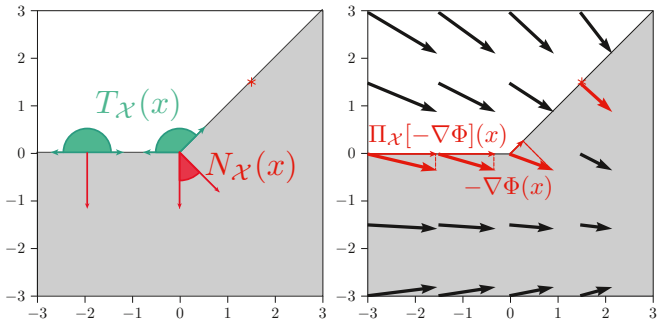


Figure 7: Left: Examples of tangent and normal cones of a set $\mathcal{X}$ (gray area is infeasible). Right: Projected vector field onto the tangent cone. See Figure 8e for the resulting projected gradient trajectory.

Figure 8e illustrates the qualitative behavior of projected gradient flows. Namely, in the interior of the feasible set, trajectories follow the gradient direction whereas at the boundary, trajectories follow the steepest *feasible* descent direction. It is also evident that, compared to penalty or barrier approaches, projected gradient flows are inherently discontinuous systems, and their study requires tools from non-smooth analysis (Aubin & Cellina, 1984; Clarke, 1990; Hauswirth et al., 2020a).

Remark 3.1. As mentioned above, the continuous-time model (13) can be obtained from (12) in the limit as $\alpha \to 0^+$ where α is a constant step size. Conversely, (12) can be interpreted as a *forward Euler* discretization of (13). Analogously, a *backward Euler* scheme for (13) takes the implicit form $x^{k+1} = P_{\mathcal{X}}\left(x^k - \alpha^k \nabla\Phi(x^{k+1})^T\right)$. Comparing optimality conditions, it easy to show that this discretization is a special case of the more general *proximal-point algorithm* (Beck, 2017, Chap. 27.1).

In contrast to the discrete-time system (12), continuous-time projected gradient flows can be generalized extensively. In particular, convexity of $\mathcal{X}$ is not required. More precisely, whereas $P_{\mathcal{X}}$ requires $\mathcal{X}$ to be convex to be well-defined, $\Pi_{\mathcal{X}}$ is generally well-defined since $T_{\mathcal{X}}(x)$ is non-empty and closed convex for a large class of non-convex sets called *(Clarke) regular*. ■

Remark 3.2. Convex projected gradient flows also fall into the category of *subgradient flows*: If $\mathcal{X}$ is convex, we may consider the minimization of $\Phi(x) + I_{\mathcal{X}}(x)$ where $I_{\mathcal{X}}$ denotes the indicator function of the set $\mathcal{X}$. Since $I_{\mathcal{X}}$ is not differentiable, instead of a gradient flow, we need to resort to the *subgradient inclusion* $\dot{x} \in -\nabla\Phi(x)^T - N_{\mathcal{X}}(x)$. (In particular, the subgradient of $I_{\mathcal{X}}$ is given by the normal cone $N_{\mathcal{X}}(x)$ of $\mathcal{X}$ at x, i.e., $\partial I_{\mathcal{X}}(x) = N_{\mathcal{X}}(x)$.) This differential inclusion is also referred to as *differential variational inequality* (Aubin & Cellina, 1984) and can be shown to be equivalent to (13) (i.e., admit the same trajectories). ■

Projected gradient flows inherit various properties from their unconstrained counterparts. For instance, similarly to Theorem 1, trajectories of (13) converge to the set of critical points (in this case, Karush-Kuhn-Tucker points of (11)), and stability and optimality can be related similarly to (iii) in Theorem 1.

Analogously to unconstrained gradient flows discussed in Example 3.1, projected gradient flows can also be defined using a variable metric. This modification, however, requires a generalization of $\Pi_{\mathcal{X}}$ to take into account the effects of the metric. Namely, instead of $\Pi_{\mathcal{X}}[v](x)$, the projection operator

$$\Pi^Q_{\mathcal{X}}[v](x) := \arg\min_{w\in T_{\mathcal{X}}(x)} (w-v)^T Q(x)(w-v) \tag{14}$$

has to be used, where $Q(\cdot)$ is a metric. Thus an "oblique" projected gradient flow takes the fom

$$\dot{x} = \Pi^Q_{\mathcal{X}}\left[-Q(x)\nabla\Phi(x)^T\right]\,.$$

This degree of freedom allows, in particular, the definition of *projected Newton flows* and will also be applied in Examples 4.7 and 4.8. See Hauswirth et al. (2020a, 2016) for precise statements.

Projected gradient flows are a special case of more general *projected dynamical systems* of the form $\dot{x} = \Pi_{\mathcal{X}}[f(x)](x)$ where f is a general vector field and not necessarily the gradient of a function. This general formalism is particularly useful in game-theoretic contexts (Nagurney & Zhang, 1996) and for saddle-point flows for constrained optimization problems discussed below.

3.3. Primal-Dual Saddle-Point Dynamics

Simply speaking, under weak technical assumptions, solutions of a constrained optimization problem are saddle-points of the associated *Lagrangian*. For this reason, dynamical systems that seek out saddle-points rather than extrema of a function are of particular interest for constrained optimization.

As a general yet basic setup, consider a differentiable function $L(x,\mu)$ that is convex in x for every μ, concave in μ for all x, and either strictly convex in x or strictly concave in μ. Then trajectories of the system

$$\dot{x} = -\nabla_x L(x,\mu)^T \qquad \dot{\mu} = \nabla_\mu L(x,\mu)^T \tag{15}$$

converge to a saddle-point of L, i.e., a point $(x^\star,\mu^\star)$ such that $L(x,\mu^\star) \geq L(x^\star,\mu^\star) \geq L(x^\star,\mu)$ for all x and all μ. In particular, (15) consists of a gradient descent in the *primal* variables x and a gradient ascent in the *dual* variables μ.

Historically, *saddle-point flows* have primarily been studied in the context of nonlinear circuit analysis (Brayton & Moser, 1964; Smale, 1972), but their potential for optimization has been observed even before that (Arrow et al., 1958; Kose, 1956). Although saddle-point flows have been studied throughout the years (Bloch et al., 1992; Venets, 1985), they have recently become a topic of intense study due their importance in the context of distributed network optimization (see Example 4.1 below and Cortés & Niederländer 2019; Feijer & Paganini 2010).

Example 3.3. Consider the linearly constrained problem

$$\text{minimize} \quad \Phi(x) \qquad \text{subject to} \quad Ax = b \tag{16}$$

where $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$, and Φ is strictly convex. The Lagrangian of (16), given by

$$L(x,\mu) := \Phi(x) + \mu^T(Ax - b)$$

is strictly convex in x and linear (thus concave) in μ. Consequently, the saddle-point flow (15) is globally convergent and takes the form

$$\dot{x} = -\nabla\Phi(x)^T - A^T\mu \qquad \dot{\mu} = Ax - b\,. \tag{17}$$

Note in particular that any equilibrium $(x^\star,\mu^\star)$ of (17) satisfies $0 = \nabla\Phi(x^\star)^T + A^T\mu^\star$ and $0 = Ax^\star - b$ and thus the Karush-Kuhn-Tucker condition which are (for convex problems) necessary and sufficient for optimality of $x^\star$.

Finally, when A is sparse and Φ is separable, then (15) is amenable to a distributed implementation. This property crucial for networked applications and will be further this discussed in Examples 4.1 and 4.2 for the control of communication networks and power systems. ■

Like gradient flows, saddle-point flows can be modified through the use of positive definite metrics Q_p, Q_d as

$$\dot{x} = -Q_p\nabla_x L(x,\mu)^T \qquad \dot{\mu} = Q_d\nabla_\mu L(x,\mu)^T\,. \tag{18}$$

Often, Q_p and Q_d are chosen to be constant diagonal matrices that speed up or slow down convergence in specific directions. Interestingly, in the limit case $Q_p \gg Q_d$, one recovers a differential-algebraic system

$$x^\star(\mu) = \arg\min_x L(x,\mu) \qquad \dot{\mu} = Q_d\nabla_\mu L(x^\star(\mu),\mu)^T\,,$$

which is a continuous-time *dual ascent*. This idea of replacing the primal gradient flow with an explicit (algebraic) minimization can also be applied partially to a subset of variables. This possibility will be applied in Example 4.2.

Differentiability of the saddle-function L is not generally required. In fact, (15) can be generalized to include projections on both x and/or μ (Cherukuri et al., 2017a, 2016, 2017b; Hauswirth et al., 2020f; Stegink et al., 2018). This possibility is particularly important to deal with inequality constraints (see Example 3.4 below). Even more generally, (15) can be formulated in terms of subgradients if L is not differentiable (Goebel, 2017; Venets, 1985).

Convergence proofs for (15) and its generalization usually rely on the monotonicity of the vector field $[-\nabla_x L(x,\mu)\ \nabla_\mu L(x,\mu)]$. This type of argument, however, does not generalize beyond convex-concave saddle-point flows. Even relaxing strict convexity or concavity requirement of either x or μ, respectively, is problematic since this may lead to oscillations (Holding & Lestas, 2014). However, these oscillations can be eliminated with a simple augmentation term (Hauswirth et al., 2020f).

The following inequality-constrained example illustrates several important aspects of saddle-point flows: Building upon the previous Example 3.3, we show how the projected dynamical systems discussed in Section 3.2 can be used to define projected saddle-point flows for inequality-constrained optimization problems, and we discuss augmentations based on penalty terms to improve convergence.

Example 3.4. Instead of (16), consider the problem

$$\begin{aligned} \text{minimize} \quad & \Phi(x) \\ \text{subject to} \quad & x \in \mathcal{X} \\ & g(x) \leq 0\,, \end{aligned} \tag{19}$$

where Φ and g are convex (but not necessarily strictly convex) and continuously differentiable. Further, let $\mathcal{X} \subset \mathbb{R}^n$ be non-empty and closed convex. We may define the *partial* Lagrangian $L : \mathcal{X}\times\mathbb{R}^m_{\geq 0} \to \mathbb{R}$ of (19) as

$$L(x,\mu) := \Phi(x) + \mu^T g(x)\,, \tag{20}$$

and note that μ must lie in the non-negative orthant $\mathbb{R}^m_{\geq 0}$ because it is associated with an inequality constraint.

To find a saddle-point of L on the set $\mathcal{X}\times\mathbb{R}^m_{\geq 0}$ we use the continuous-time projection formalism introduced in Section 3.2 for projected gradient flows. Namely, we consider the *projected saddle-point flow*

$$\dot{x} = \Pi_{\mathcal{X}}\big[\overbrace{-\nabla_x L(x,\mu)^T}^{-\nabla\Phi(x)^T-\nabla g(x)^T\mu}\big](x) \tag{21}$$

$$\dot{\mu} = \Pi_{\mathbb{R}^m_{\geq 0}}\big[\underbrace{\nabla_\mu L(x,\mu)^T}_{g(x)}\big](\mu) \tag{22}$$

where $\Pi_{\mathcal{X}}[w](x)$ and $\Pi_{\mathbb{R}^m_{\geq 0}}[v](x)$ project w and v onto the tangent cone of $\mathcal{X}$ and on the non-negative orthant $\mathbb{R}^m_{\geq 0}$ at x and μ, respectively. Consequently, trajectories of (21) cannot leave $\mathcal{X}\times\mathbb{R}^m_{\geq 0}$.

Importantly, two different constraint enforcement mechanisms are at play in (21): On one hand, the constraint $x\in\mathcal{X}$ is enforced directly by projection, similarly to the projected gradient flow in Section 3.2. The constraint $g(x)\leq 0$ on the other hand is enforced by dualization. Namely, the dual variable μ is updated in response to a constraint violation $g(x)>0$ and converges to a dual solution of (19).

As shown, e.g., by Goebel (2017), under weak technical assumptions and if Φ is strictly convex, trajectories of (21) are guaranteed to converge to a Karush-Kuhn-Tucker point (and thereby to a global optimizer) of (19).

If Φ is non-strictly convex, then trajectories of (21) converge to the optimizer of (19) if, instead of L, a penalty augmented Lagrangian of the form

$$L^a(x,\mu) := \Phi(x) + \mu^T g(x) + \phi(x) \tag{23}$$

is used, where ϕ is a penalty function, as introduced in Example 3.2. Namely, ϕ is often chosen to be of the form $\phi(x) := \frac{\rho}{2}\|\max\{g(x),0\}\|^2$, but other constructions are possible to guarantee asymptotic convergence.

If Φ or g are non-convex, convergence of trajectories to a saddle-point is generally not guaranteed. For this reason, Bernstein et al. (2019); Dall'Anese & Simonetto (2018); Tang et al. (2018a,b) have used an additional regularization on the dual variables of the form

$$L^b(x,\mu) := \Phi(x) + \mu^T g(x) + \phi(x) + \tfrac{\hat{\rho}}{2}\|\mu\|^2\,. \tag{24}$$

Using this modification and a large enough $\hat{\rho}$, the corresponding saddle-point flow is convergent, even with exponential stability guarantees. However, saddle-points of L^b do generally not coincide with Karush-Kuhn-Tucker points of (19) anymore. In particular, notice that the dual update takes the form $\dot{\mu} = \Pi_{\mathbb{R}^n_{\geq 0}}[g(x)+\hat{\rho}\mu]$ and thus, at an equilibrium, it holds that $\mu_i^\star(g_i(x^\star)+\hat{\rho}\mu_i^\star) = 0$. In the absence of dual regularization, the same equilibrium condition simplifies to $\mu_i^\star g_i(x^\star) = 0$ which is the *complementary slackness* part of the KKT optimality conditions of (19). ■

Example 3.4 yields two insights into saddle-point flows: First, projections can be used in two different ways to enforce inequality constraints. Namely, we can either constrain the primal variables directly (as with projected gradient flows with $\Pi_{\mathcal{X}}$) or integrate constraint violations to compute dual variables. In this second case, the projection $\Pi_{\mathbb{R}^n_{\geq 0}}$ ensures the non-negativity of the dual variables. Second, it is possible to augment the Lagrangian saddle-point function to improve convergence. However, regularizing the dual variables will in general modify the location of saddle-points.

Remark 3.3. We note that penalty functions (Example 3.2) and unprojected saddle-point flows (as in Example 3.3) exhibit strong parallels with classical PI-controllers: Assume that a plant exhibits nonlinear dynamics with affine inputs and outputs of the form

$$\dot{x} = -\nabla\Phi(x)^T + A^T u \qquad y = Ax\,.$$

Applying a proportional controller $u = -\rho(y-b)$ realizes a closed-loop system of the form

$$\dot{x} = -\nabla\Phi(x)^T - \rho A^T(Ax-b)$$

which can easily be identified as a gradient flow of the penalty-augmented cost $\Phi(x)+\frac{\rho}{2}\|Ax-b\|^2$.

An integral controller $\dot{u} = (y-b)$, on the other hand, leads exactly to the saddle-point flow (17).

Hence, combining a penalty function approach and a saddle-point flow leads to an augmented saddle-point flow that is analogous to a PI-controller, i.e., we get

$$\begin{aligned}\dot{x} &= -\nabla\Phi(x)^T - \rho A^T(Ax-b) - A^T u\\ \dot{u} &= Ax-b\,.\end{aligned}$$

which is a saddle-point flow for the augmented Lagrangian

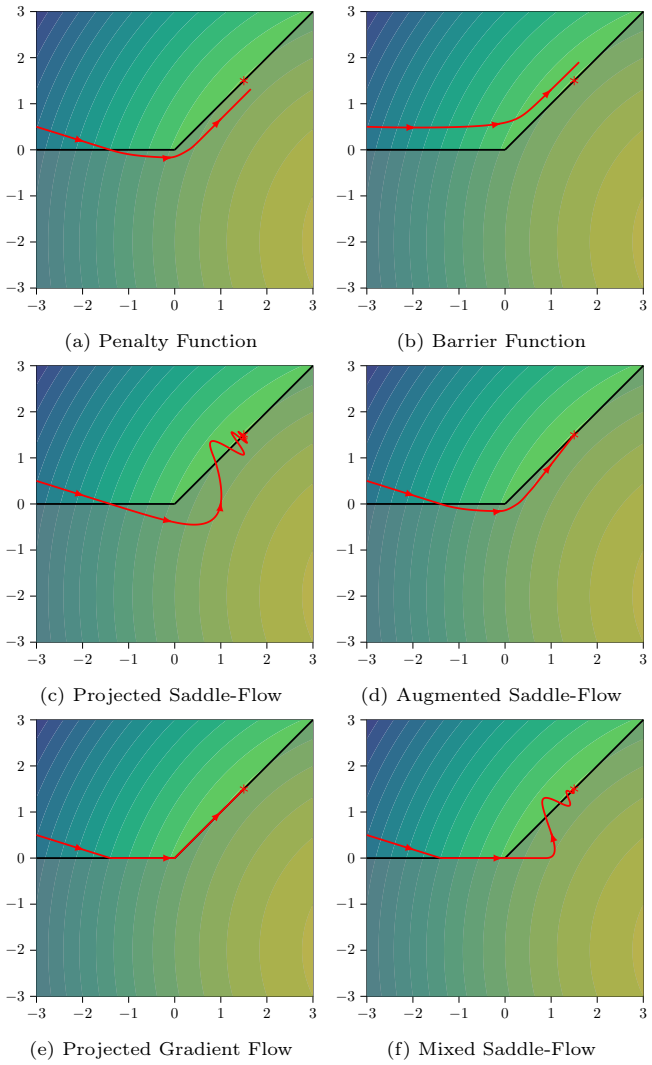
$$L^a(x,u) := \Phi(x) + u^T(Ax-b) + \tfrac{\rho}{2}\|Ax-b\|^2\,. \qquad ■$$

All in all, saddle-point flows have proven to be very versatile and particularly useful for distributed optimization where agents share certain constraints. However, convergence and stability results for non-convex problems are not generally available. Moreover, even for convex problems, tuning can be difficult, especially for nonlinear problems. Suboptimal parameter choices can lead to severely under- or over-damped transients that may venture far outside the feasible domain, which is undesirable in online and closed-loop applications. This problem gets only more challenging for high-dimensional and ill-conditioned problems.

