Control-Based Design of Online Optimization Algorithms

Nicola Bastianello

KTH Royal Institute of Technology, Sweden

November 14, 2025

CentraleSupélec, France

In this talk, I will discuss my joint work with:

Sandro Zampieri
(Univ. of Padova)

Ruggero Carli
(Univ. of Padova)

Umberto Casti
(Univ. of Padova)

- Wouter J. A. van Weerelt (KTH)

Partially funded through:

- Research Projects of National Interest (PRIN), Ministry of University and Research (Italy), project n. 2017CWMF93
- HORIZON Research and Innovation Actions, European Commission (EU), project no. 101070162

Outline

From static to online

(Static) convex optimization is a fundamental tool in many engineering applications:

- *e.g.* machine learning, power systems, transportation networks, signal/image processing, ...

However, recent technological advances in these applications have introduced new challenges:

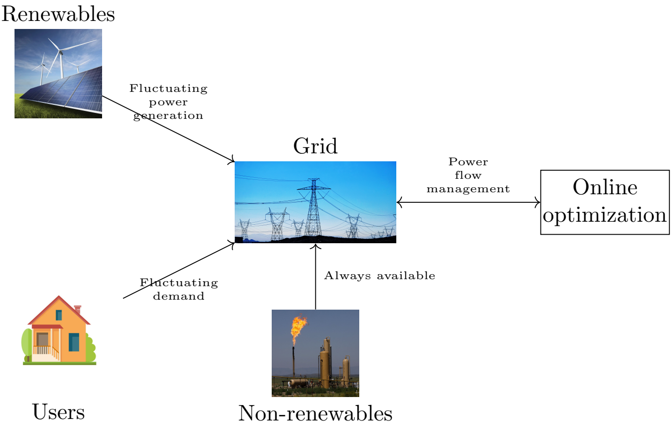
- we deal with *large-scale*, *interconnected*, *rapidly evolving* systems

for which traditional optimization techniques are not sufficient:

- there is a need to *revisit* and *redesign* them

 Introduction

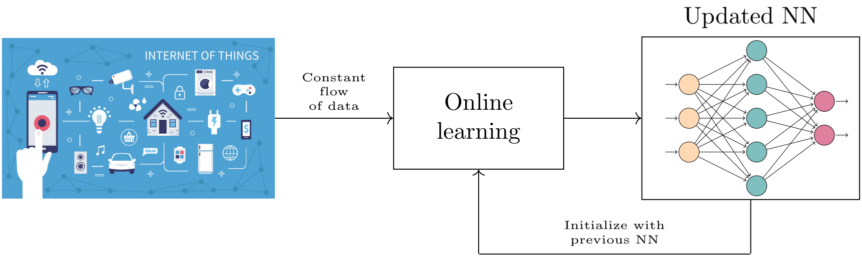
Example: Power grids



E. Dall'Anese and A. Simonetto. "Optimal Power Flow Pursuit". In: *IEEE Transactions on Smart Grid* 9.2 (Mar. 2018), pp. 942–952.

A. Lesage-Landry and D. S. Callaway. "Dynamic and Distributed Online Convex Optimization for Demand Response of Commercial Buildings". In: *IEEE Control Systems Letters* 4.3 (July 2020), pp. 632–637.

Example: Online learning



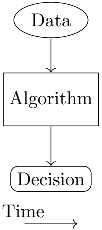
S. Shalev-Shwartz. "Online Learning and Online Convex Optimization". In: *Foundations and Trends® in Machine Learning* 4.2 (2011), pp. 107–194.

R. Dixit et al. "Online Learning with Inexact Proximal Online Gradient Descent Algorithms". In: *IEEE Transactions on Signal Processing* 67.5 (2019), pp. 1338 –1352.

Design constraints

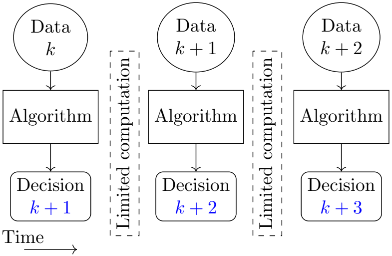
Static optimization

- static data, collected once
- large time for computation



Online optimization

- streaming, time-varying data
- very limited computation time



▶ We need algorithms that can handle *dynamic problems* at the *relevant time-scales*

Unstructured *v.* structured

We can classify online algorithms in:

- *Unstructured*:
  - we tweak static algorithms for the online set-up
  - they are "model-agnostic"
- *Structured*:
  - we design tailored algorithms
  - they are "model-based"

- I will talk about using **control theory** to design structured algorithms

A. Simonetto et al. "Time-Varying Convex Optimization: Time-Structured Algorithms and Applications". In: *Proceedings of the IEEE* 108.11 (Nov. 2020), pp. 2032–2048.

