# Challenges in Model Agnostic Controller Learning for Unstable Systems

Mario Sznaier, *Fellow, IEEE*, and Mustafa Bozdag, *Graduate Student Member, IEEE*

***Abstract*—Model agnostic controller learning, for instance by direct policy optimization, has been the object of renewed attention lately, since it avoids a computationally expensive system identification step. Indeed, direct policy search has been empirically shown to lead to optimal controllers in a number of cases of practical importance. However, to date, these empirical results have not been backed up with a comprehensive theoretical analysis for general problems. In this letter we use a simple example to show that direct policy optimization is not directly generalizable to other seemingly simple problems. In such cases, direct optimization of a performance index can lead to unstable pole/zero cancellations, resulting in the loss of internal stability and unbounded outputs in response to arbitrarily small perturbations. We conclude this letter by analyzing several alternatives to avoid this phenomenon, suggesting some new directions in direct control policy optimization.**

***Index Terms*—Data-driven control, robust control, machine learning.**

## I. Introduction

RECENTLY, there has been renewed interest in "model free" control design techniques where the goal is to design a controller that optimizes performance based purely on experimental data. These techniques are attractive since they hold the promise of optimizing performance while avoiding a computationally expensive systems identification step [1]. Indeed, direct control policy optimization techniques have achieved remarkable success in a range of classical control problems, ranging from Linear Quadratic (LQR, LQG) to $\mathcal{H}_\infty$ and controller auto-tuning. While a general theory supporting these results is still emerging [2], recent results show that, in spite of lack of convexity, direct optimization can lead to optimal policies in these problems [3], [4], [5], [6], [7], [8], [9], [10] and provide bounds on the sample complexity [11], [12]. Hence, the hope is that these techniques can provide a viable alternative to the traditional systems identification-control design pipeline. The goal of this letter is to point out to the dangers of using direct optimization even in seemingly simple problems. As we illustrate with a simple first order system, direct policy optimization can lead to unstable pole zero cancellations and hence the loss of internal stability. In turn, this can result in unbounded signals in response to arbitrarily small perturbations to the control action.

This letter is organized as follows: Section II introduces the notation and required definitions; Section III contains the main result of this letter: a simple example where model agnostic performance optimization over all continuous stabilizing controllers leads to a pole/zero cancellation and loss of internal stability; Section IV connects this example with the empirical observation in [13] that adding noise during training increases robustness, and discusses some ideas to avoid the loss of internal stability. Section V offers some conclusions and points out to directions for further research.

Received 14 March 2025; revised 16 May 2025; accepted 10 June 2025. Date of publication 19 June 2025; date of current version 11 July 2025. This work was supported in part by NSF under Grant CNS–2038493 and Grant CMMI–2208182; in part by the Air Force Office of Scientific Research (AFOSR) under Grant FA9550-19-1-0005; in part by the Office of Naval Research (ONR) under Grant N00014-21-1-2431; and in part by the Sentry DHS Center of Excellence under Award 22STESE00001-03-03. Recommended by Senior Editor P. Tesi. *(Corresponding author: Mario Sznaier.)*

The authors are with the Robust Systems Lab, ECE Department, Northeastern University, Boston, MA 02115 USA (e-mail: m.sznaier@northeastern.edu; bozdag.m@northeastern.edu).

Digital Object Identifier 10.1109/LCSYS.2025.3581262

## II. Notation and Definitions

$\|u\| \doteq \sqrt{\boldsymbol{u}^T\boldsymbol{u}}$ denotes the usual Euclidean norm in $R^n$. For a given sequence $x_k$, $\|x\|_2 \doteq \sqrt{\sum \|x_k\|^2}$. $\ell^2$ denotes the Hilbert space of real vector sequences with finite $\|x\|_2$, equipped with the inner product $\langle x, y\rangle \doteq \sum x_i y_i$. $\ell^\infty$ denotes the Banach space of bounded real sequences equipped with the norm $\|x_k\|_\infty \doteq \sup_k |x_k|$. We will denote with capital letters the z-transform of sequences in $\ell^2$, e.g., $X(z) = \sum x_k z^{-k}$. By a slight abuse of notation, sometimes we will write $\|X(z)\|_2$ to denote the $\ell^2$ norm of the sequence $\{x_k\}$. Given a sequence $x_k$, we will denote by $(x_k)_\tau$ its truncation, that is, $(x_k)_\tau = x_k$ for $0 \le k \le \tau$ and $(x_k)_\tau = 0$ otherwise. In this context the extended space $\ell_e^\infty$ is defined as $\ell_e^\infty = \{u\colon (u)_\tau \in \ell^\infty \ \forall\ \tau \in [0, \infty)\}$. $\mathcal{RH}_\infty$ denotes the Lebesgue space of complex valued rational functions with bounded analytic continuation in $|z| > 1$, equipped with the norm $\|G(z)\|_{\mathcal{H}_\infty} \doteq \sup_{|\lambda|>1} |G(z)|$. In the sequel, we will represent a linear time-invariant system $\mathcal{G} : \ell_e^\infty \to \ell_e^\infty$ either by its convolution kernel $g$ or the transfer function $G(z)$. It is well known [14] that, if $\mathcal{G}$ is stable, then its $\ell^2$ induced norm $\|\mathcal{G}\|_{\ell^2\to\ell^2} = \|G(z)\|_{\mathcal{H}_\infty}$. Finally, given a matrix $\boldsymbol{M}$, $\|\boldsymbol{M}\|_2$ denotes its $\ell^2 \to \ell^2$ induced norm.