3.4. Comparison of Constraint Enforcement Mechanisms

In the previous subsections we have presented several ways to design dynamics that can solve constrained optimization problems. Their characteristic behaviors and different combinations are illustrated in Figure 8. In the following, we summarize, contrast, and compare these different mechanisms.

Penalty and barrier functions, as discussed in Example 3.2, can be used to transform problems into unconstrained problems which can then be tackled with a simple



(a) Penalty Function (b) Barrier Function

(c) Projected Saddle-Flow (d) Augmented Saddle-Flow

(e) Projected Gradient Flow (f) Mixed Saddle-Flow

Figure 8: Behavior of different constraint enforcement mechanisms
All panels show the minimization of a quadratic function subject to two constraints $x_2 \geq 0$ and $x_2 \geq x_1$ (the grayed out area is infeasible). Penalty (a) and barrier (b) functions allow for smooth outer and inner approximations of constraints with an unconstrained gradient flow (Example 3.2). Saddle-point flows (c) enforce constraints only asymptotically by integrating constraint violation over time, but are often amenable to a distributed implementation. Augmenting saddle-point flows with a penalty term can improve convergence (d) as explained in Example 3.4. Projected gradient flows (e) enforce constraints directly by projection, which results in non-smooth trajectories, but their implementation is not immediate (for one possibility see Example 4.6 below). Individual constraints can also be enforced with a combination of these mechanisms, e.g., as in (f) with a projection for $x_2 \geq 0$ and dualization (saddle-point flow) for $x_2 \geq x_1$, as in Example 3.4.

gradient flow. However, both approaches by themselves can enforce constraints only approximately. Barrier functions achieve an "inner" (i.e., conservative) approximation, whereas penalty functions generally allow for a small constraint violation and thus constitute an "outer" approximation. Both approaches are widely applicable (under minor technical assumptions) and do not require convexity of the constraints. However, theoretical guarantees often rely on additional assumptions and, because of practical considerations, penalty and barrier function cannot be chosen arbitrarily steep.

Constraint enforcement by (infinitesimal) projection, as for continuous-time projected gradient flows, is mathematically well-posed and works in very general settings. In particular, convexity of the constraint set is not generally required (as opposed to discrete-time projected gradient descent). Furthermore, constraints are represented exactly and satisfied at all times. However, these continuous-time discontinuous dynamical systems are often not directly implementable. Instead, discrete-time approximations, for example, rely upon the fact that the numerical projection onto the feasible constraint set are computationally inexpensive. In continuous-time, projected dynamical systems can be implemented by exploiting physical saturation and applying anti-windup control. This possibility will be further discussed in Section 4.3 and, particularly, in Example 4.6. Another possibility, to approximate projected gradient flows, is by appropriate discretizations which, in contrast to (12), do not require an explicit projection $P_{\mathcal{X}}$ onto the feasible set. Such an algorithm will be discussed in Section 4.3.2.

Dualization of constraints leads to saddle-point flows where dual variables are computed by integrating the constraint violation over time. Hence, transient constraint violation are generally unavoidable. For an inequality constraint, the corresponding dual variable must be kept nonnegative by projection. Formal convergence guarantees are available only for convex problems (or with dual augmentation which alters equilibrium points) and, in practice, this method is difficult to tune, but often allows for distributed implementation that requires only the communication of dual multipliers.

In theory, each constraint (in functional form) can be enforced with one these mechanisms independently of the other constraints. For example, in Figure 8f, the constraint $x_2 \geq 0$ is enforced by projection whereas $x_2 \geq x_1$ is dualized. As we will see in the forthcoming section, this freedom of choice is particularly useful in control setups where the real-world nature of constraints can dictate the appropriate enforcement mechanism. For instance, barrier functions may be considered for constraints that may not be violated under any circumstances. Constraints that are naturally enforced by physical saturation, mechanical constraints or similar are best represented by projections. Dualization in combination with a penalty term is particularly helpful to enforce constraints asymptotically and often allow for distributed implementations.

As we will see in Section 4.3, all of these constraint enforcement methods can be applied in an online feedback setup, albeit their suitability depends on the specific constraint type, problem size and available model information.

3.5. Time-Varying Online Optimization

A topic that is central to online optimization, in open or closed loop, is the study of problems that vary over time.

In recent years, this topic has garnered significant interest because of its relevance to many applications in control, robotics, machine learning, and others (Dall'Anese et al., 2020; Simonetto et al., 2020). The focus has been the development of algorithms that can track the solution of a non-stationary optimization problem with performance guarantees. Historically, two perspectives can be distinguished:

On the one hand, Besbes et al. (2015); Hall & Willett (2015); Jadbabaie et al. (2015); Lesage-Landry et al. (2020); Shi et al. (2020); Zinkevich (2003) and others frame time-varying optimization as an *iterative learning problem* in following sense: At every iteration k an agent chooses an action x_k and subsequently a convex function Φ_k is revealed. The agent's goal is to minimize her *regret*, i.e., some measure of accumulated suboptimality.

On the other hand, Rahili & Ren (2017); Rahili et al. (2015); Simonetto et al. (2017); Tang et al. (2018a) are inspired more by control theory and describe time-varying optimization as a *tracking problem* whereby an optimization algorithm defines an time-varying solution map $t \mapsto x^\star(t)$ that needs to be followed as closely as possible by the online optimization scheme.

Both viewpoints share a common base: First, convexity is generally assumed to guarantee the existence of a unique minimum and, in the case of strong convexity, a unique minimizer. Exceptions are Subotić et al. (2020); Tang et al. (2018a) where results for non-convex optimization problems are presented. Second, to give meaningful performance guarantees, some sort of "bounded variation" in the optimization problem has to be assumed. A common starting point for this purpose is to assume that the rate of change of the optimizers is bounded by a known constant. This assumption is relaxed in Subotić et al. (2020) which shows how, in special cases, the rate of change of the optimizer can be bounded using information about the objective and the constraint functions only.

Depending on the application, the time-varying optimization problem might be unconstrained (Popkov, 2005), constrained to a stationary set (Hall & Willett, 2015; Jadbabaie et al., 2015; Mokhtari et al., 2016; Zinkevich, 2003), or have time-varying constraints (Rahili & Ren, 2017; Rahili et al., 2015; Subotić et al., 2020; Tang et al., 2018a). Time-varying constraints have been dealt with using barrier functions (Fazlyab et al., 2016, 2018) or perturbed sweeping processes (Hauswirth et al., 2018; Subotić et al., 2020; Tang et al., 2018a).

Roughly speaking, algorithms for time-varying optimization can be divided into *running* algorithms that do not incorporate any information about the evolution of the problem (Bastianello et al., 2020a; Bernstein et al., 2019; Popkov, 2005; Simonetto, 2017; Simonetto & Leus, 2014; Tang et al., 2018a) and *predictive* schemes that exploit some knowledge or estimate about the change in the optimization problem (Bastianello et al., 2020b; Fazlyab et al., 2018; Lesage-Landry et al., 2020; Simonetto & Dall'Anese, 2017; Simonetto et al., 2017).

The following examples generalize Example 1.1 to a time-varying setting and illustrate a running and a predictive scheme.

Example 3.5. Consider the same setup as in Example 1.1. Namely, let a physical plant be characterize by the steady-state input-output map $y = h(u) + d(t)$. We now assume that $d(t)$ is time-varying and (Lebesgue) measurable. Consequently, we wish to track the solution of

$$\text{minimize} \quad \tilde{\Phi}(u,t) := \Phi(h(u) + d(t)) \,. \tag{25}$$

Without modifying the controller we get the (non-autonomous) closed-loop system

$$\dot{u} = -\nabla_u \tilde{\Phi}(u,t)^T \tag{26}$$

$$= -\nabla h(u)^T \nabla \Phi(y)^T \qquad y = h(u) + d(t) \tag{27}$$

which is a running algorithm to solve (25).

Assume that $\tilde{\Phi}$ is β-strongly convex for every t and consequently has a unique global minimizer $u^\star(t)$ for every t. Further, assume that $\|u^\star(t) - u^\star(t')\| \leq \ell \|t - t'\|$ for all t, t'. In other words, the unique optimizer is ℓ-Lipschitz.

The quantity ℓ can sometimes be bounded from problem parameters (Subotić et al., 2020). For instance, if it is known that d is ℓ_d-Lipschitz in t and Φ is β'-strongly convex, then, the estimate $\ell \leq \beta' \ell_d$ holds.

Exploiting strong convexity and Cauchy-Schwarz inequality, the distance between $u(t)$ of (26) and $u^\star(t)$ can be shown to differentiable for almost all t and satisfy

$$\begin{aligned} &\tfrac{d}{dt} \tfrac{1}{2} \|u(t) - u^\star(t)\|^2 \\ &\quad \leq \langle \dot{u}(t), u(t) - u^\star(t) \rangle + \ell \|u(t) - u^\star(t)\| \\ &\quad \leq -\beta \|u(t) - u^\star(t)\|^2 + \ell \|u(t) - u^\star(t)\| \,. \end{aligned}$$

Consequently, $\|u(t) - u^\star(t)\|$ is decreasing as long as $\|u(t) - u^\star(t)\| > \ell/\beta$. It follows from standard invariance arguments that, as $t \to \infty$, $u(t)$ will be ℓ/β-close to $u^\star(t)$. A similar statement holds for the discrete-time case. ■

Example 3.6. To illustrate a continuous-time predictive algorithm, consider the same setup as in Example 3.5 and assume, in addition, that Φ is twice continuously differentiable and the time derivative $\dot{d}(t)$ is available (e.g., can be estimated using finite differences). Then, we may consider the control law

$$\begin{aligned} \dot{u} &= -\nabla h(u)^T \nabla_u \Phi(y)^T - \nabla h(u)^T \nabla^2_{yy} \Phi(y) \dot{d}(t) \\ y &= h(u) + d(t) \end{aligned}$$

which reduces to

$$\dot{u} = -\nabla_u \tilde{\Phi}(u,t)^T - \tfrac{d}{dt} (\nabla_u \tilde{\Phi})^T (u,t) \,.$$

One may consider the time-varying Lyapunov function $W(u,t) := \frac{1}{2} \|\nabla_u \tilde{\Phi}(u,t)\|^2$. It holds that

$$\begin{aligned} \tfrac{d}{dt} W(u(t),t) &= \nabla_u W(u(t),t) \dot{u}(t) + \nabla_t W(u(t),t) \\ &= \|\nabla_u \tilde{\Phi}(u(t),t)\|^2 < 0 \,. \end{aligned}$$

for all t and all $u(t) \neq u^\star(t)$. This fact can be used to guarantee zero tracking error as $t \to \infty$. ■

4. Online Feedback-Based Optimization

We now turn to the key idea and main topic of this article—the implementation of optimization algorithms in closed loop with physical systems. The unconstrained feedback gradient flow from Example 1.1 in the introduction will serve as a prototypical example for this approach. However, our ultimate objective is to devise feedback controllers to robustly steer a dynamic plant (1) to an operating condition that solves the more general constrained nonlinear optimization problem (4). For this purpose, we use the optimization dynamics and mechanisms presented in Section 3 as blueprints for the design of nonlinear integral multi-input-multi-output controllers.

Early designs for this type of feedback-based optimization can be traced back to Brunner et al. (2012); Jokic et al. (2009). Both works consider steering a dynamical plant towards the solution of a constrained convex optimization problem and resort to saddle-point formulations to enforce constraints. While Jokic et al. (2009) proposes controllers using so-called "complementarity integrators", Brunner et al. (2012) considers a smooth saddle-point flow variation and applies backstepping to design the feedback controller. Hence, compared to the general problem (4), Brunner et al. (2012); Jokic et al. (2009) study a simple convex optimization problem in the output variable y only, i.e., without constraints and cost on u.

Generally speaking, almost any optimization algorithm in the form of a dynamical system can be defined as open systems and interconnected with physical systems, but not every such control design makes sense in the real world. Notably, physical realizations need to be robust, they are constrained by the computational and communication infrastructure, and they are limited by the availability of accurate model data and physical measurements.

In the following subsections, after reviewing existing literature, we therefore present the proposed feedback-based optimization approach by focusing on two robustness aspects: First, we survey approaches to guarantee *closed-loop stability*. Second, we review the possibilities to *robustly enforce constraints* in closed-loop optimization, showing how the "abstract" algorithmic mechanisms in Section 3 can be mapped to the specific nature of constraints.

In this section, we also provide three application examples that illustrate specific problem domains where feedback-based optimization has been considered to design and retrofit complex high-dimensional control systems. The following two of these examples concern the historical problems of congestion avoidance in communication networks and frequency regulation in AC power grids. At the end of the section, we further study the timely application of optimal reserve dispatch in electricity grids in the presence of intermittent renewable power infeed. In contrast to the previous examples, this last application exhausts the generality of (4), combines the main ideas from this article, and is amenable to further variations and extensions.

Example 4.1 (Network Congestion Control). Arguably one of the largest man-made distributed feedback systems is formed by the congestion management mechanisms at the heart of the Internet (Low et al., 2002). These protocols manage the allocation of link capacities to individual connections in highly dynamic environments, for heterogeneous agents, and subject to real-world imperfections such as delays, among others (Paganini et al., 2005; Tang et al., 2007; Vinnicombe, 2002; Wen & Arcak, 2004). Deterministic, continuous-time flow models have been particularly successful in explaining these protocols, by providing an optimization-based perspective in terms of network utility maximization (Kelly et al., 1998; Low, 2017; Low & Lapsley, 1999; Shakkottai & Srikant, 2007), and enabling improved designs (Wei et al., 2006, 2013; Zargham et al., 2013).

Given a communication network, let a set of N sources share M links. The set of links used by each to communicate to its destination are collected in the routing matrix $R \in \mathbb{R}^{N\times M}$ in the sense that $R_{ij} = 1$ if link j is used by source i and $R_{ij} = 0$ otherwise. Each link j in the network has an associated congestion measure μ_j (also referred to as *price*) which may describe queuing delays or packet loss probabilities. Each link j has a finite capacity c_j. Each source i has a controllable *source rate* x_i, e.g., in the form of its *window size*. The different source rates define a vector of aggregate flows for each line given by $y = Rx$ where each row of R corresponds to a link. Conversely, sources are assumed to have access to their respective *aggregate price* $p = R^T\mu$. In practice, each source can estimate the total price over its own path by measuring delays, estimating the packet loss probability, or detecting the presence of specific "congestion markers" on its packets.

The simplest model to describe the *link dynamics* is given by the *projected gradient* flow

$$\dot{\mu}_j = \Pi_{\mathbb{R}^M_{\geq 0}}[y_j - c_j](\mu_j) \tag{28}$$

where

$$\Pi_{\mathbb{R}^M_{\geq 0}}[y_j - c_j](\mu_j) = \begin{cases} y_j - c_j & \mu_j = 0 \text{ and } y_j > c_j, \\ 0 & \text{otherwise.} \end{cases}$$

In other words, the price increases when the link is overloaded (and package loss occurs) and decreases whenever it is not overloaded, but the price never drops below zero.

Source controllers can often be modeled as $\dot{x}_i = -\nabla\Phi_i(x_i) - p_i$, where $\Phi_i(x_i)$ is a specific type of cost function, reverse-engineered and depending on the particular protocol. Namely, sources adapt their rate to minimize a given cost function (equivalently, maximize their utility) corrected by the aggregate price.