Problem formulation

Formally, we are interested in solving the sequence of problems

$$\boldsymbol{x}_k^* = \arg\min_{\boldsymbol{x} \in \mathbb{R}^n} f_k(\boldsymbol{x}), \quad k \in \mathbb{N}$$

where a new problem is revealed every T_s seconds

Assumptions

- $\{f_k\}_{k \in \mathbb{N}}$ are $\underline{\lambda}$-strongly convex and $\bar{\lambda}$-smooth
- bounded rate of change: $\exists \mathit{\Gamma}_1 \geq 0$ such that

$$\|\nabla f_{k+1}(\boldsymbol{x}) - \nabla f_k(\boldsymbol{x})\| \leq \mathit{\Gamma}_1, \quad \forall k \in \mathbb{N},\ \boldsymbol{x} \in \mathbb{R}^n$$

- 1) unique solution trajectory $\{\boldsymbol{x}_k^*\}_{k \in \mathbb{N}}$; 2) bounded $\left\|\boldsymbol{x}_k^* - \boldsymbol{x}_{k-1}^*\right\|$

Solving online optimization

What do we mean by "solving an online problem"?

- *Track the optimal trajectory – within some precision and in real time*

Formally, we design an algorithm $\mathcal{A}_k : \mathbb{R}^n \to \mathbb{R}^n$

$$\boldsymbol{x}_{k+1} = \mathcal{A}_k(\boldsymbol{x}_k)$$

so that:

$$\limsup_{k\to\infty} \|\boldsymbol{x}_k - \boldsymbol{x}_k^*\| \leq B < \infty.$$

In the following:

- we want our algorithm to be *predictive*: $\boldsymbol{x}_{k+1}$ will be computed during $[kT_\mathrm{s}, (k+1)T_\mathrm{s})$ using f_k – not f_{k+1}

Unstructured: online gradient

A first approach is the online gradient:

$$\boldsymbol{x}_{k+1} = \mathcal{A}_k(\boldsymbol{x}_k) := \boldsymbol{x}_k - \alpha \nabla f_k(\boldsymbol{x}_k), \quad k \in \mathbb{N}$$

with $\alpha < 2/\bar{\lambda}$

Its tracking error is bounded by

$$\limsup_{k \to \infty} \|\boldsymbol{x}_k - \boldsymbol{x}_k^*\| \leq B := \frac{1}{1-\zeta} \frac{\mathit{\Gamma}_1 T_{\mathrm{s}}}{\underline{\lambda}}$$

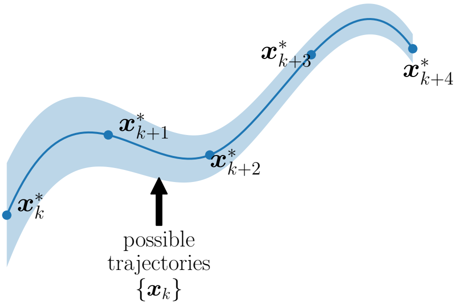
where $\zeta := \max\{|1 - \alpha \underline{\lambda}|, |1 - \alpha \bar{\lambda}|\} \in (0, 1)$

E. Dall'Anese et al. "Optimization and Learning With Information Streams: Time-varying algorithms and applications". In: *IEEE Signal Processing Magazine* 37.3 (May 2020), pp. 71–83.

Unstructured: online gradient (cont'd)

As the online gradient highlights:

- tracking in general is not exact: we can only reach a *neighborhood* of the optimal trajectory



The question now is

- Can we design algorithms with *smaller (or zero) tracking error*?

Outline

Control and optimization

Control and optimization come together in two different scenarios:

- *optimization as a tool*: we design the control input by solving an optimization problem[1]
  - *e.g.* MPC: we choose the control input by *solving an optimization problem that changes as the state of the system changes*
- *control-based design*: we use control theory to design optimization algorithms[2] $\Leftarrow$ this is what we do

We change our perspective:

- the online optimization problem is the "plant"

[1] A. Hauswirth et al. "Timescale Separation in Autonomous Optimization". In: *IEEE Transactions on Automatic Control* 66.2 (2021), pp. 611–624.