*Definition 1 [15]:* An operator $H : \ell_e^\infty \to \ell_e^\infty$ is finite $\ell^\infty$-gain stable if there exist constants $\gamma \ge 0, \beta \ge 0$ such that $\|(Hw)_\tau\|_\infty \le \gamma\|(w)_\tau\|_\infty + \beta$, $\forall w \in \ell_e^\infty$ and $\tau \in [0, \infty)$.

In the sequel, by a slight abuse of notation we will restrict this definition to the case where $\beta = 0$, that is, we will only consider mappings where $H0 = 0$. Thus, in the case of linear systems, finite-$\ell^\infty$ gain stability reduces to the standard bounded-input bounded-output stability.

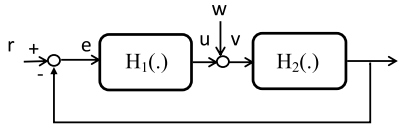



Fig. 1. The closed loop is internally stable if all four mappings $[rw]^T \to [ev]^T$ are stable.

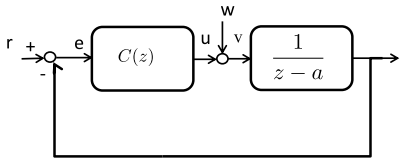



Fig. 2. Closed loop for the simple example.

*Definition 2:* The loop shown in Fig. 1 is finite $\ell^\infty$ gain internally stable if all four mappings $\begin{bmatrix} r & w \end{bmatrix}^T \to \begin{bmatrix} e & v \end{bmatrix}^T$ are finite $\ell^\infty$ gain stable.

## III. A Simple First Order Example

Here we present a simple example where input/output optimization leads to loss of internal stability: Given the open loop unstable system shown in Fig. 2, where $a > 1$, find a controller $C$ that minimizes $\|e\|_2$ to an input $r$ of the form:

$$r_k = \begin{cases} r_o, & 0 \le k \le n-1 \\ 0, & \text{otherwise} \end{cases} \Rightarrow R(z) = r_o \frac{z^n - 1}{(z-1)z^{n-1}} \quad (1)$$

where the amplitude $r_o$ and width $n$ are unknown. We consider three scenarios: (A) optimization over all internally stabilizing LTI controllers; (B) optimization over all LTI controllers that only stabilize the mapping $r \to e$; and (C) optimization over all time-invariant continuous nonlinear controllers that render the mapping $r \to e$ finite gain $\ell^\infty$-stable.

### A. Optimization Over All Internally Stabilizing Controllers

Consider first the case where the optimization is performed over all internally stabilizing LTI controllers, that is:

$$C = \underset{C \text{int. stab.}}{arg\,min} \left\| \frac{1}{1 + C(z)\frac{1}{z-a}} R(z) \right\|_2$$

Using the Youla parameterization [14] and expressing $S(z)$ in terms of the Youla parameter $Q(z)$ leads to the following (weighted) model matching problem[1]:

$$\min_{Q(z) \in \mathcal{RH}_\infty} \|R(z)M(z)(Y(z) + N(z)Q(z))\|_2 \quad (2)$$

where $N$ and $M$ are a coprime factorization of the plant $P = \frac{1}{z-a}$ and where $Y(z)$ is a solution of the following Bezout equation in $X(z), Y(z) \in \mathcal{RH}_\infty$:

$$N(z)X(z) + M(z)Y(z) = 1 \quad (3)$$

The problem above can be simplified by choosing a coprime factorization where $M(z)$ satisfies $|M(z)| = 1$ for all $|z| = 1$ and thus, for all signals $x \in \ell^2$, $\|M(z)X(z)\|_2 = \|X(z)\|_2$:

$$N = \frac{1}{az-1}, \; M = \frac{z-a}{az-1}, \; X = a^2 - 1, \; Y = a \quad (4)$$

[1]Please see the Appendix in [16].

leading to:

$$\min_{Q \in \mathcal{RH}_\infty} \|R(z)M(z)[Y(z) + N(z)Q(z)]\|_2 = \min_{\tilde{Q} \in \mathcal{RH}_\infty} \left\| aR(z) + \frac{R(z)}{z}\tilde{Q} \right\|_2 \quad (5)$$

where we used the fact that $M(z)$ is all pass and we defined

$$\tilde{Q}(z) \doteq Q(z)\frac{z}{az-1}$$

Problem (5) can be explicitly solved by considering an expansion of the optimal $\tilde{Q}$ of the form

$$\tilde{Q}(z) = q_o + \frac{q_1}{z} + \cdots \frac{q_{n-1}}{z^{n-1}} + \frac{\tilde{Q}(z)_{tail}}{z^n}$$