Hence, congestion control mechanisms interconnect source controllers and link dynamics into a feedback loop as illustrated in Figure 9, where $\Phi(x) := \sum_i \Phi_i(x_i)$. Importantly, each controller is fully distributed and requires

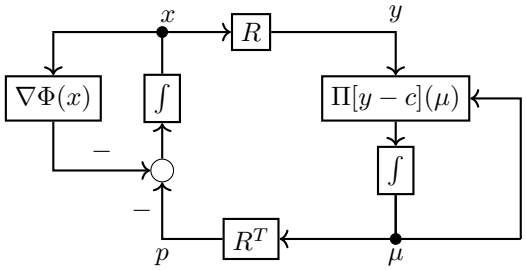


Figure 9: Simple network utility maximization

only locally available information. Written more compactly, the closed-loop dynamics are defined as

$$\dot{x} = -\nabla\Phi(x) - R^T\mu \qquad \dot{\mu} = \Pi_{\mathbb{R}^M_{\geq 0}}[Rx - c](\mu)\,.$$

This system defines a *projected saddle-point flow*, as in Example 3.4, whose trajectories converge to a solution of

$$\text{minimize} \quad \Phi(x) \qquad \text{subject to} \quad Rx \leq c\,, \tag{29}$$

whenever Φ is strictly convex.

From a control and optimization perspective, source controllers and link dynamics form a closed feedback loop that implicitly tracks the solution of the underlying utility maximization problem (Wang & Elia, 2011).

From the viewpoint of closed-loop optimization advocated in this article, one can argue that the links implement the dual dynamics that have to be complemented by controllers that form the primal dynamics. Together they optimize (29). ■

Example 4.2 (Optimal Frequency Control). Recent years have seen renewed interest in the control of power systems because of the new challenges associated with an increasingly dominating number of power electronic devices connected to the electricity grid and the growth of highly intermittent infeed from new renewable energy sources.

Frequency control in AC grids is one control task that has been revisited in this context. In this example, we illustrate how recent work has framed this challenge as a closed-loop optimization problem in order to improve upon classical frequency control schemes.

The reader is referred to Molzahn et al. (2017) for a recent survey on this topic and to Chen et al. (2020); Simpson-Porco (2021); Zhao et al. (2016) and references therein for latest results. The ensuing model is based on Li et al. (2016).

In contrast to the optimal reserve dispatch problem outlined in Section 1.2, we assume a *lossless* AC power transmission network with a set $\mathcal{N}$ of buses and a set $\mathcal{M}$ of transmission lines. We consider the power system around an operating point and all quantities are to be interpreted as deviations from their nominal values.

The frequency deviation ω_j at every bus j is governed by the *swing equation*

$$\dot{\omega}_j = \tfrac{1}{M_j}\left(p_j^M - D_j\omega_j - p_j^{\mathrm{L}} - \sum\nolimits_{k:j\to k} p_{jk}\right) \tag{30}$$

where M_j is the inertia of the generator and D_j is the damping constant at bus j. The mechanical power output of the generator and the power consumed at bus j are denoted by p_j^{M} and p_j^{L}, respectively. The power flowing out of bus j towards an adjacent bus k is written as p_{jk}. The notation $j \to k$ indicates a transmission line between buses j and k, i.e., $(j,k) \in \mathcal{M}$.

For every $j \to k$, *line flow dynamics* are linearized and modeled as

$$\dot{p}_{jk} = \mathrm{b}_{jk}(\omega_j - \omega_k) \tag{31}$$

where b_{jk} is the line susceptance. Note that transmission lines are undirected, and, throughout, the constraint $p_{jk} = -p_{kj}$ holds for every line implicitly.

At steady state and nominal frequency $\omega_j = 0$, the swing equation (30) expresses the power balance at each node, and (31) yields the so-called *DC-flow* approximation of the AC power flow equations (5b–c) (assuming nominal voltages, lossless lines, and small angle differences).

The mechanical power output p_j^{M} is described by a simplified *governor-turbine control* model of the form

$$\dot{p}_j^M = -\tfrac{1}{T_j}\left(p_j^{\mathrm{M}} - p_j^{\mathrm{C}} + \tfrac{1}{R_j}\omega_j\right) \tag{32}$$

where p_j^{C} is a power adjustment signal and T_j and R_j are constants. In particular, R_j is understood as a generator's participation factor in primary frequency control.

The power adjustment itself is evaluated using the so-called *area control error* (ACE) which combines frequency and line flow deviations. More specifically, we have

$$\dot{p}_j^C = -K_j\left(B_j\omega_j + \sum\nolimits_{k:j\to k} p_{jk}\right). \tag{33}$$

with B_j and K_j being constant control parameters.

The role of this type of *automatic generation control* (or *secondary frequency control*) is to react to changes in p_j^{L}, driving frequency deviation to zero, and to make sure that deviations in p_j^{L} are compensated by the local generator through adjustments of p_j^{M}. Consequently, at steady state, the deviations P_{ij} of the line flows from their nominal values are zero.[2]

Although not immediately obvious, the system (30–33) realizes a (partial) saddle-point flow, as introduced in Section 3.3. To see this, we consider the optimization problem

$$\begin{aligned} \text{minimize} \quad & \sum\nolimits_{j\in\mathcal{N}}\left(\tfrac{\beta_j}{2}\left(p_j^{\mathrm{M}}\right)^2 + \tfrac{D_j}{2}\omega_j^2\right) \\ \text{subject to} \quad & 0 = p_j^{\mathrm{M}} - p_j^{\mathrm{L}} \\ \forall j\in\mathcal{N} \quad & 0 = p_j^{\mathrm{M}} - D_j\omega_j - p_j^{\mathrm{L}} - \sum\nolimits_{k:j\to k} p_{jk}\,. \end{aligned} \tag{34}$$

[2] In this example, the area control error is evaluated and regulated on a per-bus basis. In more general settings, multiple buses belong to the same area, and therefore secondary frequency control regulates the aggregate power balance in each area and the power flows between areas.

where β_j is a positive parameter to be determined. The solution of this problem guarantees power balance and exact frequency regulation. We can write out the corresponding (unprojected) primal-dual saddle-point flow from Example 3.3 as

$$\begin{aligned}
\text{primal}\quad &\begin{cases} \dot{\omega}_j &= -\epsilon_{\omega_j}\left(D_j\omega_j - D_j\lambda_j\right) \\ \dot{p}_j^M &= -\epsilon_{P_i^M}\left(\beta_j p_j^{\mathrm{M}} - \mu_j - \lambda_j\right) \\ \dot{p}_{jk} &= -\epsilon_{p_{jk}}(\lambda_j - \lambda_k) \end{cases} \\
\text{dual}\quad &\begin{cases} \dot{\mu}_j &= \epsilon_{\mu_j}(p_j^{\mathrm{M}} - p_j^{\mathrm{L}}) \\ \dot{\lambda}_j &= \epsilon_{\lambda_j}\left(p_j^{\mathrm{M}} - D_j\omega_j - p_j^{\mathrm{L}} - \textstyle\sum_{k:j\to k} p_{jk}\right) \end{cases}
\end{aligned} \tag{35}$$

for all $j,k \in \mathcal{N}$ and $j \to k$, and where we use separate positive gains $\epsilon_{(\cdot)}$ for every variable.

By taking the limit $\epsilon_{\omega_j} \to \infty$, we can replace the corresponding ODE with the algebraic expression $0 = D_j(\omega_j - \lambda_j)$ and thus substitute $\omega_j = \lambda_j$. This results in a so-called *partial* saddle-point flow. In this special case, the ω_j can hence be interpreted as a primal or dual variable.

We further apply a transformation to μ_j by defining $p_j^{\mathrm{C}} := K_j\left(M_j\omega_j - \frac{1}{\epsilon_{\mu_j}}\mu_j\right)$. Consequently, we have

$$\dot{p}_j^C = K_j\left(M_j\dot{\omega}_j - \tfrac{1}{\epsilon_{\mu_j}}\dot{\mu}_j\right) = K_j\left(M_j\dot{\omega}_j - (p_j^{\mathrm{M}} - p_j^{\mathrm{L}})\right).$$

Combining these insights, we can rewrite the partial saddle-point flow derived from (35) as

$$\begin{aligned}
\text{primal}\quad &\begin{cases} \dot{p}_j^M &= -\epsilon_{P_i^M}\left(\beta_j p_j^{\mathrm{M}} - \mu_j - \omega_j\right) \\ \dot{p}_{ij} &= -\epsilon_{P_{ij}}\left(\omega_k - \omega_j\right) \end{cases} \\
\text{primal or dual}\quad &\left\{ \dot{\omega}_j = \epsilon_{\lambda_j}\left(p_j^{\mathrm{M}} - D_j\omega_j - p_j^{\mathrm{L}} - \sum_{k:j\to k} p_{jk}\right) \right. \\
\text{dual}\quad &\left\{ \dot{p}_j^C = -K_j\left(D_j\omega_j + \sum_{k:j\to k} p_{jk}\right). \right.
\end{aligned} \tag{36}$$

By carefully choosing the remaining gains $\epsilon_{(\cdot)}$ and β_j (e.g., $\epsilon_{P_{ij}} := \mathrm{b}_{ij}$ or $\omega_{\lambda_j} = \frac{1}{M_j}$) as well as, under $B_j = D_j$, (36) is equal to (30–33). The choice of $B_j = D_j$ is fragile, since it means that the controller (33) requires an accurate estimate of the damping D_j at every bus. However, numerical experiments in Li et al. (2016) show that (30–33) is stable even if $B_j = D_j$ does not hold exactly. A different choice of B_j will be used in Example 4.4 where we revisit the same frequency control problem and apply a timescale separation approach for the design of feedback-based optimization schemes.

Starting from this observation that automatic generation control can be interpreted from a closed-loop optimization perspective, various works have proposed optimal frequency control schemes that allow for more general objective functions than in (34), relax the constraint $p_j^{\mathrm{M}} = p_j^{\mathrm{L}}$, or incorporate additional constraints on generation and transmission capacity. ■

Examples 4.1 and 4.2 are instances of feedback-based optimization schemes that implement optimization dynamics as the closed-loop behavior. Particularly, both examples implement the (projected; partial) saddle-point flows discussed in Section 3.3. What sets these two examples apart from Example 1.1, is the fact that the plant dynamics are themselves part of the optimization algorithm and do not need to be assumed fast-decaying. While this construction is elegant, its theoretical convergence guarantee is directly tied to the plant model, e.g., the link congestion dynamics (28), the turbine-governor (32), the line dynamics (31). More realistic and complex models will not result in equally clean formulations as convex-concave saddle-point flows.

For this reason, rather than relying on the special plant structure and specific models, we present in the following section alternative and more general approaches to certify closed-loop stability of optimization algorithms interconnected with general (nonlinear) plants. Example 4.4 will revisit the frequency control problem Example 4.2 from this alternative perspective.

4.1. Review of Existing Works

The general topic of feedback-based optimization, as outlined so far, has recently generated substantial results which we review in the following. Most of these works have been developed with power systems applications in mind. However, we limit ourselves to a discussion of the control-theoretic aspects and do not dwell on specific applications (e.g., optimal frequency control) and topics treated elsewhere in this article.

The common denominator of the papers discussed in the following is the fact that they treat special cases of optimization problem (4), i.e., the minimization of a cost functions subject to the steady-state input-output behavior of a physical plant, input constraints, and a set of engineering constraints (usually on the plant outputs).

Table 1 attempts to compare these different problem setups, but does not make any statement about solution methods or technical contributions. While most papers in Table 1 propose new control schemes, others investigate particular properties (e.g., closed-loop stability or robustness) of existing methods. Also, for papers that study different problem setups, the most general and challenging properties are reported in Table 1.

The properties reported in Table 1 primarily illustrate the various dimensions of problem complexity in (4): A convex problem implies that the system needs to be steered to a (unique) global minimizer, whereas for non-convex problems convergence is generally only to one of multiple local minima. Input constraints $u \in \mathcal{U}$ are hard physical limits enforced by saturation (see Section 4.3.1) and require projection operations as part of the system dynamics (yielding non-smooth dynamics in continuous-time setups). Similarly, unilateral (i.e., inequality) constraints are often more demanding than equality constraints to enforce, because their nature leads to non-smoothness (es-

pecially in continuous-time models). Finally, if the controlled plant is not algebraic, closed-loop stability is not inherently guaranteed (see Section 4.2).

The control designs for solving convex problems are, to a large degree, based on saddle-point flow algorithms (Section 3.3). For instance, Chang et al. (2019); Tang et al. (2018a) implement a double projection saddle-point flow (see Example 3.4), Colombino et al. (2019a) is based on the proximal augmented Lagrangian method (Dhingra et al., 2019), and Bernstein et al. (2019) implements a saddle-point flow with regularization in the dual variables. These saddle-point flows with dual augmentation have also been applied in Dall'Anese & Simonetto (2018); Tang et al. (2018b) to non-convex problems, although the convergence conditions are involved.

More suited for non-convex constrained problems are projected gradient methods because of their general global convergence properties (see Theorem 1). Variations of these schemes have been studied in Gan & Low (2016); Häberle et al. (2021); Hauswirth et al. (2016, 2020b,c,d,e). In the same class of descent methods, a quasi-Newton method has been proposed Tang et al. (2017), although constraints cannot be easily enforced by projection and, instead, a penalty/barrier function is required (see Example 3.2).

The control architectures in Lawrence et al. (2018, 2020); Nelson & Mallada (2018); Picallo et al. (2020); Simpson-Porco (2020); Zhang et al. (2018) are more difficult to categorize since they also include the design of estimators or stabilizing controllers.

The issue of closed-loop stability (i.e., when interconnected with a dynamic plant) has so far been studied only for continuous-time models. Works on discrete-time feedback-based algorithms have so far always assumed a algebraic plant that is fully characterized by its steady-state input-output map.

Among the papers that have considered dynamic plants, most have assumed linear time-invariant plants. Notable exceptions are Hauswirth et al. (2020b); Simpson-Porco (2020) (although the closed-loop optimization example in the latter features only an LTI plant).

Finally, time-varying problems are studied in Bernstein et al. (2019); Colombino et al. (2019a); Tang et al. (2018a,b, 2017) which also provide bounds on the tracking performance under various assumptions (see also Section 3.5). Furthermore, Bernstein et al. (2019) also models the effect of measurement noise.

4.2. Closed-Loop Stability

Example 1.2 in the introduction has illustrated that a simple gradient-based controller interconnected with a dynamical system is not necessarily stable, unless the control gain ϵ is small enough (Figure 2). In other words, *sufficient timescale separation* between the fast plant behavior and the slow optimization dynamics is generally required. As discussed in Section 2.1, extremum seeking also relies on this type of argument, with the difference that ES dynamics evolve on three timescales.

For control design, it is convenient to reduce fast transient dynamics to their algebraic steady-state map. We have explicitly applied this design step in Examples 1.1, 2.1, and 4.1 and will do so implicitly in the rest of the article. In this section, however, we will investigate the problems of timescale separation and closed-loop stability in detail and survey works that have tackled this issue.

We identify and present two main research streams: First, we discuss stability results inspired by singular perturbation analysis which formalize and quantify stability in terms of timescale separation. This approach is very general and applicable to nonlinear (but asymptotically stable) plant dynamics and non-convex optimization dynamics, but potentially very conservative. A second line of research taps into robust control and provides computational stability certificates in the form of linear matrix inequalities. While these stability guarantees are less conservative they apply only to linear time-invariant plants and convex optimization dynamics.

Remark 4.1. Exploiting *passivity* is sometimes a third possibility to certify closed-loop stability (Khalil, 2002, Chap. 6). This approach requires that the control loop can be identified as a feedback interconnection of passive systems, which is not always possible. Saddle-point flows (discussed in Section 3.3) are one class of systems that are amenable to passivity arguments to study stability and robustness (Simpson-Porco, 2016; van der Schaft, 2011). These ideas have been applied for the congestion avoidance in communication networks (Example 4.1) in Wen & Arcak (2004) and (Low, 2017, Chap. 4). Similarly, passivity has been exploited in power systems applications (like Example 4.2) in Stegink et al. (2017); Trip et al. (2019). ■

4.2.1. Singular Perturbation Analysis

For the purpose of providing explicit bounds on ϵ, Hauswirth et al. (2020b); Menta et al. (2018) pursue a singular perturbation approach and arrive at easy-to-compute sufficient conditions for closed-loop stability. In particular, Hauswirth et al. (2020b) considers general nonlinear (but stable) plant dynamics and investigates a variety of optimization dynamics for convex and non-convex problems including (projected) gradient, saddle-point, and momentum methods. Using similar techniques, Simpson-Porco (2020) provides stability guarantees for general low-gain integral controllers that satisfy an infinitesimal contraction property, but with a discussion of feedback-based optimization limited to LTI plant dynamics. The following example illustrates the main idea behind these stability conditions.