[2] L. Lessard, B. Recht, and A. Packard. "Analysis and Design of Optimization Algorithms via Integral Quadratic Constraints". In: *SIAM Journal on Optimization* 26.1 (Jan. 2016), pp. 57–95.

Online quadratic problems

We start our exploration by restricting to online *quadratic* problems

$$\boldsymbol{x}_k^* = \underset{\boldsymbol{x} \in \mathbb{R}^n}{\arg\min} \frac{1}{2} \boldsymbol{x}^\top \boldsymbol{A} \boldsymbol{x} + \langle \boldsymbol{b}_k, \boldsymbol{x} \rangle, \quad k \in \mathbb{N}$$

Assumptions

- $\underline{\lambda} \boldsymbol{I} \preceq \boldsymbol{A} = \boldsymbol{A}^\top \preceq \bar{\lambda} \boldsymbol{I}$
- $\boldsymbol{b}_k$ has transfer function

$$\boldsymbol{B}(z) = \frac{\boldsymbol{B}_{\mathrm{N}}(z)}{B_{\mathrm{D}}(z)}, \quad \boldsymbol{B}_{\mathrm{N}}(z) \in \mathbb{R}^n[z], B_{\mathrm{D}}(z) \in \mathbb{R}[z]$$

with $B_{\mathrm{D}}(z) = z^m + \sum_{i=0}^{m-1} b_i z^i$ $\Leftarrow$ this is all we need to know

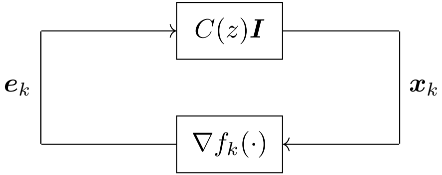
Remark: we assume that $\boldsymbol{A}^{-1}$ is *not* accessible

Control scheme

In order to achieve *zero* tracking error we need

$$\limsup_{k\to\infty} \nabla f_k(\boldsymbol{x}_k) = \boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}_k = 0$$

To this end we employ the control scheme:



where

- the gradient ∇f_k is the *plant*, and
- we need to design the controller (via its transfer function $C(z) \in \mathbb{R}[z]$)

Control design

With some manipulations, the Z-transform of $\boldsymbol{e}_k = \nabla f_k(\boldsymbol{x}_k)$ is given by

$$\boldsymbol{E}(z) = \left(\boldsymbol{I} - C(z)\boldsymbol{A}\right)^{-1} \boldsymbol{B}(z)$$

We choose the controller

$$C(z) = \frac{C_{\mathrm{N}}(z)}{B_{\mathrm{D}}(z)}, \quad \text{with} \quad C_{\mathrm{N}}(z) = \sum_{i=0}^{m-1} c_i z^i$$

where the denominator serves as *internal model*, and we get

$$\boldsymbol{E}(z) = \left(B_{\mathrm{D}}(z)\boldsymbol{I} - C_{\mathrm{N}}(z)\boldsymbol{A}\right)^{-1} \boldsymbol{B}_{\mathrm{N}}(z)$$

- the goal then is to design $C_{\mathrm{N}}(z)$ to stabilize the feedback

Stabilizing controller

The poles of $(B_{\mathrm{D}}(z)\boldsymbol{I} - C_{\mathrm{N}}(z)\boldsymbol{A})^{-1}$ are stable if the roots of

$$B_{\mathrm{D}}(z) - C_{\mathrm{N}}(z)\lambda, \quad \forall\lambda \in [\underline{\lambda}, \bar{\lambda}]$$

are inside the unit circle

- this is a linear *robust control* problem

By using[3]

- the controller (if it exists) can be found by solving a set of **two** LMIs
- the LMIs scale with the degree of $B_{\mathrm{D}}(z)$ **not** with the size of the problem

[3] M. de Oliveira, J. Bernussou, and J. Geromel. "A new discrete-time robust stability condition". In: *Systems & Control Letters* 37.4 (July 1999), pp. 261–265.