Parseval's Theorem combined with the explicit expression for $R(z)$ yields:

$$\left\| aR(z) + \frac{R(z)}{z}\tilde{Q} \right\|_2^2 = r_o^2 \left\| a + \frac{a+q_o}{z} + \frac{a+q_o+q_1}{z^2} \cdots + \frac{a+q_o+\cdots q_{n-2}}{z^{n-1}} + \frac{\sum q_i}{z^n} + \frac{\mathcal{O}(q_1, \cdots, q_{n-1}, \tilde{Q}_{tail})}{z^{n+1}} \right\|_2^2$$
$$= r_o^2 \left[ a^2 + (a+q_o)^2 + \cdots + (a+q_o+\cdots q_{n-1})^2 \right] + r_o^2 (\sum q_i)^2 + r_o^2 \left\| \mathcal{O}(q_1, \cdots, q_{n-1}, \tilde{Q}_{tail}) \right\|_2^2$$

Hence the optimal solution is given by $q_o = -a$, $q_i = 0$, $i \ge 1$, $\tilde{Q}_{tail} = 0$, with the corresponding $Q$, controller $C$, closed-loop sensitivity $S$, and optimal cost given by

$$Q = -\frac{a(az-1)}{z}, \quad C = \frac{(a^2+a-1)z - a^2}{a(z-1)}$$
$$S = \frac{a(z-a)(z-1)}{(az-1)z}, \quad S(z)R(z) = \frac{ar_o(z-a)(z^n-1)}{(az-1)z^n}$$
$$\|S(z)R(z)\|_2 = ar_o\sqrt{2} \quad (6)$$

Since the cost $ar_o\sqrt{2}$ in (6) is optimal, any controller yielding a lower cost **cannot be internally stabilizing.** Note that since $Q$ is stable and proper, the closed loop system must satisfy $S(\infty) = 1$ and $S(a) = 0$. These interpolation conditions follow from (5) and the fact that $M(a)N(a) = M(\infty)N(\infty) = 0$. Indeed, problem (2) can be recast as:

$$\min_{S \in \mathcal{RH}_\infty} \|S(z)R(z)\|_2 \quad \text{s.t. } S(\infty) = 1, \; S(a) = 0 \quad (7)$$

### B. Input/Output Optimization Over LTI Controllers

In this section we show that simply optimizing $\|S(z)R(z)\|_2$ without taking into account the interpolation constraint $S(a) = 0$ leads to controllers that are not internally stabilizing. Consider a controller in the form:

$$C_1(z) = K\frac{\prod_{i=1}^{n_z}(z - z_i)}{\prod_{i=1}^{n_p}(z - p_i)}. \quad (8)$$

Direct minimization of $\|S(z)R(z)\|_2$ with respect to $K, z_i, p_i$, with $n_p = n_z = 1$, using MATLAB's [17] command `fminsearch`[2] leads to

$$C_1(z) = \frac{(z-a)}{(z-1)} \quad (9)$$

[2]Interestingly, this model agnostic optimization yields a controller with a pole at $z = 1$, which is consistent with the internal model principle [18].

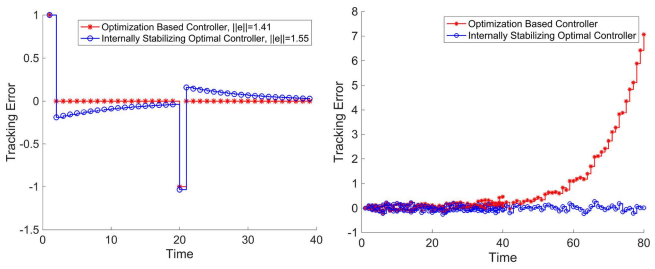



Fig. 3. Closed loop responses for the controllers (6) and (9): (a) Tracking error; (b) Response to a random perturbation $w$.

that yields the closed–loop mapping $r \to e$:

$$S = \frac{z-1}{z} \quad \text{with} \quad \|S(z)R(z)\|_2 = \left\| \frac{z-1}{z} \cdot \frac{r_o(z^n-1)}{(z-1)z^{n-1}} \right\|_2$$
$$= r_o \|1 - \frac{1}{z^n}\|_2 = r_o\sqrt{2} < a r_o \sqrt{2} \tag{10}$$

By parameterizing all controllers as $C = Q(z-a)/z(1-\frac{Q}{z})$ and dropping the no unstable pole/zero cancellation requirement, it can be shown that this controller with $Q = 1$ is globally optimal over the set of all LTI controllers that optimize $\|S(z)R(z)\|_2$ s.t. the input/output stability constraint.

A comparison of the closed loop response obtained using the controllers (6) and (9) is shown in Fig. 3. For the experiments, we use $r_o = 1$, $a = 1.1$. As expected, the controller (9) achieves a lower cost than (6). However, the resulting closed loop system $r \to e$ is not internally stable, due to the unstable pole/zero cancellation. Specifically, the closed-loop transfer functions from $w$ to $e$ and $u$ are:

$$T_{ew} = \frac{z-1}{z(z-a)}, \; T_{uw} = \frac{1}{z} \tag{11}$$

Therefore, any arbitrarily small perturbation to the control action will lead to an unbounded output (Fig. 3, (b)). This instability does not show in the performance index being optimized. Thus, any algorithm that seeks to optimize it with respect to the parameters of a controller of the form (8) with $n_z \geq 1, n_p \geq 1$ and achieves global optimality will lead to a controller that has to perform as well as (9), resulting in an input/output optimal controller that is not internally stabilizing. Further, since $T_{uw}$ is stable, the control action remains bounded, even if the output does not. Hence the loss of internal stability cannot be detected by adding noise to the signal $r$ during training and monitoring the magnitude of the control. Indeed, adding a disturbance $w \sim \mathcal{N}(0, 0.1)$ to $r$ during training still leads to the controller (9) and the unstable pole/zero cancellation.