Example 4.3. We wish to characterize the stability of the feedback loop introduced in Example 1.1. In particular, we want formulate conditions on the gain ϵ in Figure 1 that guarantee closed-loop stability. For this purpose we pass to the singular perturbation decomposition into reduced

	convex problem	input constraints	output constraints	continuous- or discrete-time	plant model
Gan & Low (2016)	no	yes	none/soft	DT	nonlin. algebraic
Hauswirth et al. (2016)	no	yes	none/soft	CT (non-smooth)	nonlin. algebraic
Tang et al. (2017)	no	yes	none/soft	DT	nonlin. algebraic
Hauswirth et al. (2017)	no	yes	none/soft	DT	nonlin. algebraic
Mazzi et al. (2018)	no	yes	unilateral	DT	nonlin. algebraic
Dall'Anese & Simonetto (2018)	no	yes	unilateral	DT	nonlin. algebraic
Hauswirth et al. (2018)	no	yes	none/soft	CT (non-smooth)	nonlin. algebraic
Tang et al. (2018b)	no	yes	unilateral	DT	nonlin. algebraic
Tang et al. (2018a)	no	yes	unilateral	DT/CT (non-smooth)	nonlin. algebraic
Nelson & Mallada (2018)	yes	no	none/soft	CT (smooth)	LTI
Lawrence et al. (2018)	yes	no	lin. equality	CT (smooth)	LTI
Menta et al. (2018)	yes	no	lin. equality	CT (smooth)	LTI
Zhang et al. (2018)	yes	no	unilateral	CT (non-smooth)	LTI
Colombino et al. (2019b)	no	yes	none/soft	DT	nonlin. algebraic
Bernstein et al. (2019)	yes	yes	unilateral	DT	nonlin. algebraic
Chang et al. (2019)	yes	yes	unilateral	CT (non-smooth)	lin. algebraic
Colombino et al. (2019a)	yes	no	linear eq.	CT (smooth)	LTI
Lawrence et al. (2020)	yes	no	linear eq.	CT (smooth)	LTI
Picallo et al. (2020)	no	yes	none/soft	DT	nonlin. algebraic
Hauswirth et al. (2020e)	yes	yes	none/soft	CT (non-smooth)	nonlin. dynamic
Hauswirth et al. (2020c)	no	yes	unilateral	CT (non-smooth)	nonlin. dynamic
Hauswirth et al. (2020b)	no	yes	unilateral	CT (non-smooth)	nonlin. dynamic
Simpson-Porco (2020)	yes	no	lin. equality	CT (smooth)	nonlin. dynamic
Häberle et al. (2021)	no	yes	unilateral	DT	nonlin. algebraic

Table 1: Comparison of problem setups; *Convexity* refers to whether the problem one wishes to solve in closed loop is convex and thus exhibits a unique global optimum. *Input constraints* refer to the presence of constraints $u \in \mathcal{U}$ on the control input. *Output constraints* refer to the presence of constraints $(u, y) \in \mathcal{X}$ in (4). *None/soft* means that output constraints are either not considered or approximately enforced by penalty or barrier functions. Continuous-time models can be either *smooth* or *non-smooth* (e.g. by including projections like discussed in Section 3.2). Finally, the physical plant may be assumed to be algebraic (in which case the closed-loop stability is not an issue), an LTI system (which results in a linear input-output map h), or governed by nonlinear but stable dynamics.

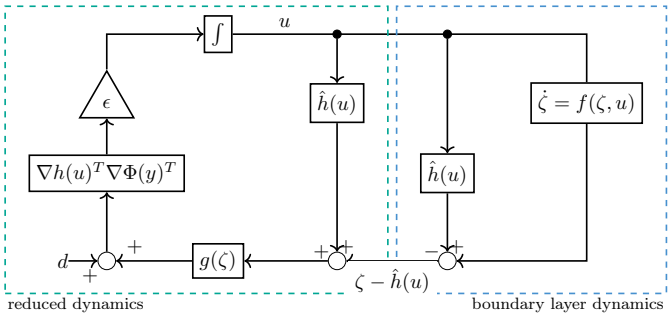


Figure 10: Feedback-based gradient flow from Figure 1 rearranged and decomposed into reduced and boundary-layer error dynamics

and boundary-layer error dynamics illustrated in Figure 10 (Khalil, 2002; Kokotovic et al., 1999). In particular, $\hat{h}$ is defined such that $f(\hat{h}(u), u) = 0$ for all u.

The resulting reduced dynamics correspond exactly to the simplified model what we have already used in the design of the optimizing controller, where the plant is replaces by its algebraic steady-state map. The boundary-layer error dynamics $z := \zeta - \hat{h}(u)$ evolve as $\dot{z} = f(\zeta, u)$ for any fixed u. If these error dynamics are exponentially stable (and other technical assumptions are satisfied), standard converse results guarantee the existence of a Lyapunov function W and parameters $\gamma, \omega > 0$ such that, for any fixed u, it holds that

$$\begin{aligned} \dot{W}(\zeta - \hat{h}(u)) &\leq -\gamma \|\zeta - \hat{h}(u)\|^2 \\ \|\nabla_u W(\zeta - \hat{h}(u))\| &\leq \omega \|\zeta - \hat{h}(u)\| \,. \end{aligned} \tag{37}$$

A class of Lyapunov function candidates to certify stability of the closed system in Figure 10 is given by

$$V_\delta(u, \zeta) = \delta \tilde{\Phi}(u) + (1 - \delta) W(\zeta - \hat{h}(u))$$

where $\delta \in (0, 1)$ is a convex combination parameter.

Let L be the Lipschitz constant of $\nabla \hat{\Phi}$. Hauswirth et al. (2020b) have shown that for all

$$\epsilon < \epsilon^\star := \frac{\gamma}{\omega L} \tag{38}$$

the parameter δ can be chosen such that V_δ is non-increasing and thus a LaSalle invariance argument guarantees (asymptotic) stability. ■

Remark 4.2. Recall from Example 3.2 that constraints can be incorporated into an optimization problem through penalty or barrier functions. The bound (38) on ϵ in Example 4.3 indicates that there is a natural limitation to this constraint enforcement mechanism. Notably, penalty functions cannot be made arbitrarily steep, thus increasing L, without at the same time reducing ϵ by the same fraction. For the same reason log-barrier functions which do not have a Lipschitz gradient have to be applied with great care, as closed-loop stability cannot be guaranteed with these singular perturbation bounds on ϵ. ■

The timescale separation argument inspired by singular perturbation analysis, as presented in Example 4.3, works under very general conditions. In particular, for the case of a gradient flow convexity of Φ is not generally required. The type of stability proof can also be established for various optimization algorithms interconnected with exponentially stable plants. Examples include but are not limited to Newton flows, projected gradient flows, and saddle-point flows encountered in the previous section. These conditions can be very conservative, but they are qualitatively tight. Namely, non-examples in Hauswirth et al. (2020b) show how subgradient flows and continuous-time accelerated gradient flows interconnected with dynamical plants are not generally stable. These setups are not amenable to the same type of timescale separation argument as above because important assumptions such as uniform asymptotic stability of the reduced dynamics are not satisfied (Poveda & Li, 2019).

Example 4.4. Reconsider the frequency control problem from Example 4.2. We have seen that the simplified model of automatic generation control in AC power grids given by (30–33) can be reformulated as a partial saddle-point flow. However, this interpretation works only under the proposed modeling assumptions on line flows, swing dynamics, and the turbine-governor.

Alternatively, automatic generation control can also be formulated as the interconnection of a plant with *fast-decaying* dynamics interconnected with a simple gradient controller as in Example 1.1. For this purpose, we identify the input-output steady-state map $h(u)$ of the system dynamics (30–32) with $u = P^{\mathrm{C}}$ being the input and $d = P^{\mathrm{L}}$ a disturbance.

Note from (31) that at steady state all frequencies are the same, i.e., $\omega_i = \omega_j$ for all $i, j \in \mathcal{N}$. Furthermore, we have from (32) that $P_j^{\mathrm{M}} = P_j^{\mathrm{C}} - \frac{1}{R_j}\omega_j$. Consequently, at steady state, it holds for all $j \in \mathcal{N}$

$$0 = p_j^{\mathrm{C}} - (D_j + \tfrac{1}{R_j})\omega - p_j^{\mathrm{L}} - \sum\nolimits_{k:j\to k} p_{jk}\,, \qquad (39)$$

where ω is the unique system frequency.

Next, we define the ACE gain $B_j := D_j + \frac{1}{R_j}$ (as opposed to $B_j = D_j$ in Example 4.2), and we choose the ACE to be the output of the system. This allows us to define the (almost trivial) input-output map h for every $j \in \mathcal{N}$ as

$$y_j = B_j\omega + \sum_{k:j\to k} p_{jk} = p_j^{\mathrm{C}} - p_j^{\mathrm{L}} = u_j - d_j =: h_j(u)\,.$$

Hence, we can almost directly refer back to Example 1.1 to minimize $\Phi(y) := \frac{1}{2}\|y\|^2$ using the gradient controller

$$\begin{aligned}\dot{u}_j = -\epsilon_i \left[\nabla h(u)^T \nabla\Phi(y)\right]_j &= -\epsilon(p_j^{\mathrm{C}} - p_j^{\mathrm{L}}) \\ &= -\epsilon\left(B_j\omega + \sum\nolimits_{k:j\to k} p_{jk}\right)\end{aligned}$$

and thus recover the ACE-controller (33) with $\epsilon_i = K_i$. The only difference to Example 1.1 is the fact that ϵ_i is different for every component rather than a global control. This detail is well-motivated since the different ϵ can be interpreted simply as a constant metric applied to the gradient flow; see Example 3.1.

In contrast to Example 4.2, the timescale separation approach to frequency control does not make any assumption on the underlying dynamical systems. As long as these dynamic transients decay quickly and (39) is satisfied at steady state, the frequency deviation (and other quantities) can be controlled and driven to their nominal values or an economically efficient operating condition.

One of the limitations compared to Example 4.2 is the fact that, from a theoretical viewpoint, closed-loop stability is guaranteed only for a small enough gains $\epsilon_i = K_i$. The partial saddle-point reformulation found in Example 4.2 is globally convergent for any positive control gains, albeit only under the given modeling assumptions. ■

4.2.2. LMI Stability Certificates

Alternatively, closed-loop stability can be certified by applying tools from robust control, namely linear matrix inequalities. Recent results on integral quadratic constraints and their use for the analysis of optimization algorithms (Fazlyab et al., 2017; Lessard et al., 2016) have proven very useful for this purpose and have been applied in Colombino et al. (2019a); Nelson & Mallada (2018). In particular, Nelson & Mallada (2018) studies the joint design of stabilizing control, estimator and optimization dynamics. Similarly, Lawrence et al. (2018, 2020) considers an output regulation framework and reduce the control design to a stabilization problem.

Limiting the applicability of these techniques from robust control is the fact that plant dynamics are generally required to be LTI, and objective functions need to be (strongly) convex. Moreover, these LMI-based conditions are in the form of computational stability certificates and do not directly translate into control design procedures or tuning recommendations.

The following example presents a LMI-IQC based stability test for the feedback-based gradient flow.

Example 4.5. Consider the same setup as in Example 1.1, and, in addition, assume that Φ is m-strongly convex and has a L-Lipschitz gradient. Further, assume that

$$\dot{\zeta} = f(\zeta, u) = Ax + Bu \qquad y = g(\zeta) + d = C\zeta + d \quad (40)$$

is an LTI system. Assuming that A is invertible, the steady-state input-output map is given by $h(u) = Hu$ with $H = -CA^{-1}B$. Furthermore, (3) has a unique equilibrium $(\zeta^\star, u^\star)$ such that $\nabla\tilde{\Phi}(u^\star) = 0$ and $0 = A\zeta^\star + Bu^\star$.

The closed-loop system (3) can be written in a standard robust control setup as an interconnection of a nonlinearity with an LTI system. This is shown in Figure 11, where

$$\mathbf{z} := \begin{bmatrix}\zeta\\ u\end{bmatrix} \qquad \mathbf{A} := \begin{bmatrix}A & B\\ 0 & 0\end{bmatrix} \qquad \mathbf{B} := \begin{bmatrix}0 & 0\\ 0 & -H^T\end{bmatrix}$$

$$\mathbf{y} := \begin{bmatrix}y\\ u\end{bmatrix} \qquad \mathbf{C} := \begin{bmatrix}0 & \mathbb{I}\\ C & 0\end{bmatrix} \qquad \mathbf{D} := \begin{bmatrix}\mathbb{I}\\ 0\end{bmatrix}$$

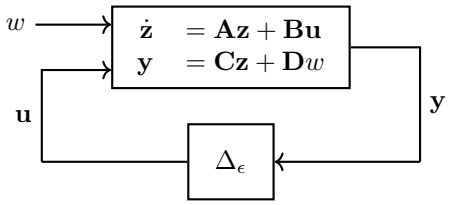


Figure 11: Feedback-based gradient flow from Figure 1 rewritten as a feedback interconnection of an LTI system with a nonlinearity Δ

and

$$\mathbf{u} = \Delta_\epsilon(\mathbf{y}) = \begin{bmatrix} 0 \\ \epsilon \nabla \Phi(y) \end{bmatrix}. \tag{41}$$

Crucially, $\Delta_\epsilon(\cdot)$ satisfies the IQC defined by

$$\begin{bmatrix} \mathbf{y} - \mathbf{y}^\star \\ \mathbf{u} - \mathbf{u}^\star \end{bmatrix}^T \underbrace{\begin{bmatrix} -2\epsilon^2 mL\mathbb{I} & \epsilon(L+m)\mathbb{I} \\ \epsilon(L+m)\mathbb{I} & -2\mathbb{I} \end{bmatrix}}_{\Xi_\epsilon} \begin{bmatrix} \mathbf{y} - \mathbf{y}^\star \\ \mathbf{u} - \mathbf{u}^\star \end{bmatrix} \geq 0 \tag{42}$$

for all (y, u) such that $y = \Delta(y)$ (Lessard et al., 2016, Lemma 6). Equivalently, we have

$$\begin{bmatrix} \mathbf{z} - \mathbf{z}^\star \\ \mathbf{u} - \mathbf{u}^\star \end{bmatrix}^T \mathbf{C}^T \Xi_\epsilon \mathbf{C} \begin{bmatrix} \mathbf{z} - \mathbf{z}^\star \\ \mathbf{u} - \mathbf{u}^\star \end{bmatrix} \geq 0 \tag{43}$$

for all $(\mathbf{z}, \mathbf{u})$ such that $\mathbf{u} = \Delta(\mathbf{C}\mathbf{z})$.

Next, consider a Lyapunov function of the form $V(\mathbf{z}) = (\mathbf{z} - \mathbf{z}^\star)^T \mathbf{P} (\mathbf{z} - \mathbf{z}^\star)$ where $\mathbf{P} \succeq 0$ remains to be determined. The time derivative of V along system trajectories can be written as a quadratic form

$$\dot{V}(\mathbf{z}) = \begin{bmatrix} \mathbf{z} - \mathbf{z}^\star \\ \mathbf{u} - \mathbf{u}^\star \end{bmatrix}^T \begin{bmatrix} \mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} & \mathbf{P}\mathbf{B} \\ \mathbf{B}^T\mathbf{P} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{z} - \mathbf{z}^\star \\ \mathbf{u} - \mathbf{u}^\star \end{bmatrix}. \tag{44}$$

Combining (43) and (44), it follows that $\dot{V}(\mathbf{z}) < 0$ for $\mathbf{z} \neq \mathbf{z}^\star$ (and thus the closed-loop is stable) if the LMI

$$\begin{bmatrix} \mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} & \mathbf{P}\mathbf{B} \\ \mathbf{B}^T\mathbf{P} & 0 \end{bmatrix} \prec -\mathbf{C}^T \Xi_\epsilon \mathbf{C} \tag{45}$$

holds for some $\mathbf{P} \succeq 0$. ■

4.3. Constraint Enforcement in Closed Loop

The capacity and various possibilities to robustly enforce complicated constraints despite model uncertainty is one of the distinguishing features of feedback-based optimization.

In Section 3 we have seen different mechanisms used in continuous-time optimization algorithms to deal with constraints. Section 3.4 summarizes these options and highlighted how they can be combined. But so far, we have encountered only saddle-point flows as means to enforce constraints in closed loop in Examples 4.1 and 4.2.

In this subsection, we will explore other possibilities to enforce constraints based on the algorithms discussed in Section 3. In particular, we will discuss on one hand the option to exploit saturation to enforce constraints on plant inputs without major computational effort. On the other hand, we review strategies to handle general engineering constraints on outputs (and inputs). For this latter problem, we will refer back to the projected gradient and saddle-point flows in Section 3 which can both be adapted for this purpose.