Control-based algorithm

We can finally characterize the online algorithm designed so far as

$$
\begin{aligned}
\boldsymbol{w}_{k+1} &= \left(\begin{bmatrix} 0 & 1 & & \\ & & \ddots & \\ 0 & \cdots & 0 & 1 \\ -b_0 & \cdots & \cdots & -b_{m-1} \end{bmatrix} \otimes \boldsymbol{I} \right) \boldsymbol{w}_k + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \otimes \nabla f_k(\boldsymbol{x}_k) \\
\boldsymbol{x}_{k+1} &= \left(\begin{bmatrix} c_0 & \cdots & c_{m-1} \end{bmatrix} \otimes \boldsymbol{I} \right) \boldsymbol{w}_{k+1}
\end{aligned}
$$

where

- $\boldsymbol{w}$ serves as the state of the internal model
- and $c_0, \ldots, c_{m-1}$ are the coefficients of the stabilizing controller

Remark: the algorithm only accesses an *oracle* of the gradient

Convergence results

Convergence

Given a stabilizing controller, the online algorithm verifies

$$\limsup_{k\to\infty} \|\boldsymbol{x}_k - \boldsymbol{x}_k^*\| = 0$$

- What if an *inexact internal model* is used?

Convergence: inexact model

Using the inexact model $\hat{B}_{\mathrm{D}}(z) = z^m + \sum_{i=0}^{m-1} \hat{b}_i z^i$ and if $\|\boldsymbol{b}_k\| \leq \beta$, we have

$$\limsup_{k\to\infty} \|\boldsymbol{x}_k - \boldsymbol{x}_k^*\| \leq O\left(\beta \, \|\boldsymbol{d}\|\right)$$

where $\boldsymbol{d} = \begin{bmatrix} b_0 - \hat{b}_0 & \cdots & b_{m-1} - \hat{b}_{m-1} \end{bmatrix}$

Application to general problems?

So far we focused on the quadratic problem

$$\boldsymbol{x}_k^* = \arg\min_{\boldsymbol{x} \in \mathbb{R}^n} \frac{1}{2} \boldsymbol{x}^\top \boldsymbol{A} \boldsymbol{x} + \langle \boldsymbol{b}_k, \boldsymbol{x} \rangle, \quad k \in \mathbb{N}$$

as a means to *design the proposed online algorithm*

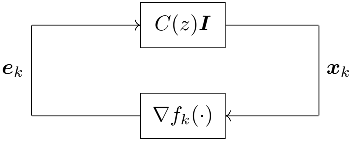
The question now is:

- can we apply it to more general problems?

Application to general problems? (cont'd)

Yes, we can apply to any cost:

- the algorithm only depends on ∇f_k



Indeed, recall the algorithm we designed is characterized by:

$$\boldsymbol{w}_{k+1} = \left(\begin{bmatrix} 0 & 1 & & \\ & & \ddots & \\ 0 & \cdots & 0 & 1 \\ -b_0 & \cdots & \cdots & -b_{m-1} \end{bmatrix} \otimes \boldsymbol{I} \right) \boldsymbol{w}_k + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \otimes \nabla f_k(\boldsymbol{x}_k)$$

$$\boldsymbol{x}_{k+1} = \left(\begin{bmatrix} c_0 & \cdots & c_{m-1} \end{bmatrix} \otimes \boldsymbol{I} \right) \boldsymbol{w}_{k+1}$$

- Question: what convergence guarantees can we give?

Convergence results: beyond quadratic

We provide here a first convergence analysis

- for the class of *"perturbed quadratic"* costs

$$f_k(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \langle \boldsymbol{b}_k, \boldsymbol{x} \rangle + \varphi_k(\boldsymbol{x})$$

Small gain assumptions

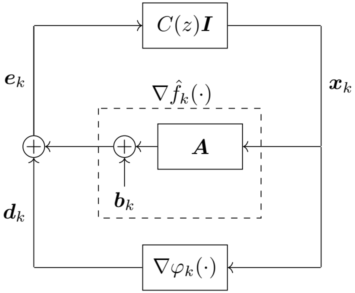
- f_k is $\underline{\lambda}$-strongly convex and $\bar{\lambda}$-smooth
- we have a model for $\{\boldsymbol{b}_k\}$ and $\|\boldsymbol{b}_k\| \leq \beta$
- there exists $\gamma > 0$ such that $\|\nabla \varphi_k(\boldsymbol{x})\| \leq \gamma \|\boldsymbol{x}\|$

Convergence results: beyond quadratic (cont'd)

The cost $f_k(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \langle \boldsymbol{b}_k, \boldsymbol{x}\rangle + \varphi_k(\boldsymbol{x})$ can be interpreted as a quadratic cost

$$\hat{f}_k(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \langle \boldsymbol{b}_k, \boldsymbol{x}\rangle$$

with a (feedback) disturbance $\varphi_k(\boldsymbol{x})$



- can we guarantee stability?