The flaw in the input/output optimization discussed above is that it does not enforce the interpolation conditions. Indeed, while the sensitivity $S = \frac{z-1}{z}$ satisfies the condition $S(\infty) = 1$, it does not satisfy the interpolation condition $S(a) = 0$. Removing the interpolation constraints leads to a super-optimal controller that is not internally stabilizing due to the unstable pole/zero cancellation.

*Remark 1:* The loss of internal stability cannot be avoided by regularizing the performance index by adding a penalty in the control action. This penalty will avoid using integral action but still leads to an unstable pole/zero cancellation. For instance, changing the objective in the optimization above to $J = \|S(z)R(z)\|_2 + \|u\|_2$ leads, for the case $a = 1.1$, to

$$C(z) = \frac{0.60843(z-1.1)}{z-0.9935}$$

which still exhibits the unstable pole/zero cancellation.[3]

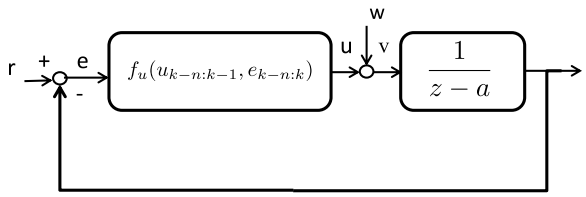



Fig. 4. A controller that minimizes $\frac{\|e\|_2}{|r_o|}$ cannot render the mapping $[r \quad w]^T \to [u \quad e]^T$ finite $\ell^\infty$ gain stable.

## C. General Optimization Based Nonlinear Controllers

In this section we show that any continuous nonlinear controller that (i) renders the input/output mapping $r \to s \doteq [u \quad e]^T$ finite-gain $\ell^\infty$ stable, and (ii) achieves tracking performance $\|e_k\|_2 = r_o\sqrt{2}$ for all $r_o$, will not internally stabilize the loop. Specifically, we will show that the resulting closed loop system shown in Fig. 4 cannot have a finite $\ell^\infty$ gain from $[r \quad w]^T \to [u \quad e]^T$.

*Theorem 1:* Consider a finite dimensional nonlinear controller of the form

$$u_k = f_u(\boldsymbol{\theta}_k) \tag{12}$$

where $\boldsymbol{\theta}_k \doteq \begin{bmatrix} \boldsymbol{u}_{k-1} \\ \boldsymbol{e}_k \end{bmatrix}$, $\boldsymbol{u}_{k-1} \doteq \begin{bmatrix} u_{k-1} \cdots u_{k-m} \end{bmatrix}^T$, $\boldsymbol{e}_k \doteq \begin{bmatrix} e_k \cdots e_{k-m} \end{bmatrix}$, $m$ is the memory of the controller, and $f_u$ is continuous. Let $\Phi_{cl}$ denote the corresponding closed loop mapping from the input $r$ to the output sequence $\{\begin{bmatrix} u_k \; e_k \end{bmatrix}^T\}$. If the controller (12) is such that:

(i) $\Phi_{cl}$ is finite $\ell^\infty$ gain stable, e.g., $\|\begin{bmatrix} u \; e \end{bmatrix}^T\|_\infty \leq K_r \|r\|_\infty$ and

(ii) when $r$ is a width-n pulse of the form (1), $\|e\|_2 = r_o\sqrt{2}$.

then the closed loop mapping $\begin{bmatrix} r \; w \end{bmatrix}^T \to \begin{bmatrix} u \; e \end{bmatrix}^T$ does not have finite $\ell^\infty$ gain.

*Proof:* Consider the controller (12) and note that finite closed-loop $\ell^\infty$ gain of $\Phi_{cl}$ implies that $f_u(\mathbf{0}) = 0$. Assume for now that $f_u$ is twice differentiable and consider its linearization around $\mathbf{0}$:

$$u_k = \frac{\partial f_u(\mathbf{0})}{\partial \boldsymbol{\theta}} \begin{bmatrix} \boldsymbol{u}_{k-1} \\ \boldsymbol{e}_k \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \boldsymbol{u}_{k-1}^T \boldsymbol{e}_k^T \end{bmatrix} \frac{\partial^2 f_u(\boldsymbol{\theta}_o)}{\partial \theta_i \partial \theta_j} \begin{bmatrix} \boldsymbol{u}_{k-1} \\ \boldsymbol{e}_k \end{bmatrix} \tag{13}$$