At the end of this subsection, we will hence be able to present fully-fledged feedback-based optimization designs to tackle the optimal reserve dispatch problem formulated in the introduction in Section 1.2.

4.3.1. Input Saturation via Projection and Anti-Windup

Physical plants are generally subject to limited actuator capabilities, simply due to the fact that any realistic system is bound by the laws of physics and can handle only signals of finite power. Such actuation limits can often be modeled as input saturation of control signal. At a physical level, such saturation is, for example, the result of mechanical constraints, or it can be observed in electric circuits involving diodes and other semi-conducting devices. From a systems perspective, low-level controllers that protect devices and subsystems from operating outside of a safe zone of operating conditions can also be modeled as saturation.

In an optimization context, these actuation limits translate into constraints on the control inputs. In (4) these constraints are collected by the expression $u \in \mathcal{U}$.

The nature of these constraints implies that they cannot be violated at any point in time. Considering the constraint enforcement possibilities from Section 3.4, it makes sense to model saturation as a projection onto the set of feasible inputs with the key property that the projection is naturally "evaluated" and applied by the physical system.

This property creates the possibility to outsource the handling of these constraints from the controller to the plant and thus reduce computational requirements and the need for exact modeling of these constraints, which can be time-varying and/or unknown.

For discrete-time feedback-based optimization schemes such as those in Bernstein et al. (2019); Dall'Anese & Simonetto (2018); Gan & Low (2016), this model-free constraint handling by saturation is relatively straightforward, under the assumption that the saturated control signal is measured and available within one sampling interval. In this case, it is possible to simply add the control increment to the measured saturated control signal of the preceding sampling interval.

For continuous-time methods, as in Chang et al. (2019); Hauswirth et al. (2016), exploiting input saturation is trickier because a continuous-time integrator in cascade with a saturation element will generally lead to *integrator windup*. For this reason Hauswirth et al. (2020c,d,e) study the use of anti-windup compensators for feedback-based optimization and rigorously show how these control designs can be used to smoothly and robustly approxi-

mate continuous-time projected gradient flows like (13). The following example illustrates this idea.

Example 4.6. Consider the same setup as in Example 1.1. Namely, we want to drive a plant (with fast decaying dynamics and steady-state map $y = h(u) + d$) to an optimal steady state minimizing the cost function $\tilde{\Phi}(u)$.

In addition, we assume that the plant is subject to input saturation that acts like a projection onto a non-empty set $\mathcal{U}$ of admissible inputs. For simplicity, assume that $\mathcal{U}$ is convex. In order to mitigate the effects of integrator windup, a simple anti-windup scheme with a tuneable gain $\frac{1}{K}$ is in place as illustrated in Figure 12.

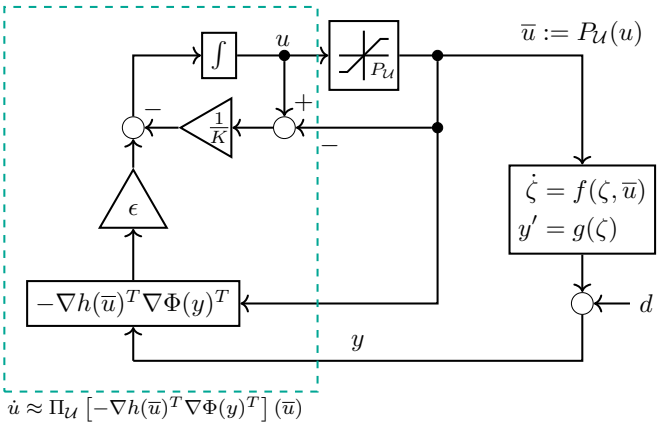


Figure 12: Anti-windup system (46) which approximates a projected gradient flow (49) as $K \to 0^+$

A state-space representation of this system is given, analogously to (3), by

$$\text{plant} \begin{cases} \overline{u} &= P_{\mathcal{U}}(u) \\ \dot{\zeta} &= f(\zeta, \overline{u}) \\ y &= g(\zeta) + d \end{cases} \qquad \text{controller} \left\{ \dot{u} = -\epsilon \nabla h(u)^T \nabla \Phi(y)^T - \tfrac{1}{K}(u - \overline{u}) \right. . \tag{46}$$

Assuming, as before, that the plant dynamics can be approximated by the algebraic map $y = h(u) + d$ and with $\tilde{\Phi}(u) = \Phi(h(u) + d)$, the system (46) reduces to

$$\dot{u} = -\epsilon \nabla \tilde{\Phi}(P_{\mathcal{U}}(u))^T - \tfrac{1}{K}(u - P_{\mathcal{U}}(u)) . \tag{47}$$

The key aspect in (47) is the fact that $\nabla \tilde{\Phi}$ is evaluated at $P_{\mathcal{U}}(u)$ rather than at u. In Hauswirth et al. (2020d) it was shown that trajectories $u(t)$ of (47) converge in the sense that $P_{\mathcal{U}}(u(t))$ converges to the set of KKT points of the optimization problem

$$\text{minimize} \quad \tilde{\Phi}(u) \qquad \text{subject to} \quad u \in \mathcal{U} . \tag{48}$$

In particular, if (48) is convex, then convergence of $P_{\mathcal{U}}(u(t))$ is to the global minimizer.

Importantly, this means that upon convergence, the plant in Figure 12 is at an optimal steady state even though the (unsaturated) control u is not optimal (only the saturated control $P_{\mathcal{U}}(u)$ is optimal).

Furthermore, Hauswirth et al. (2020c,e) have shown that (47) approximates the projected gradient flow

$$\dot{u} = \Pi_{\mathcal{U}} \left[-\nabla h(\overline{u})^T \nabla \Phi(y)^T\right](\overline{u}) \tag{49}$$

as $K \to 0^+$. This insight extends to other anti-windup schemes which can be shown to approximate more general projected dynamical systems. ■

4.3.2. Engineering Constraints via Dualization and Approximate Projections

We now turn to mechanisms that allow us to enforce more general constraints, and in particular those that apply to plant outputs. In our general problem (4) these constraints are denoted by $(u, y) \in \mathcal{X}$.

In the following, we focus on two approaches. First, we illustrate how augmented and projected saddle-point flows, as in Example 3.4, can be implemented as feedback controllers. This approach is particularly suited for solving convex problems in closed loop and distributed over a network (assuming the problem exhibits a suitable sparsity structure). For non-convex problems, these algorithms can still be applied, but theoretical global convergence guarantees are not generally available.

As a second possibility to enforce engineering constraints, we discuss a special discretization of projected gradient flows that can be implemented as a feedback controller and comes with strong global convergence guarantees, even for non-convex setups. However, it is less easily amenable to a distributed implementation and, instead, requires the solution of a simple quadratic program at every iteration.

Projected Saddle-Flows as Feedback Controllers. Consider the optimal steady-state problem (4). Assume that the disturbance d is fixed and that the engineering constraints $(u, y) \in \mathcal{X}$ can be expressed as $\mathcal{X} := \{(u, y) \,|\, g(y, u) \leq 0\}$, where $g : \mathbb{R}^n \to \mathbb{R}^m$ denotes a continuously differentiable constraint function. After eliminating y from (4), we are left with the reduced problem

$$\begin{aligned} \text{minimize} \quad & \hat{\Phi}(u) := \Phi(u, h(u, d)) && \text{(50a)} \\ \text{subject to} \quad & u \in \mathcal{U} && \text{(50b)} \\ & u \in \hat{\mathcal{X}} := \{u \,|\, \hat{g}(u) \leq 0\} && \text{(50c)} \end{aligned}$$

with $\hat{g}(u) := g(u, h(u))$.

This problem can be tackled with a projected saddle-point flow with a primal augmentation term $\phi(u) = \frac{\rho}{2} \| \max\{\hat{g}(u), 0\} \|^2$ as presented in Example 3.4. Namely, we have

$$\dot{u} = \Pi_{\mathcal{U}} \left[-\nabla \hat{\Phi}(u)^T - \nabla \hat{g}(u)^T (\mu + \rho \max\{\hat{g}(u), 0\})\right](u) \tag{51a}$$

$$\dot{\mu} = \Pi_{\mathbb{R}^m_{\geq 0}} \left[\hat{g}(u)\right](\mu) . \tag{51b}$$

Recall from Example 3.4 that, in the special case where (50) is convex, trajectories of (51) converge to a global minimizer. In particular thanks to the primal augmentation, as in (23). The primal augmentation term reduces oscillations, especially with non-strictly convex objective functions. If the problem is non-convex, a dual augmentation term, as in (24), can be added to guarantee global convergence at the expense of changing equilibrium points away from the true KKT points of (50).

We wish to realize (51) as the closed-loop dynamics of a feedback loop incorporating the physical plant. To achieve this, we replace any evaluation of $h(u,d)$ with the plant output y. This yields the controller

$$\dot{u} = \Pi_{\mathcal{U}}^{Q}\left[-\epsilon Q(u) H_d(u) G(u,y)\right](u) \tag{52a}$$

$$\dot{\mu} = \Pi_{\mathbb{R}^m_{\geq 0}}\left[g(u,y)\right](\mu) \tag{52b}$$

where $H_d(u) := \begin{bmatrix}\mathbb{I} & \nabla_u h(u,d)^T\end{bmatrix}$ and

$$G(u,y) := \begin{bmatrix}\nabla_u \Phi(u,y)^T - \nabla_u g(u,y)^T(\mu + \rho \max\{g(u,y),0\}) \\ \nabla_y \Phi(u,y)^T - \nabla_y g(u,y)^T(\mu + \rho \max\{g(u,y),0\})\end{bmatrix}.$$

We have also included in (52) a variable metric $Q(u)$ on the primal variables. This degree of freedom will be exploited in the forthcoming Examples 4.7 and 4.8.

The subscript for H_d indicates the dependence on the disturbance d whereas $G(u,y)$ can be evaluated independently of d (given the measurement of y). In practice, however, it is often not necessary to estimate d to compute $H_d(u)$. Instead, one can build an estimate $\hat{H}$ of $H_d(u)$ by using y and u, i.e., $\hat{H}(u,y) \approx H_d(u)$.

Furthermore, if the plant (1) is an asymptotically stable LTI system subject to constant disturbances

$$\dot{\xi} = A\xi + Bu + d_1 \qquad y = C\xi + Du + d_2\,, \tag{53}$$

then the steady-state sensitivity matrix $\nabla_u h(u,d) = -CA^{-1}B + D$ is constant and independent of d, which renders its estimation much easier.

Note, however, that those existing work in Table 1 that propose primal-dual algorithms have all assumed that plants are algebraic. Thus, closed-loop stability of primal-dual saddle-point flows interconnected with dynamic plants has not yet been studied (with the exception of Hauswirth et al. 2020b where preliminary results are developed for a simplified setup).

Despite a lack of theoretical convergence guarantees for non-convex problems and closed-loop stability certificates, primal-dual saddle-point algorithms (and their projected, proximal, and augmented variations) have enjoyed great popularity. In particular, saddle-point flows often lend themselves to a distributed implementation. Examples 4.1 and 4.2 are prime examples. In the context of (52), a distributed implementation is possible if $\nabla_u h$ has a fixed sparsity pattern, the cost function is separable, and the constraints $u \in \mathcal{U}$ and $g(u,y) \leq 0$ are appropriately localized. In this case, the computation of a given component u_i requires only the quantity $\mu_j + \rho \max\{g_j(u,y),0\}$ to be communicated from all neighboring nodes j.

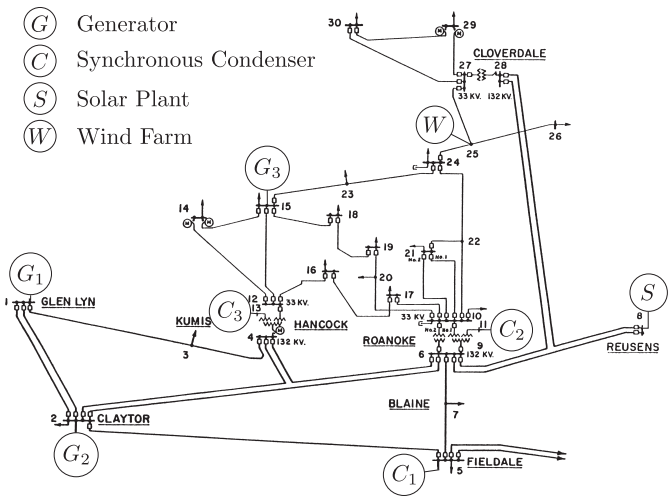


Figure 13: Grid diagram of adapted 30-bus power systems test case used in Examples 4.7 and 4.8.

The feedback controller (52) can be directly applied to the optimal reserve dispatch problem in Section 1.2 as the following numerical example illustrates.

Example 4.7. Consider the optimal reserve dispatch problem formulated in Section 1.2. Namely, we wish to steer a power system to an operating state that solves the ACOPF problem (5) and track its (time-varying) solution as consumption and generation availability varies. To achieve this, we implement a feedback controller according to (52) to implement a projected and primal augmented saddle-point flow as explained above.

As a numerical testbed, we use a modified version of the IEEE 30-bus power systems test case adopted from Hauswirth (2020); Hauswirth et al. (2017) and illustrated in Figure 13. In particular, in addition to three generators and three synchronous condensers, a wind and solar plant provide intermittent generation capacity. Their respective profiles of available generation capacity are illustrated in Figure 14. One of the conventional generators suffers an outage at 4:00 upon which its generation capacity is set to zero for the remainder of the simulation. The goal is hence to optimally use these renewable resources without violating engineering constraints such as voltage limits and thermal line ratings.

We implement (52) as a discretized controller using a simple explicit Euler discretization with a step size corresponding to 1 minute intervals between control actions. The physical system is simulated without dynamics, and an off-the-shelf AC power flow solver (Zimmerman et al., 2011) is used to compute a solution to the ACPF equations (5b–c) and thus evaluate the input-output map $h(u,d)$. We further compute $\nabla_u h(u,d)$ based on input and output measurements and the power flow model in (5) (rather than an explicit estimate of d). Finally, we choose $Q(u) := H_d(u)^T H_d(u)$ as a metric. This choice of "implicit" metric has proven to increase numerical stability

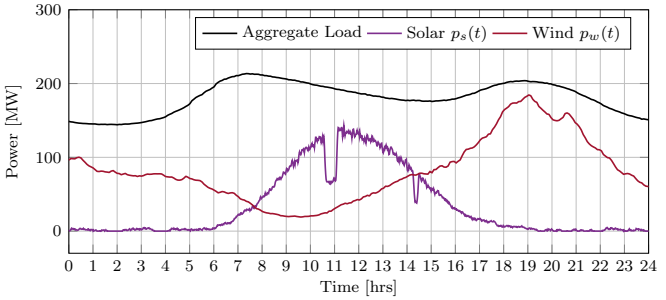


Figure 14: Load and available generation profiles for the modified 30-bus test case in Examples 4.7 and 4.8 over a 24h horizon.

in the face of an ill-conditioned input-output map. For a more detailed discussion and motivation see Chapter 14 in Hauswirth (2020).

The left panel in Figure 15 illustrates the (almost perfect) performance achieved by this feedback-based optimization approach in terms of cost compared to the a-posteriori sequential solution of the ACOPF problem (which is based on omniscient and perfect information).

More importantly, however, the left panel in Figure 16 shows that, over the entire simulation horizon, constraint violations are very minor and only temporary. The controller achieves this by jointly managing active and reactive power infeed to manage voltage magnitudes and line currents. In particular, both solar and wind generation have to be curtailed to prevent line overloads.

The controller (52) used in this example is similar to the designs proposed in Bernstein et al. (2019); Dall'Anese & Simonetto (2018); Tang et al. (2018a,b). However, the controller does not come with the same theoretical convergence and tracking guarantees because we have not incorporated a dual regularization which can be used to enforce convergence for non-convex optimization problems. Nevertheless, the controller performs well. ■

Linearized Output Projections. As a second possibility to solve (4) in closed loop (and thus tackle the reserve dispatch problem from Section 1.2), we highlight the method proposed in Häberle et al. (2021) and extended in Hauswirth (2020).

We have seen how projected (and augmented) saddle-point flows can be easily turned into feedback controllers. The key mechanism for enforcing the engineering constraints $(u, y) \in \mathcal{X}$ thereby lies in integrating the constraint violation $g(u, y)$ that is based on plant output measurements. In Example 4.7 we have noted the ease of deriving a discrete controller by simply applying an explicit Euler discretization.

Now, instead, we present a discrete controller that approximates the projected gradient flow

$$\dot{u} = \Pi_{\mathcal{U}\cap\hat{\mathcal{X}}}[-\nabla\hat{\Phi}(u)](u)\,. \tag{54}$$

The trajectories of (54) are guaranteed to converge to the KKT points of the reduced problem (50), even under non-convexity (but under technical assumptions similar to those in Theorem 1).