Convergence results: beyond quadratic (cont'd)

If we design the controller $C(z) = \frac{C_N(z)}{C_D(z)}$ such that

1. *internal model*: $C_D(z)$ includes all the poles of $B_D(z)$
2. *stability*: $C_N(z)$ stabilizes the feedback without disturbance
3. *small gain*: $\left\| C(z)(\boldsymbol{I} - C(z)\boldsymbol{A})^{-1} \right\|_\infty \leq 1/\gamma$

Then the output $\boldsymbol{x}_k$ of the algorithm verifies

$$\limsup_{k\to\infty} \|\boldsymbol{x}_k - \boldsymbol{x}_k^*\| \leq \left\| (\boldsymbol{I} - C(z)\boldsymbol{A})^{-1} \right\|_\infty \frac{\beta\gamma \left\| C(z)(\boldsymbol{I} - C(z)\boldsymbol{A})^{-1} \right\|_\infty}{1 - \gamma \left\| C(z)(\boldsymbol{I} - C(z)\boldsymbol{A})^{-1} \right\|_\infty}$$

Designing the controller

For stability of the feedback loop we need:

1. *internal model*: $C_D(z)$ includes all the poles of $B_D(z)$
2. *stability*: $C_N(z)$ stabilizes the feedback without disturbance
3. *small gain*: $\left\| C(z)(\boldsymbol{I} - C(z)\boldsymbol{A})^{-1} \right\|_\infty \leq 1/\gamma$

This means that we can choose $C_D(z) = B_D(z)P(z)$

- where $B_D(z)$ accounts for the poles of the linear term 1
- and $P(z)$ is a new design parameter

The goal then is to

- choose $P(z)$ to improve convergence for the "quadratically perturbed" problems (and verify 2, 3)

Time-varying $\boldsymbol{b}_k$

We consider problem

$$\boldsymbol{x}_k^* = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \langle \boldsymbol{b}_k, \boldsymbol{x}\rangle, \quad k \in \mathbb{N}$$

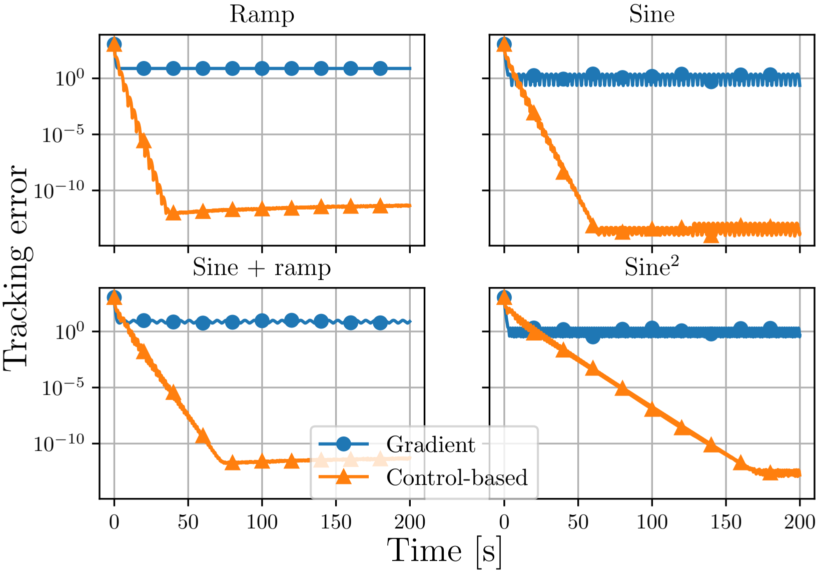
with

- $n = 500$
- $\underline{\lambda} = 1$, $\bar{\lambda} = 10$

and, four different models of $\boldsymbol{b}_k$:

1. ramp: $\boldsymbol{b}_k = k\bar{\boldsymbol{b}}$
2. sine: $\boldsymbol{b}_k = \sin(\omega k)\,\mathbf{1}$, $\omega = 1$
3. sine+ramp: $\boldsymbol{b}_k = \sin(\omega k)\,\mathbf{1} + k\bar{\boldsymbol{b}}$
4. squared sine: $\boldsymbol{b}_k = \sin^2(\omega k)\,\mathbf{1}$

Time-varying $\boldsymbol{b}_k$ (cont'd)