By contradiction, assume that the closed loop mapping $\begin{bmatrix} r \; w \end{bmatrix}^T \to \begin{bmatrix} u \; e \end{bmatrix}^T$ has finite $\ell^\infty$ gain $K_w$. We will show that under this assumption, the linear controller

$$u_k = \mathcal{L}(\boldsymbol{u}_{k-1}, \boldsymbol{e}_k) \doteq \frac{\partial f_u(\mathbf{0})}{\partial u} \boldsymbol{u}_{k-1} + \frac{\partial f_u(\mathbf{0})}{\partial e} \boldsymbol{e}_k \tag{14}$$

is internally stabilizing. By construction, the control sequence generated by the linear controller (14) is the same sequence

[3]Please see the Appendix in [16] for a comparison against LQG control.

generated by the nonlinear one in the presence of a fictitious disturbance

$$\hat{w}_k = -\frac{1}{2}\left[\boldsymbol{u}_{k-1}^T \boldsymbol{e}_k^T\right] \frac{\partial^2 f_u(\boldsymbol{\theta}_o)}{\partial\theta_i \partial\theta_j} \begin{bmatrix} \boldsymbol{u}_{k-1} \\ \boldsymbol{e}_k \end{bmatrix}$$

$$\text{with} \quad |\hat{w}_k| \leq 0.5 \left\| \frac{\partial^2 f_u(\boldsymbol{\theta}_o)}{\partial\theta_i \partial\theta_j} \right\|_2 \left\| \begin{bmatrix} \boldsymbol{u}_{k-1} \\ \boldsymbol{e}_k \end{bmatrix} \right\|^2$$

Hence

$$\|\hat{w}\|_\infty \leq C_1 \left\| \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{e} \end{bmatrix} \right\|_\infty^2 \leq C_1 K_w^2 \left\| \begin{bmatrix} r \\ w \end{bmatrix} \right\|_\infty^2 \tag{15}$$

for some constant $C_1$ that depends on $m$, the memory of the controller and the norm of the Hessian. Since the plant is linear, the error generated by the linear controller $\mathcal{L}$ satisfies

$$e_{\mathcal{L}} = e_{\mathcal{N}\mathcal{L}} + e_{\hat{w}} \tag{16}$$

where $e_{\hat{w}}$ denotes the error due effect of the fictitious perturbation, with

$$\|e_{\hat{w}}\|_\infty \leq K_w \left\| \begin{bmatrix} r \\ \hat{w} \end{bmatrix} \right\|_\infty \tag{17}$$

Thus

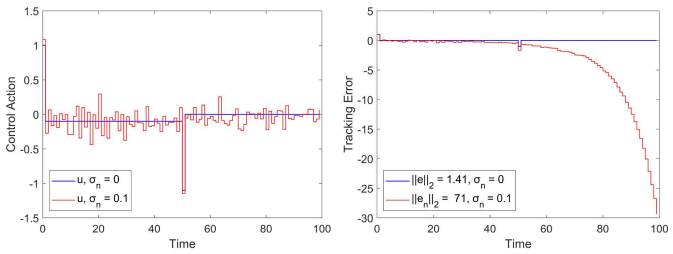
$$\begin{aligned} \left\| \begin{bmatrix} u_{\mathcal{L}} \\ e_{\mathcal{L}} \end{bmatrix} \right\|_\infty &\leq \left\| \begin{bmatrix} u_{\mathcal{N}\mathcal{L}} \\ e_{\mathcal{N}\mathcal{L}} \end{bmatrix} \right\|_\infty + \left\| \begin{bmatrix} \hat{w} \\ e_{\hat{w}} \end{bmatrix} \right\|_\infty \\ &\leq K_w \left\| \begin{bmatrix} r \\ w \end{bmatrix} \right\|_\infty + \left\| \begin{bmatrix} \hat{w} \\ e_{\hat{w}} \end{bmatrix} \right\|_\infty \end{aligned}$$

This inequality, combined with (15) and (17), shows that if $\begin{bmatrix} r \\ w \end{bmatrix} \in \ell^\infty$, then $\begin{bmatrix} u_{\mathcal{L}} \\ e_{\mathcal{L}} \end{bmatrix} \in \ell^\infty$. Since both the plant and the controller $\mathcal{L}$ are linear, this implies that the mapping $\begin{bmatrix} r \\ w \end{bmatrix} \to \begin{bmatrix} u_{\mathcal{L}} \\ e_{\mathcal{L}} \end{bmatrix}$ has all its poles in the open disk $|z| < 1$ (e.g., it is bounded input bounded output (BIBO) stable). To complete the proof, we need to show that the linear controller $\mathcal{L}$ achieves a tracking error $\|e\|_2 < ar_o\sqrt{2}$ to the input (1). Let $T_{ew}$ denote the closed loop mapping $w \to e$ achieved by the linear controller $\mathcal{L}$. Since the nonlinear controller achieves an error $\|e_{\mathcal{N}\mathcal{L}}\|_2 = r_o\sqrt{2}$ to the input (1), from the open-loop optimization in Section III-B, it follows that $u_{\mathcal{N}\mathcal{L}k} = 0$ and $e_k = 0\ \forall k \geq n{+}1$, which implies $\hat{w}_k = 0,\ k \geq n{+}2{+}m$. Hence $\|\hat{w}\|_2 \leq (n+2+m)\|\hat{w}\|_\infty$. From (15), (16) and assumptions (i) and (ii) we have:

$$\begin{aligned} \|e_{\mathcal{L}}\|_2 \leq \|e_{\mathcal{N}\mathcal{L}}\|_2 + \|T_{ew}\|_{\ell^2\to\ell^2}\|\hat{w}\|_2 \leq \|e_{\mathcal{N}\mathcal{L}}\|_2 + \\ \|T_{ew}\|_{\ell^2\to\ell^2} C_2 K_r^2 \|r\|_2^2 \leq r_o\sqrt{2} + \mathcal{O}(r_o^2) < ar_o\sqrt{2} \end{aligned} \tag{18}$$

if $r_o$ is small enough. It follows that the LTI controller (14) internally stabilizes the loop and achieves $\|e\|_2 < ar_o\sqrt{2}$ which contradicts (6).

Consider now the general case where $f_u(.)$ is continuous but not necessarily smooth. Since by assumption $f_u(.)$ renders the mapping $\begin{bmatrix} r \\ w \end{bmatrix} \to \begin{bmatrix} u \\ e \end{bmatrix}$ finite $\ell^\infty$ gain stable, then, as long as $r, w$ are confined to a compact set, so are $u, e$. From Stone-Weierstrass theorem [19] it follows that $f_u$ can be uniformly approximated arbitrarily close in this set by a polynomial $p_u(u, e)$, with $p_u(0) = f_u(0) = 0$. Thus, the effect of this approximation can be absorbed into $\hat{w}$, as another term with $|w_{ap}| \leq \epsilon$, in the proof that $\mathcal{L}$ is internally stabilizing. In terms of performance, since $p_u(0) = 0$ by construction, a same reasoning as before shows that, for the input (1), $w_{ap} = 0$ for $k > n + m + 2$. Hence, $\|w_{ap}\|_2 \leq (n + m + 2)\epsilon$ and the proof that the controller $\mathcal{L}$ (obtained now by linearizing the polynomial $p_u$) achieves $\|e\|_2 < ar_o\sqrt{2}$ still holds. ■




(a) (b)

Fig. 5. Illustration of Theorem 1. (a) Control action for the neural net controller, with and without noise added. (b) A small perturbation $w$ leads to an unbounded output.

To empirically validate our results, we use a simple neural network controller with the following architecture:

$$\begin{aligned} &\textbf{Input: } x \in \mathbb{R}^{3\times 1}, \ \textbf{Output: } y := z^{(2)}, \quad y \in \mathbb{R}, \\ &\textbf{FC1: } z^{(1)} := W^{(1)}x + b^{(1)}, \ W^{(1)} \in \mathbb{R}^{2\times 3}, \ b^{(1)} \in \mathbb{R}^{2\times 1} \\ &\textbf{Activation: } a^{(1)} := ReLU(z^{(1)}), \ a^{(1)} \in \mathbb{R}^{2\times 1} \\ &\textbf{FC2: } z^{(2)} := W^{(2)}a^{(1)} + b^{(2)}, \ W^{(2)} \in \mathbb{R}^{1\times 2}, \ b^{(2)} \in \mathbb{R} \end{aligned} \tag{19}$$

The input to the neural network is the vector $[u[k-1], e[k-1], e[k]]$, which includes the previous values for the control action and tracking error to fit the described mapping in (12). The bias terms $b^{(1)}$ and $b^{(2)}$ are set to $b^{(1)} = [0, 0, 0]^T$ and $b^{(2)} = 0$ due to the finite $\ell_2$ gain assumption. In this simple case, the optimal weights $W^{(1)} := [-0.6460, 0.7106, -0.6460; 0.4119, -0.4335, 3.1555]$, $W^{(2)} := [-1.5480, 0.3169]$ were found in 36 seconds using MATLAB's command `fminsearch` to minimize the tracking error to a pulse of the form (1) with both positive and negative amplitudes and different pulse-widths. In Figure 5, we observe the neural network achieving the optimal $\|e\|_2$ given in (12) as $r_o\sqrt{2}$. However, as we introduce a zero-mean random normal noise signal with $\sigma_n = 0.1$ to the control input $u$, as expected, the output becomes unbounded.

## IV. Possible Solutions to Avoid Losing Internal Stability and Their Limitations

As noted in Section III-B, the loss of internal stability can be traced to the existence of interpolation conditions. Direct unconstrained optimization of the tracking error leads to super-optimal controllers that violate these constraints. This issue can be solved by enforcing the interpolation constraints during the optimization. However, this can be difficult to accomplish in a model-agnostic setting. Below, we briefly discuss some options for enforcing these constraints.

*Adding noise while training:* Recall that internal stability is equivalent to input-output stability, provided that there are no unstable pole/zero cancellations between the plant and the controller [14]. Thus, perhaps the simplest way to implicitly avoid pole/zero cancellations is to add random noise to the control action during training, a technique that was empirically

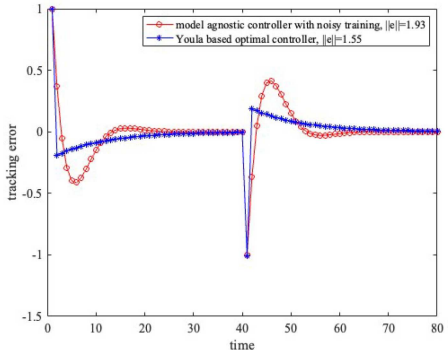



Fig. 6. Adding noise to the control action during training avoids the loss of internal stability at the price of 25% performance degradation.