We have previously seen in Section 4.3.1 that input constraints $u \in \mathcal{U}$ are often enforced by input saturation (possibly requiring an anti-windup compensator to avoid integrator windup). The challenge in (54) lies in the fact that also the engineering constraints $\mathcal{X}$ need to be enforced "by projection" and in feedback using measurements of the plant output, rather than using a model-based evaluation of $h(u, d)$.

To address this problem, rather than pondering on the continuous-time dynamics (54), we directly consider a discrete-time controller of the form

$$u^+ = u + \alpha\,\Sigma_\alpha(u, y)\,, \tag{55}$$

where $\alpha > 0$ is a fixed step-size, $y = h(u, d)$ is the measured system output, and $\Sigma_\alpha(u, y)$ is defined as the solution of

$$\underset{w\in\mathbb{R}^p}{\text{minimize}}\ \|w + Q(u)H(u,d)^T\nabla\Phi(u,y)^T\|^2_{Q(u)} \tag{56a}$$

$$\text{subject to}\quad u + \alpha w \le \mathcal{U} \tag{56b}$$

$$g(u + \alpha w, y + \alpha\nabla_u h(u,d)w) \le 0\,, \tag{56c}$$

where, as before, $H_d(u)^T := \begin{bmatrix}\mathbb{I}_p & \nabla_u h(u,d)^T\end{bmatrix}$ and $Q(u)$ is a metric that assigns to every $u \in \mathcal{U}$ a positive definite matrix.

The feedback law (55) approximates a projected gradient descent, by computing a descent direction w that is feasible with respect to $\mathcal{U}$ (see (56b)) and approximately feasible (up to first order) with respect to $\mathcal{X}$ (see (56c). In particular, it can be rigorously shown that (55) approximates (54) as $\alpha \to 0^+$ (Hauswirth, 2020).

Any equilibrium point of (55) is feasible and a KKT point of (4). Global convergence of (55) to KKT points of (4) is guaranteed under weak technical assumptions and without convexity, and for a small enough step size α (Häberle et al., 2021).

Note that, computing $\Sigma_\alpha(u, y)$ requires the solution of a quadratic program. However, in comparison to RTI schemes discussed in Section 2.3, the computational effort does not scale with a prediction horizon and no explicit model of the plant dynamics is required. Instead, it is enough to estimate $\nabla_u h(u, d)$. On the other hand, in general, (55) does not lend itself to a natural distributed implementation. Although, depending on the problem structure, one can solve (56) distributedly at every iteration.

Example 4.8. We reconsider the numerical test case from Example 4.7, but apply the feedback control strategy (55) instead of (52). Again, we choose a sampling period (corresponding to the step size α) of 1 minute. As before we approximate $\nabla_u h(u, d)$ based on input and output measurements and the power flow model, and we choose $Q(u) := H_d(u)^T H_d(u)$ as the metric.

The right-hand side panels in Figures 15 and 16 show how this linearized output projection method achieves similar performance to the discretized projected saddle-point flow controller (52).

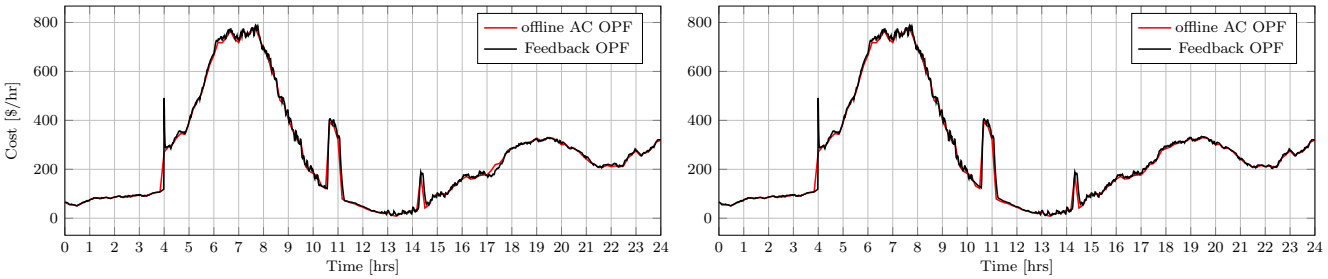


Figure 15: Cost realized by feedback-based optimization in comparison to a-posteriori ACOPF solutions using perfect information. Left: cost achieved by the projected augmented saddle-point flow (52); right: cost achieved by the linearized output projection method (55).

In fact, the constraint satisfaction is even superior. The enforcement of output constraints such as line flow limits is even more direct. Whereas for the saddle-point transient constraint violation is generally unavoidable to achieve convergence to non-zero dual variables at an equilibrium, the linearized output projection method predicts (and avoids) constraint violations based on first-order model information. This effect is illustrated in Figure 17 where the enforcement of a line flow constraint from the previous simulations are compared. It is readily noticeable that the augmented projected saddle-point flow exhibits transient constraint violation and only asymptotically satisfies the constraint. This observation corresponds to the mechanism illustrated in Figure 8d. In contrast, the linearized output projection exhibits almost no transient constraint violation and this close in behavior to the projected gradient flow it approximates (see also Figure 8e). ■

5. Conclusions and Outlook

In this article, we have surveyed various approaches for solving optimization problems in closed loop with a physical system. As a particular focus, we have presented off-the-shelf numerical optimization algorithms as dynamical systems and the conceptually simple idea of directly interconnecting these algorithms with asymptotically stable plants with well-defined input-output behavior.

Solving optimization problems online and with feedback, rather than offline and in open loop, leads to greater robustness against uncertain problem data, reduces required model information as well as computational effort, because the physical system inherently enforces certain constraints. The design of such controllers is aided by the recent rediscovery of and interest in the dynamical systems perspective of optimization algorithms. However, in contrast to the analysis of existing algorithms with tools from dynamical systems theory, deploying these algorithms as feedback controllers raises novel and unique challenges such as the study of closed-loop stability and robustness. Feedback-based optimization calls for smart control designs that meaningfully exploit physical properties of a system such as its steady-state response and saturation effects. We have particularly delved into robustness and constraint satisfaction.

5.1. Ongoing and Future Research Avenues

Even though feedback-based optimization has been an active research area for some time, many questions remain unanswered. Hence, to complete this article, we propose some worthwhile avenues for future inquiry.

Robustness against Model Uncertainty and Online Sensitivity Estimation

As already illustrated in Example 1.1, the feedback-based optimization approach advocated for in this article does not rely on a precise modeling of the plant. However, except in special cases, the controllers require an estimate of ∇h, the steady-state input-output sensitivities. This Jacobian matrix is constant if the plant is LTI. In more realistic settings, however, ∇h is often state- and time-dependent.

The results in Colombino et al. (2019b) indicate that feedback-based optimization is highly robust against inaccurate sensitivity estimates. Nevertheless, it is highly desirable learn and online adapt these input-output sensitivities using measured data rather than relying on first-principle modeling.

General Equilibrium Seeking in Closed Loop

In this article, we have focused on solving optimization problems and thus steering a physical plant to a state that solves the corresponding KKT conditions. Instead of optimality conditions, it is also possible to design controllers that seek out states that satisfy more general equilibrium conditions, such as solving variational inequalities or generalized equalities. It thus becomes possible, for example, to design feedback-controllers in a non-cooperative setting that allow players to converge to a Nash equilibrium (De Persis & Monshizadeh, 2019).

Time-Varying Non-Convex Optimization

As discussed in Section 3.5, the study of time-varying optimization problems has so far focused on convex problems (often with strongly convex objective). This offers the convenience of a unique global optimizer that forms a well-defined trajectory along time and can be tracked by appropriate controllers and algorithms.

Solving non-convex problems, such as the optimal reserve dispatch problem from Section 1.2, in closed loop

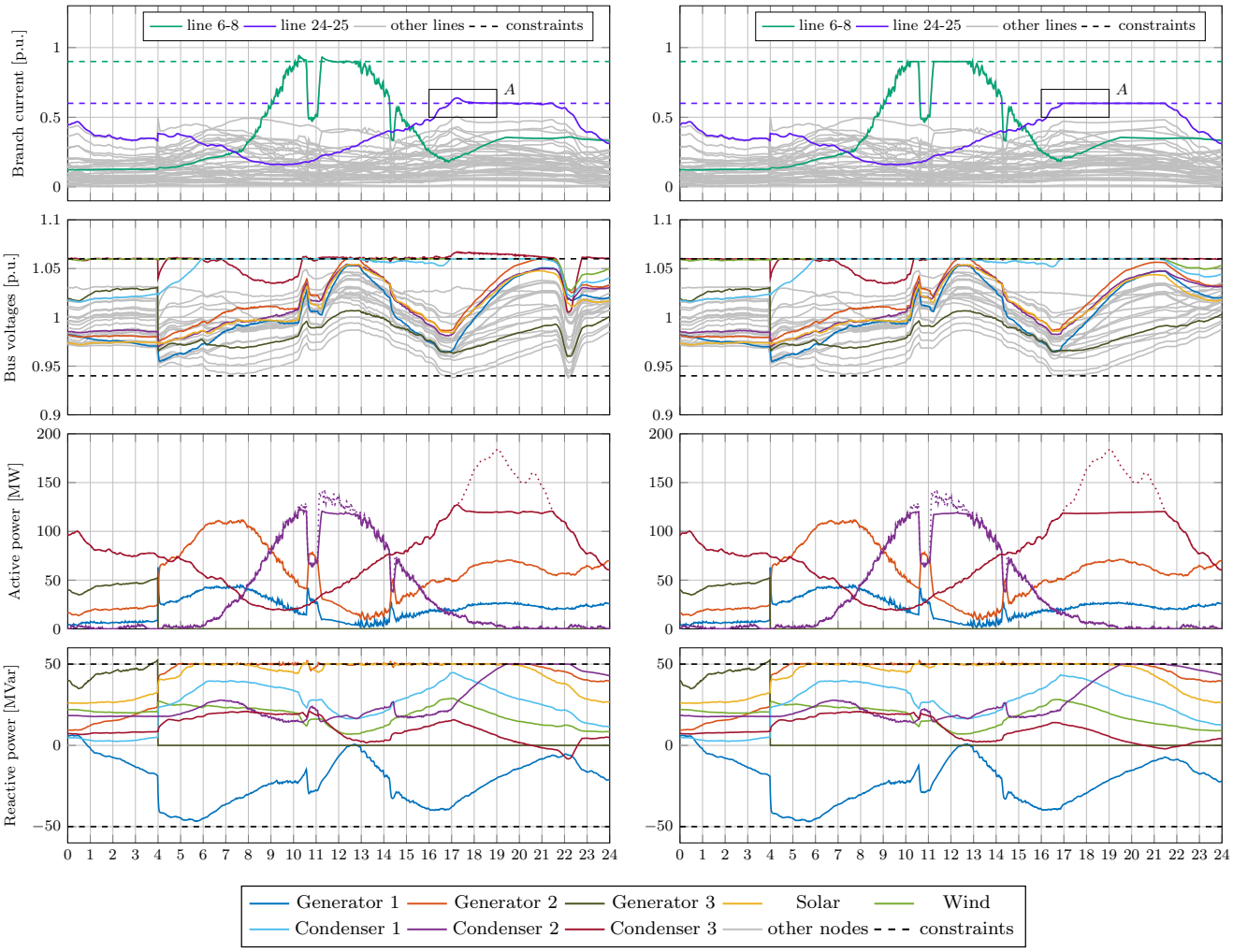


Figure 16: State trajectories realized by feedback-based optimization. Left: trajectories of the projected augmented saddle-point flow (52); right: trajectories the linearized output projection method (55). See Figure 17 for a discussion of the inset A.

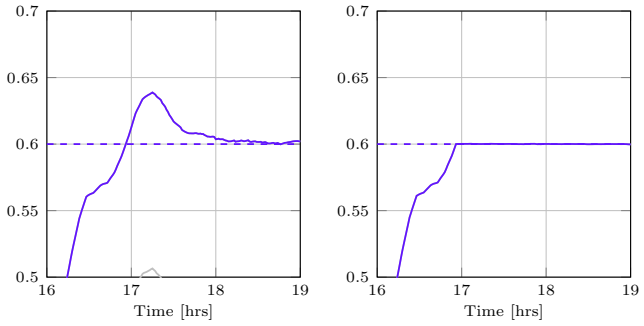


Figure 17: Comparison of line flow limit violation and enforcement (Detail A in Figure 16); Left: projected saddle-point flow; Right: linearized output projection;

poses a much more challenging problem. Stationary non-convex problems are generally hard to solve due to the existence of multiple local minimizers. Time-varying non-convex problems pose additional problems since the possible appearance and disappearance of minimizers make it generally impossible to continuously track a single global minimizer. Nevertheless, it is a practical necessity to understand how feedback-based optimization performs in non-convex settings and to develop controllers that can cope with such pathologies. Recent papers exploring this direction include Ding et al. (2019); Lin et al. (2020).

Acknowledgements

The research leading to these results was supported by ETH Zürich funds and by the Swiss Federal Office of Energy grant #SI/501708 UNICORN.

References

Absil, P., Mahony, R., & Andrews, B. (2005). Convergence of the iterates of descent methods for analytic cost functions. *SIAM Journal on Optimization*, *16*, 531–547.

Absil, P.-A., & Kurdyka, K. (2006). On the stable equilibrium points of gradient systems. *Systems & Control Letters*, *55*, 573–577.

Absil, P.-A., Mahony, R., & Sepulchre, R. (2008). *Optimization Algorithms on Matrix Manifolds*. Princeton, NJ: Princeton University Press.

Alessio, A., & Bemporad, A. (2009). A survey on explicit model predictive control. In *Nonlinear Model Predictive Control* (pp. 345–369). Berlin Heidelberg, Germany: Springer volume 384 of *Lecture Notes in Control and Information Sciences*.

Ambrosio, L., Gigli, N., & Savaré, G. (2005). *Gradient Flows in Metric Spaces and in the Space of Probability Measures*. Lecture Notes in Mathematics. Birkhäuser Boston.

Ariyur, K. B., & Krstic, M. (2003). *Real Time Optimization by Extremum Seeking Control*. (1st ed.). Hoboken, NJ: Wiley Interscience.

Arnold, D. B., Negrete-Pincetic, M., Sankur, M. D., Auslander, D. M., & Callaway, D. S. (2016). Model-free optimal control of var resources in distribution systems: An extremum seeking approach. *IEEE Transactions on Power Systems*, *31*, 3583–3593.

Arrow, K. J., Hurwicz, L., & Uzawa, H. (1958). *Studies in Linear and Nonlinear Programming*. Stanford, CA: Stanford University Press.

Åström, K. J., & Wittenmark, B. (2008). *Adaptive Control*. Dover Books on Engineering (2nd ed.). Mineola, N.Y: Dover Publications.

Aubin, J.-P., & Cellina, A. (1984). *Differential Inclusions: Set-Valued Maps and Viability Theory*. Number 264 in Grundlehren Der Mathematischen Wissenschaften. Berlin Heidelberg, Germany: Springer.

Bastianello, N., Ajalloeian, A., & Dall'Anese, E. (2020a). Distributed and inexact proximal gradient method for online convex optimization. *arXiv:2001.00870 [math]*, .

Bastianello, N., Simonetto, A., & Carli, R. (2020b). Primal and dual prediction-correction methods for time-varying convex optimization. *arXiv:2004.11709 [cs, math]*, .

Beck, A. (2017). *First-Order Methods in Optimization*. Philadelphia, PA: SIAM.

Bemporad, A., Morari, M., Dua, V., & Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, *38*.

Ben-Tal, A., El Ghaoui, L., & Nemirovski, A. (2009). *Robust Optimization*. Princeton Series in Applied Mathematics. Princeton, NJ: Princeton University Press.

Bernstein, A., Dall'Anese, E., & Simonetto, A. (2019). Online primal-dual methods with measurement feedback for time-varying convex optimization. *IEEE Transactions on Signal Processing*, *67*, 1978–1991.

Bertsekas, D. P. (2017). *Dynamic Programming and Optimal Control* volume 1. (4th ed.). Belmont, MA: Athena Scientific.

Bertsimas, D., & Freund, R. (2004). *Data, Models, and Decisions: The Fundamentals of Management Science*. Charlestown, MA: Dynamic Ideas.

Besbes, O., Gur, Y., & Zeevi, A. (2015). Non-stationary stochastic optimization. *Operations Research*, *63*, 1227–1244.

Binetti, P., Ariyur, K. B., Krstic, M., & Bernelli, F. (2003). Formation flight optimization using extremum seeking feedback. *Journal of Guidance, Control, and Dynamics*, *26*, 132–142.

Bishop, C. (2009). *Pattern Recognition and Machine Learning*. Information Science and Statistics. New York, NY: Springer.