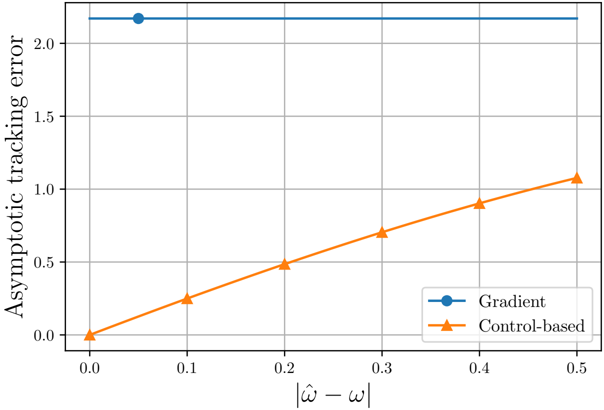


Time-varying $\boldsymbol{b}_k$ (cont'd)

Consider the inexact sinusoidal model:

$$\hat{B}_{\mathrm{D}}(z) = z^2 - 2\cos(\hat{\omega})z - 1$$

where $\hat{\omega} \in [0.5, 1]$ (recalling $\omega = 1$)



Non-quadratic problem

We consider the "perturbed quadratic" cost

$$f_k(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \langle \boldsymbol{b}, \boldsymbol{x} \rangle + \sin(\omega k) \log\left(1 + \exp\langle \boldsymbol{c}, \boldsymbol{x} \rangle\right)$$

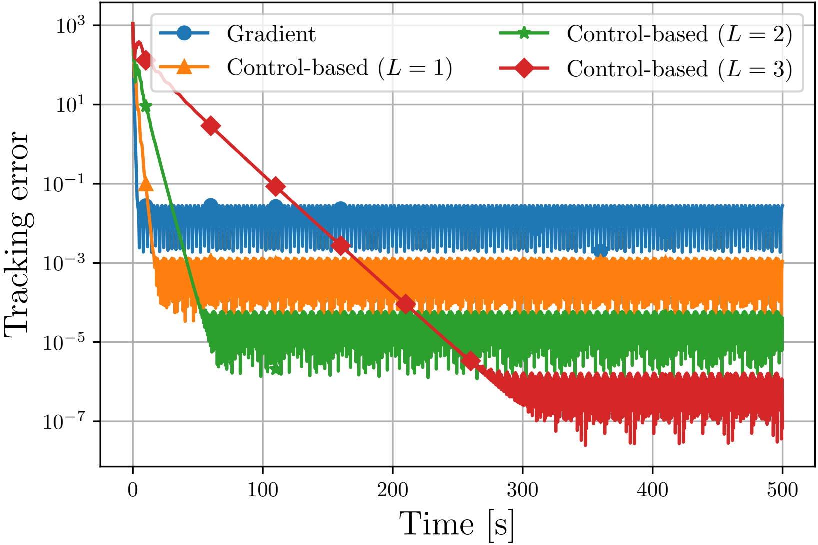
Considering that the problem is periodic

- as internal model we choose the first L terms of the Fourier series of a periodic signal

$$C_{\mathrm{D}}(z) = (z-1)\prod_{\ell=1}^{L}(z^2 - 2\cos(\ell\omega)z + 1)$$

with $L = 1, 2, 3$

Non-quadratic problem (cont'd)



Outline

Constrained problems

- Can we apply our approach to *constrained* problems?
  - Yes if linear equality constraints $\boldsymbol{G}\boldsymbol{x} = \boldsymbol{h}_k$
  - More difficult with inequality constraints $\boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h}_k$

Consider the second case:

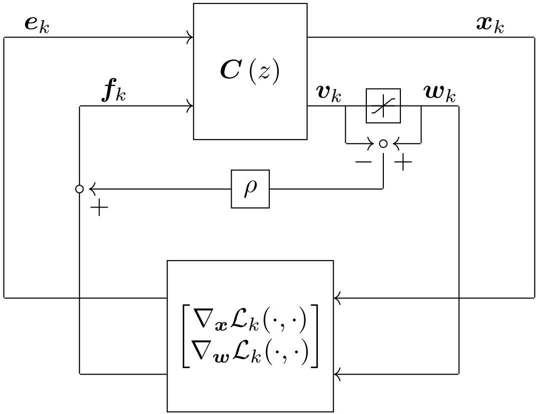
$$\min f_k(\boldsymbol{x}) \quad \text{s.t.} \quad \boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h}_k$$

which can be reformulated as

$$\min_{\boldsymbol{x}} \max_{\boldsymbol{w} \geq 0} \mathcal{L}_k\left(\boldsymbol{x}, \boldsymbol{w}\right) := f_k\left(\boldsymbol{x}\right) + \boldsymbol{w}^\top \left(\boldsymbol{G}\boldsymbol{x} - \boldsymbol{h}_k\right)$$