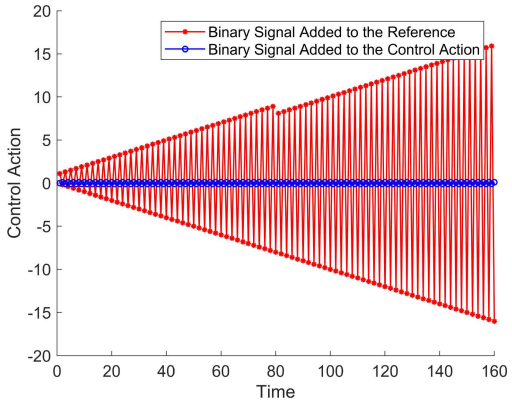



Fig. 7. Closed loop control response to perturbations in the reference (red) and the control (blue), for a model agnostic controller for the non-minimum phase plant $G = \frac{z+1}{z^2}$. Perturbing the reference signal leads to unbounded control.

shown to improve robustness in [13]. While this is easy to implement in a model agnostic framework, it has the drawback of leading to suboptimal performance. This is illustrated in Fig. 6, where adding a signal $w \sim \mathcal{N}(0, 10^{-4})$ leads to an internally stabilizing controller, albeit with a 25% performance degradation.

A further problem is that this approach critically hinges on adding noise at the right location. For instance, as illustrated in Section III-B, if the noise was instead added to the reference signal $r$, the resulting controller will still exhibit the unstable pole/zero cancellation. Similarly, if the plant is changed to (the non-minimum phase one) $G = \frac{z+1}{z^2}$, then the optimal model agnostic controller is given by $C = \frac{z^2}{(z+1)(z-1)}$. Fig. 7 shows the control action for a model agnostic controller, trained with noise added to the control action, in response to perturbations $w_1$ and $w_2$, both chosen as a binary signal with amplitude 0.1, added to the reference input $r$ and the control $u$, respectively. As shown there, in this case the control action in response to $w_1$, the perturbation of the reference, grows unbounded, even though the system was trained with noise added to the control. This highlights the importance of placing the perturbations at the "right" place when training, so that the interpolation constraints are implicitly enforced.

*Prestabilizing the Plant:* An alternative to adding noise is to use a two step process where a prestabilizing controller $C_{ps}$ is learned first and then a second controller is added to optimize

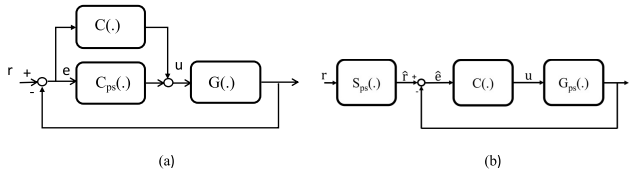



Fig. 8. Prestabilizing a minimum-phase plant leads to optimal model agnostic controllers. (a) Loop showing the overall control action. (b) Equivalent input/output mapping.

performance. In this case, the mapping $r \to e$ is given by (see Fig. 8)

$$T_{er} = \frac{1}{1 + G(C + C_{ps})} = S_{ps}\frac{1}{1 + G_{ps}C} \tag{20}$$

where

$$G_{ps} \doteq \frac{G}{1 + GC_{ps}} \text{ and } S_{ps} \doteq \frac{1}{1 + GC_{ps}} \tag{21}$$

The advantage of this approach is that, as long as the unknown plant is minimum phase, the interpolation constraints (and hence internal stability) are automatically satisfied if the controller $C$ achieves input/output stability of the prestabilized plant, that is $\frac{1}{1+CG_{ps}}$ is stable. This follows from [14, Lemma 2.3] and the fact that if $G$ is minimum phase so is $G_{ps}$. Since $G_{ps}$ is stable by construction, there cannot be unstable pole/zero cancellations between $C$ and $G_{ps}$.

Further, in this case, the controller $C$ can recover any performance achieved by a controller working directly with the original plant. Since by assumption the plant $G$ is minimum phase, it satisfies the so-called parity-interlacing property. Hence, it can be stabilized with an open-loop stable controller ([14], page 91). In turn, this implies that pre-stabilizing the plant does not affect achievable performance ([14], page 87).

For the simple example in this letter, one can search for a static stabilizing controller by simply minimizing the $\ell^2$ norm of the impulse response $\|1/(1+K/(z-a)\|_2$. Therefore, we use a 3 layer single-input, single output neural network with the input $e[k]$ to estimate such a controller. The resulting $C_{ps}$ has the weights $W^{(1)} := [0.4561; 1.5840; -0.7662]$, $W^{(2)} := [0.1.2235, 0.3421, -1.4357]$, with a ReLU activation in between and biases set to zero. Once a suitable $C_{ps}$ has been found, we then use the neural network in (19) as the $C$ in Fig. 8(b) and optimize the weights. The resulting controller weights (computed in 428 seconds) are $W^{(1)} := [0.5138, 0.3661, -0.6312; 2.3475, 0.5361, -0.8748]$, $W^{(2)} := [-0.7884, 0.5315]$. The tracking error $e$ achieved by the overall controller $(C + C_{ps})$ is shown in Fig. 9, both with and without noise added to the control action. As shown there, for the simple example in this letter, this approach indeed leads to an internally stabilizing controller that achieves near optimal performance.