Bloch, A. M., Brockett, R. W., & Ratiu, T. S. (1992). On the geometry of saddle point algorithms. In *IEEE Conference on Decision and Control (CDC)* (pp. 1482–1487). Tucson, AZ.

Bock, H. G., Diehl, M., Leineweber, D., & Schlöder, J. P. (2000). A direct multiple shooting method for real-time optimization of nonlinear dae processes. In *Nonlinear Model Predictive Control* (pp. 245–268). Basel, Switzerland: Springer volume 26 of *Progress in Systems and Control Theory*.

Bolognani, S., & Dörfler, F. (2015). Fast power system analysis via implicit linearization of the power flow manifold. In *53rd Annual Allerton Conference on Communication, Control, and Computing* (pp. 402–409). Monticello, IL.

Bonnans, J. F. (2019). *Convex and Stochastic Optimization*. Universitext. Cham, Switzerland: Springer.

Borkar, V. S. (2008). *Stochastic Approximation: A Dynamical Systems Viewpoint*. Number 48 in Texts and Readings in Mathematics. New Delhi: Hindustan Book Agency.

Brayton, R. K., & Moser, J. K. (1964). A theory of nonlinear networks - i. *Quarterly of Applied Mathematics*, *22*.

Bristow, D. A., Tharayil, M., & Alleyne, A. G. (2006). A survey of iterative learning control. *IEEE Control Systems Magazine*, *26*, 96–114.

Brockett, R. W. (1988). Dynamical systems that sort lists, diagonalize matrices and solve linear programming problems. In *IEEE Conference on Decision and Control (CDC)* (pp. 799–803). Austin, TX volume 1.

Brunner, F. D., Dürr, H., & Ebenbauer, C. (2012). Feedback design for multi-agent systems: A saddle point approach. In *IEEE Conference on Decision and Control (CDC)* (pp. 3783–3789). Maui, HI.

Chachuat, B., Srinivasan, B., & Bonvin, D. (2009). Adaptation strategies for real-time optimization. *Computers & Chemical Engineering*, *33*, 1557–1567.

Chang, C.-Y., Colombino, M., Cortes, J., & Dall'Anese, E. (2019). Saddle-flow dynamics for distributed feedback-based optimization. *IEEE Control Syst. Lett.*, *3*, 948–953.

Chen, X., Zhao, C., & Li, N. (2020). Distributed automatic load frequency control with optimality in power systems. *IEEE Trans. Control Netw. Syst.*, . In press.

Cherukuri, A., Gharesifard, B., & Cortés, J. (2017a). Saddle-point dynamics: Conditions for asymptotic stability of saddle points. *SIAM Journal on Control and Optimization*, *55*, 486–511.

Cherukuri, A., Mallada, E., & Cortés, J. (2016). Asymptotic convergence of constrained primal–dual dynamics. *Systems & Control Letters*, *87*, 10–15.

Cherukuri, A., Mallada, E., Low, S., & Cortés, J. (2017b). The role of convexity on saddle-point dynamics: Lyapunov function and robustness. *IEEE Trans. Autom. Control*, *63*, 2449–2464.

Clarke, F. (1990). *Optimization and Nonsmooth Analysis*. Classics in Applied Mathematics. Philadelphia, PA: SIAM.

Coito, F., Lemos, J., & Alves, S. (2005). Stochastic extremum seeking in the presence of constraints. In *16th IFAC World Congress* (pp. 276–281). Prague, Czech Republic.

Colombino, M., Dall'Anese, E., & Bernstein, A. (2019a). Online optimization as a feedback controller: Stability and tracking. *IEEE Trans. Control Network Syst.*, .

Colombino, M., Simpson-Porco, J. W., & Bernstein, A. (2019b). Towards robustness guarantees for feedback-based optimization. *arXiv 1905.07363 [math]*, .

Cortés, J. (2008). Discontinuous dynamical systems. *IEEE Control Syst. Mag.*, *28*, 36–73.

Cortés, J., & Niederländer, S. K. (2019). Distributed coordination for nonsmooth convex optimization via saddle-point dynamics. *Journal of Nonlinear Science*, *29*, 1247–1272.

Costello, S., François, G., Bonvin, D., & Marchetti, A. (2014). Modifier adaptation for constrained closed-loop systems. In *19th IFAC World Congress* (pp. 11080–11086). Cape Town, South Africa.

Dall'Anese, E., & Simonetto, A. (2018). Optimal power flow pursuit. *IEEE Trans. Smart Grid*, *9*, 942–952.

Dall'Anese, E., Simonetto, A., Becker, S., & Madden, L. (2020). Optimization and learning with information streams: Time-varying algorithms and applications. *IEEE Signal Process. Mag.*, *37*, 71–83.

De Persis, C., & Monshizadeh, N. (2019). A feedback control algorithm to steer networks to a cournot-nash equilibrium. *IEEE Trans. Control Netw. Syst.*, *6*, 1486–1497.

DeHaan, D., & Guay, M. (2005). Extremum-seeking control of state-constrained nonlinear systems. *Automatica*, *41*, 1567–1574.

Dhingra, N. K., Khong, S. Z., & Jovanovic, M. R. (2019). The

proximal augmented lagrangian method for nonsmooth composite optimization. *IEEE Transactions on Automatic Control*, *64*, 2861–2868.

Di Pillo, G., & Grippo, L. (1989). Exact penalty functions in constrained optimization. *SIAM J. Control Optim.*, *27*, 1333–1360.

Diehl, M., Bock, H., Diedam, H., & Wieber, P.-B. (2006). Fast direct multiple shooting algorithms for optimal robot control. In *Fast Motions in Biomechanics and Robotics* (pp. 65–93). Berlin Heidelberg, Germany: Springer volume 340.

Diehl, M., Bock, H., & Schlöder, J. (2005a). A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on Control and Optimization*, *43*, 1714–1736.

Diehl, M., Bock, H. G., Schlöder, J. P., Findeisen, R., Nagy, Z., & Allgöwer, F. (2002). Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, *12*, 577–585.

Diehl, M., Findeisen, R., & Allgöwer, F. (2007). A stabilizing real-time implementation of nonlinear model predictive control. In *Real-Time PDE-Constrained Optimization* (pp. 25–52). Philadelphia, PA: SIAM.

Diehl, M., Findeisen, R., Allgöwer, F., Bock, H., & Schlöder, J. (2005b). Nominal stability of real-time iteration scheme for nonlinear model predictive control. *IEE Proceedings - Control Theory and Applications*, *152*, 296–308.

Diehl, M., Findeisen, R., Schwarzkopf, S., Uslu, I., Allgöwer, F., Bock, H. G., Gilles, E. D., & Schlöder, J. P. (2003). An efficient algorithm for nonlinear model predictive control of large-scale systems part ii: Experimental evaluation for a distillation column. *at - Automatisierungstechnik*, *51*, 22–29.

Ding, Y., Lavaei, J., & Arcak, M. (2019). Escaping spurious local minimum trajectories in online time-varying nonconvex optimization. *arXiv:1912.00561 [cs, eess, math]*, .

Doyle, J. C., Francis, B., & Tannenbaum, A. (2009). *Feedback Control Theory*. New York, NY: Dover.

Dürr, H.-B., Krstić, M., Scheinker, A., & Ebenbauer, C. (2017). Extremum seeking for dynamic maps using lie brackets and singular perturbations. *Automatica*, *83*, 91–99.

Dürr, H.-B., Stanković, M. S., Ebenbauer, C., & Johansson, K. H. (2013a). Lie bracket approximation of extremum seeking systems. *Automatica*, *49*, 1538–1552.

Dürr, H.-B., Stanković, M. S., Johansson, K. H., & Ebenbauer, C. (2014). Extremum seeking on submanifolds in the euclidian space. *Automatica*, *50*, 2591–2596.

Dürr, H.-B., Zeng, C., & Ebenbauer, C. (2013b). Saddle point seeking for convex optimization problems. In *9th IFAC Symposium on Nonlinear Control Systems* (pp. 540–545). Tolouse, France.

Ellis, M., Durand, H., & Christofides, P. D. (2014). A tutorial review of economic model predictive control methods. *Journal of Process Control*, *24*, 1156–1178.

Faulwasser, T., Grüne, L., & Müller, M. A. (2018). Economic nonlinear model predictive control. *Foundations and Trends in Systems and Control*, *5*.

Fazlyab, M., Paternain, S., Preciado, V. M., & Ribeiro, A. (2016). Interior point method for dynamic constrained optimization in continuous time. In *American Control Conference (ACC)* (pp. 5612–5618). Boston, MA.

Fazlyab, M., Paternain, S., Preciado, V. M., & Ribeiro, A. (2018). Prediction-correction interior-point method for time-varying convex optimization. *IEEE Transactions on Automatic Control*, *63*, 1973–1986.

Fazlyab, M., Ribeiro, A., Morari, M., & Preciado, V. M. (2017). Analysis of optimization algorithms via integral quadratic constraints: Nonstrongly convex problems. *arXiv 1705.03615 [math]*, .

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

Feiling, J., Zeller, A., & Ebenbauer, C. (2018). Derivative-free optimization algorithms based on non-commutative maps. *IEEE Control Systems Letters*, *2*, 743–748.

Francois, G., & Bonvin, D. (2013). Measurement-based real-time optimization of chemical processes. *Advances in Chemical Engineering*, *43*.

François, G., Srinivasan, B., & Bonvin, D. (2005). Use of measurements for enforcing the necessary conditions of optimality in the presence of constraints and uncertainty. *Journal of Process Control*, *15*, 701–712.

Frank, S., & Rebennack, S. (2016). An introduction to optimal power flow: Theory, formulation, and examples. *IIE Transactions*, *48*, 1172–1197.

Frank, S., Steponavice, I., & Rebennack, S. (2012a). Optimal power flow: A bibliographic survey i: Formulations and deterministic methods. *Energy Systems*, *3*, 221–258.

Frank, S., Steponavice, I., & Rebennack, S. (2012b). Optimal power flow: A bibliographic survey ii: Non-deterministic and hybrid methods. *Energy Systems*, *3*, 259–289.

Franklin, G. F., Powell, J. D., & Emami-Naeini, A. (2010). *Feedback Control of Dynamic Systems*. (Sixth ed.). Upper Saddle River, NJ: Pearson.

Frihauf, P., Krstic, M., & Basar, T. (2012). Nash equilibrium seeking in noncooperative games. *IEEE Transactions on Automatic Control*, *57*, 1192–1207.

Frihauf, P., Krstic, M., & Başar, T. (2013). Finite-horizon lq control for unknown discrete-time linear systems via extremum seeking. *European Journal of Control*, *19*, 399–407.

Gan, L., & Low, S. H. (2016). An online gradient algorithm for optimal power flow on radial networks. *IEEE Journal on Selected Areas in Communications*, *34*, 625–638.

Gao, W., & Engell, S. (2005). Iterative set-point optimization of batch chromatography. *Computers & Chemical Engineering*, *29*, 1401–1409.

Ghaffari, A., Krstic, M., & Seshagiri, S. (2014). Power optimization and control in wind energy conversion systems using extremum seeking. *IEEE Transactions on Control Systems Technology*, *22*, 1684–1695.

Ghaffari, A., Seshagiri, S., & Krstić, M. (2015). Multivariable maximum power point tracking for photovoltaic micro-converters using extremum seeking. *Control Engineering Practice*, *35*, 83–91.

Goebel, R. (2017). Stability and robustness for saddle-point dynamics through monotone mappings. *Syst. Control. Lett.*, *108*, 16–22.

Graichen, K., & Kugi, A. (2010). Stability and incremental improvement of suboptimal mpc without terminal constraints. *IEEE Trans. Automat. Contr.*, *55*, 2576–2580.

Grégory, G., Ois, F., & Bonvin, D. (2014). Use of transient measurements for the optimization of steady-state performance via modifier adaptation. *Industrial Engineering Chemistry Research*, *53*, 5148–5159.

Grüne, L., & Pannek, J. (2010). Analysis of unconstrained nmpc schemes with incomplete optimization *. *IFAC Proceedings Volumes*, *43*, 238–243.

Grushkovskaya, V., Zuyev, A., & Ebenbauer, C. (2018). On a class of generating vector fields for the extremum seeking problem: Lie bracket approximation and stability properties. *Automatica*, *94*, 151–160.

Guay, M., Dochain, D., & Perrier, M. (2004). Adaptive extremum seeking control of continuous stirred tank bioreactors with unknown growth kinetics. *Automatica*, *40*, 881–888.

Guay, M., & Zhang, T. (2003). Adaptive extremum seeking control of nonlinear dynamic systems with parametric uncertainties. *Automatica*, *39*, 1283–1293.

Häberle, V., Hauswirth, A., Ortmann, L., Bolognani, S., & Dörfler, F. (2021). Non-convex feedback optimization with input and output constraints. *IEEE Control Syst. Lett.*, *5*, 343–348.

Hall, E. C., & Willett, R. M. (2015). Online convex optimization in dynamic environments. *IEEE Journal of Selected Topics in Signal Processing*, *9*, 647–662.

Hauswirth, A. (2020). *Optimization Algorithms as Feedback Controllers for Power System Operations*. Ph.d. thesis ETH Zürich Zürich, Switzerland.

Hauswirth, A., Bolognani, S., & Dörfler, F. (2020a). Projected dynamical systems on irregular, non-euclidean domains for nonlinear optimization. *SIAM J. Control Optim.*, . In press.

Hauswirth, A., Bolognani, S., Hug, G., & Dörfler, F. (2016). Projected gradient descent on riemannian manifolds with applications to online power system optimization. In *54th Annual Allerton Conference on Communication, Control, and Computing* (pp. 225–232). Monticello, IL.

Hauswirth, A., Bolognani, S., Hug, G., & Dörfler, F. (2020b). Timescale separation in autonomous optimization. *IEEE Trans. Autom. Control*, . In press.

Hauswirth, A., Dörfler, F., & Teel, A. R. (2020c). Anti-windup approximations of oblique projected dynamical systems for feedback-based optimization. *arXiv:2003.00478 [math.OC]*, .

Hauswirth, A., Dörfler, F., & Teel, A. R. (2020d). On the differentiability of projected trajectories and the robust convergence of non-convex anti-windup gradient flows. *IEEE Control Syst. Lett.*, *4*, 620–625.

Hauswirth, A., Dörfler, F., & Teel, A. R. (2020e). On the robust implementation of projected dynamical systems with anti-windup controllers. In *American Control Conference (ACC)* (pp. 1286–1291). Denver, CO.

Hauswirth, A., Ortmann, L., Bolognani, S., & Dörfler, F. (2020f). Limit behavior and the role of augmentation in projected saddle flows for convex optimization. In *21st IFAC World Congress*. Berlin, Germany.

Hauswirth, A., Subotić, I., Bolognani, S., Hug, G., & Dörfler, F. (2018). Time-varying projected dynamical systems with applications to feedback optimization of power systems. In *IEEE Conference on Decision and Control (CDC)* (pp. 3258–3263). Miami Beach, FL.

Hauswirth, A., Zanardi, A., Bolognani, S., Dörfler, F., & Hug, G. (2017). Online optimization in closed loop on the power flow manifold. In *IEEE PowerTech Conference*. Manchester, UK.

Helmke, U., & Moore, J. B. (1996). *Optimization and Dynamical Systems*. Communications and Control Engineering (2nd ed.). London, UK: Springer.

Hillier, F. S., & Lieberman, G. J. (2001). *Introduction to Operations Research*. (Seventh ed.). New York, NY: McGraw-Hill.

Hjalmarsson, H. (2002). Iterative feedback tuning—an overview. *Int. J. Adapt. Control Signal Process.*, *16*, 373–395.

Holding, T., & Lestas, I. (2014). On the convergence to saddle points of concave-convex functions, the gradient method and emergence of oscillations. In *IEEE Conference on Decision and Control (CDC)* (pp. 1143–1148). Los Angeles, CA.

Huneault, M., & Galiana, F. D. (1991). A survey of the optimal power flow literature. *IEEE Transactions on Power Systems*, *6*, 762–770.

Jadbabaie, A., Rakhlin, A., Shahrampour, S., & Sridharan, K. (2015). Online optimization : Competing with dynamic comparators. In *International Conference on Artificial Intelligence and Statistics (AISTATS)* (pp. 398–406). San Diego, CA.

Jokic, A., Lazar, M., & van den P.P.J. Bosch (2009). On constrained steady-state regulation : Dynamic kkt controllers. *IEEE Trans. Autom. Control*, *54*, 2250–2254.

Jongen, H. T., Jonker, P., & Twilt, F. (2001). *Nonlinear Optimization in Finite Dimensions* volume 47 of *Nonconvex Optimization and Its Applications*. Boston, MA: Springer US.

Kelly, F. P., Maulloo, A. K., & Tan, D. K. H. (1998). Rate control for communication networks: Shadow prices, proportional fairness and stability. *J. Oper. Res. Soc.*, *49*, 237–252.