- we still need to ensure $\lim_{k\to\infty} \left\| \begin{bmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}_k\left(\boldsymbol{x}_k, \boldsymbol{w}_k\right) \\ \nabla_{\boldsymbol{w}} \mathcal{L}_k\left(\boldsymbol{x}_k, \boldsymbol{w}_k\right) \end{bmatrix} \right\| = 0$
- **but**: nonnegativity of $\boldsymbol{w}$ acts as *saturation* $\Rightarrow$ we can apply *anti-windup*

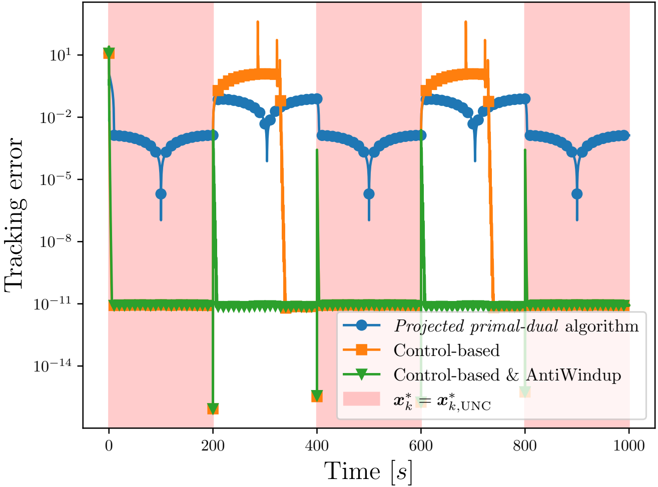
Algorithm design



- if $\rho = 0$: controller without anti-windup

Numerical results

We consider $\min f_k(\boldsymbol{x})$ s.t. $\boldsymbol{G}\boldsymbol{x} \leq \boldsymbol{h}_k$ with $\boldsymbol{b}_k$ and $\boldsymbol{h}_k$ *sinusoidal signals*



Outline

Identifying the internal model

So far we know that: exact convergence requires an *exact internal model*

- How to get this information in practice?

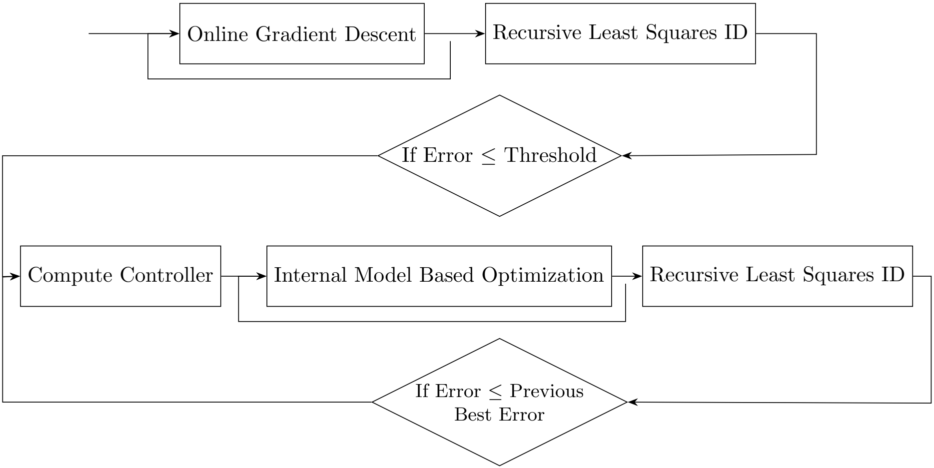
Consider the usual quadratic problem

$$\boldsymbol{x}_k^* = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} + \langle \boldsymbol{b}_k, \boldsymbol{x}\rangle, \quad k \in \mathbb{N}$$