Khalil, H. K. (2002). *Nonlinear Systems*. (3rd ed.). Upper Saddle River, NJ: Prentice Hall.

Killingsworth, N., Aceves, S., Flowers, D., Espinosa-Loza, F., & Krstic, M. (2009). Hcci engine combustion-timing control: Optimizing gains and fuel consumption via extremum seeking. *IEEE Transactions on Control Systems Technology*, *17*, 1350–1361.

Kokotovic, P., Khalil, H., & O'Reilly, J. (1999). *Singular Perturbation Methods in Control: Analysis and Design*. Number 25 in Classics in Applied Mathematics. Philadelphia, PA: SIAM.

Kose, T. (1956). Solutions of saddle value problems by differential equations. *Econometrica*, *24*, 59–70.

Krstic, M., Ghaffari, A., & Seshagiri, S. (2014). Extremum seeking for wind and solar energy applications. In *World Congress on Intelligent Control and Automation* (pp. 6184–6193). Shenyang, China.

Krstic, M., & Wang, H.-H. (2000). Stability of extremum seeking feedback for general nonlinear dynamic systems. *Automatica*, *36*, 595–601.

Lawrence, L. S. P., Nelson, Z. E., Mallada, E., & Simpson-Porco, J. W. (2018). Optimal steady-state control for linear time-invariant systems. In *IEEE Conference on Decision and Control (CDC)* (pp. 3251–3257). Miami Beach, FL.

Lawrence, L. S. P., Simpson-Porco, J. W., & Mallada, E. (2020). Linear-convex optimal steady-state control. *arXiv 1810.12892 [math]*, .

Lee, J. M. (1997). *Riemannian Manifolds - an Introduction to Curvature*. Number 176 in Graduate Texts in Mathematics (1st ed.). New York, NY: Springer.

Lesage-Landry, A., Shames, I., & Taylor, J. A. (2020). Predictive online convex optimization. *Automatica*, *113*, 108771.

Lessard, L., Recht, B., & Packard, A. (2016). Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, *26*, 57–95.

Lewis, F. L., & Liu, D. (Eds.) (2012). *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. Hoboken, NJ, USA: John Wiley & Sons, Inc.

Li, N., Zhao, C., & Chen, L. (2016). Connecting automatic generation control and economic dispatch from an optimization view. *IEEE Transactions on Control of Network Systems*, *3*, 254–264.

Liao-McPherson, D., Nicotra, M. M., & Kolmanovsky, I. (2020). Time-distributed optimization for real-time model predictive control: Stability, robustness, and constraint satisfaction. *Automatica*, *117*, 108973.

Lin, Y., Goel, G., & Wierman, A. (2020). Online optimization with predictions and non-convex losses. *arXiv:1911.03827 [cs, stat]*, .

Low, S. H. (2014). Convex relaxation of optimal power flow; part i: Formulations and equivalence. *IEEE Transactions on Control of Network Systems*, *1*, 15–27.

Low, S. H. (2017). *Analytical Methods for Network Congestion Control*. Number 18 in Synthesis Lectures on Communication Networks. Williston, VT: Morgan & Claypool.

Low, S. H., & Lapsley, D. E. (1999). Optimization flow control. i. basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, *7*, 861–874.

Low, S. H., Paganini, F., & Doyle, J. C. (2002). Internet congestion control. *IEEE Control Syst. Mag.*, *22*, 28–43.

Mansour, M., & Ellis, J. (2003). Comparison of methods for estimating real process derivatives in on-line optimization. *Applied Mathematical Modelling*, *27*, 275–291.

Marchetti, A., Chachuat, B., & Bonvin, D. (2009a). Modifier-adaptation methodology for real-time optimization. *Industrial & Engineering Chemistry Research*, *48*, 6022–6033.

Marchetti, A., Chachuat, B., & Bonvin, D. (2009b). Real-time optimization with estimation of experimental gradients. In *7th IFAC Symposium on Advanced Control of Chemical Processes* (pp. 524–529). Tolouse, France.

Mazzi, N., Zhang, B., & Kirschen, D. S. (2018). An online optimization algorithm for alleviating contingencies in transmission networks. *arXiv 1709.03965 [cs, math]*, .

Menta, S., Hauswirth, A., Bolognani, S., Hug, G., & Dörfler, F. (2018). Stability of dynamic feedback optimization with applications to power systems. In *56th Annual Allerton Conference on Communication, Control, and Computing* (pp. 136–143). Monticello, IL.

Mokhtari, A., Shahrampour, S., Jadbabaie, A., & Ribeiro, A. (2016). Online optimization in dynamic environments: Improved regret rates for strongly convex problems. In *IEEE Conference on Decision and Control (CDC)* (pp. 7195–7201). Las Vegas, NV.

Molzahn, D. K., Dörfler, F., Sandberg, H., Low, S. H., Chakrabarti, S., Baldick, R., & Lavaei, J. (2017). A survey of distributed optimization and control algorithms for electric power systems. *IEEE Trans. Smart Grid*, *8*, 2941–2962.

Molzahn, D. K., & Hiskens, I. A. (2019). A survey of relaxations and approximations of the power flow equations. *FNT in Electric*

Energy Systems, *4*, 1–221.

Montenbruck, J. M., Dürr, H.-B., Ebenbauer, C., & Allgöwer, F. (2014). Extremum seeking and obstacle avoidance on the special orthogonal group. In *19th IFAC World Congress* (pp. 8229–8234). Cape Town, South Africa.

Nagurney, A., & Zhang, D. (1996). *Projected Dynamical Systems and Variational Inequalities with Applications*. (1st ed.). New York, NY: Springer.

Nelson, Z. E., & Mallada, E. (2018). An integral quadratic constraint framework for real-time steady-state optimization of linear time-invariant systems. In *American Control Conference (ACC)* (pp. 597–603). Milwaukee, WI.

Nesterov, Y., & Nemirovskii, A. S. (1994). *Interior-Point Polynomial Algorithms in Convex Programming*. Number vol. 13 in SIAM Studies in Applied Mathematics. Philadelphia: Society for Industrial and Applied Mathematics.

Nocedal, J., & Wright, S. J. (2006). *Numerical Optimization*. Series in Operations Research (2nd ed.). New York, NY: Springer.

Paganini, F., Zhikui Wang, Doyle, J., & Low, S. (2005). Congestion control for high performance, stability, and fairness in general networks. *IEEE/ACM Transactions on Networking*, *13*, 43–56.

Palis, J. J., & De Melo, W. (1982). *Geometric Theory of Dynamical Systems: An Introduction*. New York, NY: Springer.

Picallo, M., Bolognani, S., & Dörfler, F. (2020). Closing the loop: Dynamic state estimation and feedback optimization of power grids. *Electric Power Systems Research*, *189*, 106753.

Popkov, A. Y. (2005). Gradient methods for nonstationary unconstrained optimization problems. *Automation and Remote Control*, *66*, 883–891.

Poveda, J., Benosman, M., & Teel, A. (2017). Distributed extremum seeking in multi-agent systems with arbitrary switching graphs. In *20th IFAC World Congress* (pp. 735–740). Toulouse, France.

Poveda, J. I., & Li, N. (2019). Inducing uniform asymptotic stability in non-autonomous accelerated optimization dynamics via hybrid regularization. In *IEEE Conference on Decision and Control (CDC)* (pp. 3000–3005). Nice, France.

Poveda, J. I., & Teel, A. R. (2017). A framework for a class of hybrid extremum seeking controllers with dynamic inclusions. *Automatica*, *76*, 113–126.

Quirynen, R., Gros, S., & Diehl, M. (2018). Inexact newton-type optimization with iterated sensitivities. *SIAM Journal on Optimization*, *28*, 74–95.

Rahili, S., & Ren, W. (2017). Distributed continuous-time convex optimization with time-varying cost functions. *IEEE Transactions on Automatic Control*, *62*, 1590–1605.

Rahili, S., Ren, W., & Ghapani, S. (2015). Distributed convex optimization of time-varying cost functions with swarm tracking behavior for continuous-time dynamics. In *54th IEEE Conference on Decision and Control (CDC)*.

Rawlings, J. B., Angeli, D., & Bates, C. N. (2012). Fundamentals of economic model predictive control. In *IEEE Conference on Decision and Control (CDC)* (pp. 3851–3861). Maui, HI.

Rockafellar, R. T., & Wets, R. J.-B. (2009). *Variational Analysis*. Number 317 in Grundlehren Der Mathematischen Wissenschaften (3rd ed.). Berlin Heidelberg, Germany: Springer.

Shakkottai, S., & Srikant, R. (2007). Network optimization and control. *FNT in Networking*, *2*, 271–379.

Shapiro, A. (1988). Sensitivity analysis of nonlinear programs and differentiability properties of metric projections. *SIAM Journal on Control and Optimization*, *26*, 628–645.

Shi, G., Lin, Y., Chung, S.-J., Yue, Y., & Wierman, A. (2020). Online optimization with memory and competitive control. *arXiv:2002.05318 [cs, eess, math, stat]*, .

Simonetto, A. (2017). Time-varying convex optimization via time-varying averaged operators. *arXiv 1704.07338 [math]*, .

Simonetto, A., & Dall'Anese, E. (2017). Prediction-correction algorithms for time-varying constrained optimization. *IEEE Transactions on Signal Processing*, *65*, 5481–5494.

Simonetto, A., Dall'Anese, E., Paternain, S., Leus, G., & Giannakis, G. B. (2020). Time-varying convex optimization: Time-structured algorithms and applications. *arXiv 2006.08500 [cs, eess, math]*, .

Simonetto, A., Koppel, A., Mokhtari, A., Leus, G., & Ribeiro, A. (2017). Decentralized prediction-correction methods for networked time-varying convex optimization. *IEEE Trans. Automat. Contr.*, *62*, 5724–5738.

Simonetto, A., & Leus, G. (2014). Distributed asynchronous time-varying constrained optimization. In *Asilomar Conference on Signals, Systems and Computers* (pp. 2142–2146). Pacific Grove, CA.

Simpson-Porco, J. W. (2016). Input/output analysis of primal-dual gradient algorithms. In *54th Annual Allerton Conference on Communication, Control, and Computing* (pp. 219–224). Monticello, IL.

Simpson-Porco, J. W. (2020). Analysis and synthesis of low-gain integral controllers for nonlinear systems with application to feedback-based optimization. *arXiv 2003.01348 [math]*, .

Simpson-Porco, J. W. (2021). On stability of distributed-averaging proportional-integral frequency control in power systems. *IEEE Control Syst. Lett.*, *5*, 677–682.

Smale, S. (1972). On the mathematical foundations of electrical circuit theory. *Journal of Differential Geometry*, *7*, 193–210.

Stankovic, M. S., Johansson, K. H., & Stipanovic, D. M. (2012). Distributed seeking of nash equilibria with applications to mobile sensor networks. *IEEE Transactions on Automatic Control*, *57*, 904–919.

Stankovic, M. S., & Stipanovic, D. M. (2009). Discrete time extremum seeking by autonomous vehicles in a stochastic environment. In *IEEE Conference on Decision and Control (CDC) and Chinese Control Conference* (pp. 4541–4546). Shanghai, China.

Stanković, M. S., & Stipanović, D. M. (2010). Extremum seeking under stochastic noise and applications to mobile sensors. *Automatica*, *46*, 1243–1251.

Stegink, T., De Persis, C., & van der Schaft, A. (2017). A unifying energy-based approach to stability of power grids with market dynamics. *IEEE Trans. Automat. Contr.*, *62*, 2612–2622.

Stegink, T. W., Damme, T. V., & Persis, C. D. (2018). Convergence of projected primal-dual dynamics with applications in data centers. In *7th IFAC Workshop on Distributed Estimation and Control in Networked Systems (NECSYS)* (pp. 88–93). Groningen, The Netherlands.

Subotić, I., Hauswirth, A., & Dörfler, F. (2020). Quantitative sensitivity bounds for nonlinear programming and time-varying optimization. *arXiv:2006.10693 [math]*, .

Tan, Y., Moase, W. H., Manzie, C., Nesic, D., & Mareels, I. M. Y. (2010). Extremum seeking from 1922 to 2010. In *Proceedings of the 29th Chinese Control Conference* (pp. 14–26). Beijing, China.

Tan, Y., Nešić, D., & Mareels, I. (2006). On non-local stability properties of extremum seeking control. *Automatica*, *42*, 889–903.

Tang, A., Wang, J., Low, S. H., & Chiang, M. (2007). Equilibrium of heterogeneous congestion control: Existence and uniqueness. *IEEE/ACM Transactions on Networking*, *15*, 824–837.

Tang, Y., Dall'Anese, E., Bernstein, A., & Low, S. (2018a). Running primal-dual gradient method for time-varying nonconvex problems. *arXiv 1812.00613 [math]*, .

Tang, Y., Dall'Anese, E., Bernstein, A., & Low, S. H. (2018b). A feedback-based regularized primal-dual gradient method for time-varying nonconvex optimization. In *IEEE Conference on Decision and Control (CDC)* (pp. 3244–3250). Miami Beach, FL.

Tang, Y., Dvijotham, K., & Low, S. (2017). Real-time optimal power flow. *IEEE Trans. Smart Grid*, *8*, 2963–2973.

Teel, A., & Popovic, D. (2001). Solving smooth and nonsmooth multivariable extremum seeking problems by the methods of nonlinear programming. In *American Control Conference (ACC)* (pp. 2394–2399). Arlington, VA.

Tondel, P., Johansen, T. A., & Bemporad, A. (2003). An algorithm for multi-parametric quadratic programming and explicit mpc solutions. *Automatica*, *39*, 489–497.

Trip, S., Cucuzzella, M., De Persis, C., van der Schaft, A., & Ferrara, A. (2019). Passivity-based design of sliding modes for optimal load frequency control. *IEEE Transactions on Control Systems Technology*, *27*, 1893–1906.

Van Cutsem, T., & Vournas, C. (1998). *Voltage Stability of Electric*

Power Systems. Dordrecht, The Netherlands: Springer.

van der Schaft, A. J. (2011). On the relation between port-hamiltonian and gradient systems. *IFAC Proceedings Volumes*, *44*, 3321–3326.

Venets, V. (1985). Continuous algorithms for solution of convex optimization problems and finding saddle points of contex-coneave functions with the use of projection operations. *Optimization*, *16*, 519–533.

Vinnicombe, G. (2002). On the stability of networks operating tcp-like congestion control. In *15th IFAC World Congress* (pp. 217–222). Barcelona, Spain.

Wang, J., & Elia, N. (2011). A control perspective for centralized and distributed convex optimization. In *IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)* (pp. 3800–3805). Orlando, FL.

Wei, D. X., Jin, C., Low, S. H., & Hegde, S. (2006). Fast tcp: Motivation, architecture, algorithms, performance. *IEEE/ACM Transactions on Networking*, *14*, 1246–1259.

Wei, E., Ozdaglar, A., & Jadbabaie, A. (2013). A distributed newton method for network utility maximization—part ii: Convergence. *IEEE Transactions on Automatic Control*, *58*, 2176–2188.

Wen, J. T., & Arcak, M. (2004). A unifying passivity framework for network flow control. *IEEE Transactions on Automatic Control*, *49*, 162–174.

Wittenmark, B., & Urquhart, A. (1995). Adaptive extremal control. In *Proceedings of 1995 34th IEEE Conference on Decision and Control* (pp. 1639–1644). New Orleans, LA, USA: IEEE volume 2.

Zanelli, A., Dinh, Q. T., & Diehl, M. (2020). A Lyapunov function for the combined system-optimizer dynamics in nonlinear model predictive control. *arXiv 2004.08578 [math]*, .

Zargham, M., Ribeiroy, A., & Jadbabaie, A. (2013). Accelerated dual descent for constrained convex network flow optimization. In *IEEE Conference on Decision and Control (CDC)* (pp. 1037–1042). Florence, Italy.

Zavala, V. M., & Biegler, L. T. (2009). Nonlinear programming strategies for state estimation and model predictive control. In *Nonlinear Model Predictive Control* (pp. 419–432). Berlin Heidelberg, Germany: Springer volume 384.

Zhang, X., Papachristodoulou, A., & Li, N. (2018). Distributed control for reaching optimal steady state in network systems: An optimization approach. *IEEE Transactions on Automatic Control*, *63*, 864–871.

Zhao, C., Mallada, E., Low, S., & Bialek, J. (2016). A unified framework for frequency control and congestion management. In *2016 Power Systems Computation Conference (PSCC)* (pp. 1–7). Genoa, Italy: IEEE.

Zimmerman, R. D., Murillo-Sanchez, C. E., & Thomas, R. J. (2011). Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on Power Systems*, *26*, 12–19.

Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In *International Conference on Machine Learning (ICML)*. Washington DC.