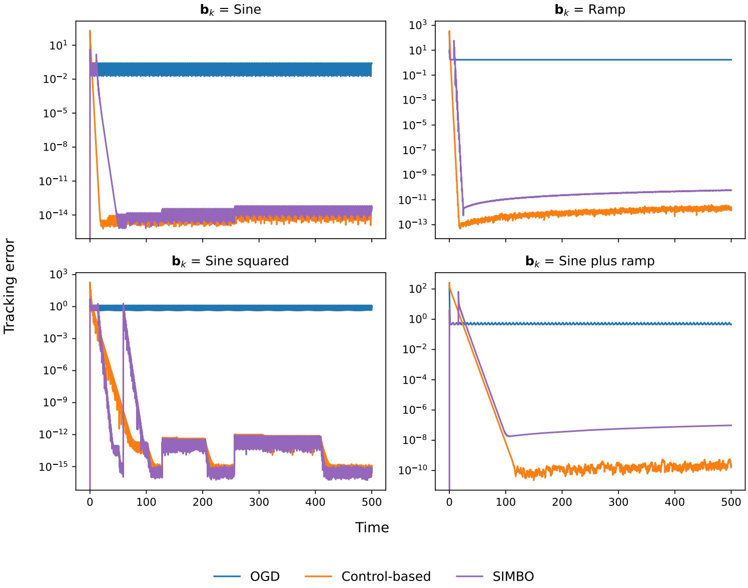
We need to identify the denominator $B_{\mathrm{D}}(z) = z^m + \sum_{i=0}^{m-1} b_i z^i$ of the transfer function:

$$\boldsymbol{B}(z) = \frac{\boldsymbol{B}_{\mathrm{N}}(z)}{B_{\mathrm{D}}(z)}, \quad \boldsymbol{B}_{\mathrm{N}}(z) \in \mathbb{R}^n[z], B_{\mathrm{D}}(z) \in \mathbb{R}[z]$$

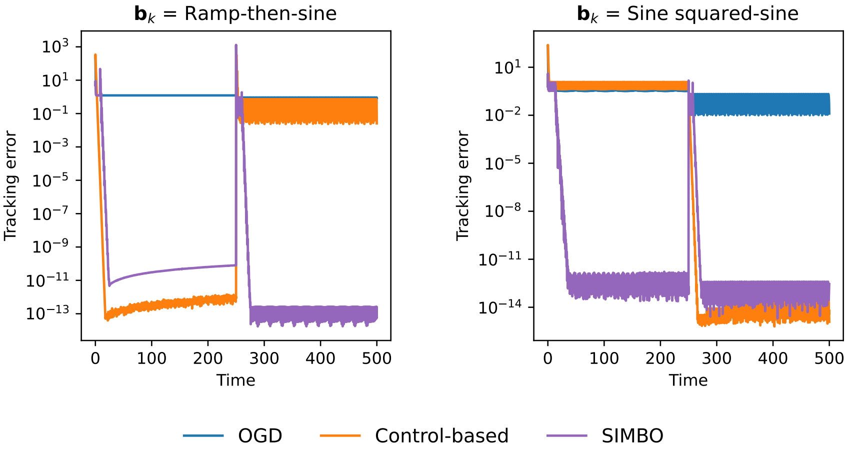
Identifying the internal model (cont'd)



Numerical results



Numerical results



Outline

Conclusions

1. The challenge of online optimization
   - tracking time-varying solution *within some precision and in real time*
2. Structured algorithms
   - exploiting a *model* of the problem allows to improve performance
3. Control for online optimization
   - leverage powerful control tools to design online algorithms
   - *e.g.* internal model, robust control, small gain theorem, anti-windup

Future directions (**we need you**):

- convergence guarantees for inequality *constrained* problems
  - analyzing the impact of **anti-windup**
- convergence guarantees for SIMBO
- applying non-linear model identification in[4]

[4] G. Bianchin and B. V. Scoy. *The Internal Model Principle of Time-Varying Optimization*. arXiv: 2407.08037 [math].

Thank you!

For more info: https://bastianello.me

References:

- N. Bastianello, R. Carli, and S. Zampieri, "Internal Model-Based Online Optimization," IEEE Trans. Automat. Contr., vol. 69, no. 1, pp. 689–696, Jan. 2024.
- U. Casti, N. Bastianello, R. Carli, and S. Zampieri, "A control theoretical approach to online constrained optimization," Automatica, vol. 176, p. 112107, 2025.
- W. J. A. van Weerelt and N. Bastianello, "Control-Based Online Distributed Optimization," to be presented at CDC'25 (arXiv:2508.15498).
- W. J. A. van Weerelt, L. Zhang, S. Zhang, N. Bastianello. "Self-Identifying Internal Model-Based Online Optimization" [available soon]