Drawbacks of this approach include the need for having a pre-stabilizing controller $C_{ps}$, which could be non-trivial to find, and its limitation to minimum phase plants. In principle, non-minimum phase plants can be handled by adding noise to the control action, but this could entail performance loss. Further, if the plant is non-strongly stabilizable, the two-step approach may not be able to recover optimal performance.

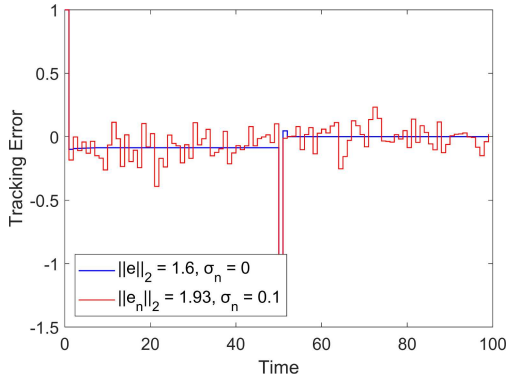



Fig. 9. Tracking error $e$ achieved by the overall NN controller $C + C_{ps}$ with and without noise added to the control action.

*Learning a Coprime Factorization of the Plant:* While not strictly model agnostic, this is a data-driven approach where a model of the plant is learned first and then used to design a controller. Consider first the case of (unknown) LTI plants. The main idea is (i) to learn a coprime factorization of the plant, $G = NM^{-1}$, where the factors $N, M$ satisfy

(3) for some stable $X, Y$, (ii) construct a prestabilizing controller $C_{ps} = X/Y$ (see [14]), and (iii) use the procedure outlined above. Alternatively, one could directly use the parameterization of all stabilizing controllers $C = \frac{X-MQ}{Y+NQ}$ and optimize over the parameter $Q$. A potential difficulty here is that, due to the use of finite, noisy data records, only approximations $\tilde{N}, \tilde{M}$ are learned. Thus, in principle there is no guarantee that the controller $C_{ps}$ will prestabilize the actual plant. As shown in a recent paper [20], this can be addressed by learning the factors $\tilde{N}, \tilde{M}$ using as a loss function the gap metric between $(\tilde{N}, \tilde{M})$ and $(N, M)$. The advantages of using this metric (vis-a-vis other metrics) is that, if the resulting gap is below a quantity that can be directly computed from the experimental data, then the controller $C_{sp}$ is guaranteed to stabilize the true plant. Further, the factors $\tilde{N}, \tilde{M}$ can be learned by solving a convex optimization problem. The main disadvantage of this approach is that, at present time, computing the gap requires knowledge of the frequency response of the plant. Hence, it cannot be directly applied to the case of interest in this letter where only finite time domain data is available. A time-domain characterization of the gap metric is needed to address this issue and to extend the approach to nonlinear plants.

## V. Conclusion

Model free direct policy optimization has the promise of optimizing performance while avoiding a computationally expensive system identification step. Further, it has been empirically shown to lead to optimal controllers in a number of cases of practical importance. However, as we show in this letter with a very simple example, the success that direct policy optimization has achieved in some classes of problems is not directly generalizable to other seemingly simple problems, where it can lead to closed-loop systems that are fragile to arbitrarily small perturbations. This effect can be traced to the fact that model agnostic optimization can lead to unstable pole/zero cancellations and hence loss of internal stability. With this in mind, we proposed several ways to overcome this difficulty, provided that some minimal a priori information about the unknown plant is available. Our results also point out to the need to develop a framework for learning stabilizing controllers from finite, time domain data records, using as loss function the gap metric. This metric is, at its core, a distance between closed-loop systems, as opposed to the open-loop metrics more commonly used when learning models from data.

## References

[1] M. Sznaier, "Control oriented learning in the era of big data," *IEEE Control Syst. Lett.*, vol. 5, no. 6, pp. 1855–1867, Dec. 2021.
[2] B. Hu, K. Zhang, N. Li, M. Mesbahi, M. Fazel, and T. Başar, "Toward a theoretical foundation of policy optimization for learning control policies," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 6, no. 1, pp. 123–158, 2023.
[3] M. Fazel, R. Ge, S. Kakade, and M. Mesbahi, "Global convergence of policy gradient methods for the linear quadratic regulator," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, 2018, pp. 1467–1476.
[4] D. Malik, A. Pananjady, K. Bhatia, K. Khamaru, P. Bartlett, and M. Wainwright, "Derivative-free methods for policy optimization: Guarantees for linear quadratic systems," in *Proc. 22nd Int. Conf. Artifi. Intell. Statist.*, vol. 89, Apr. 2019, pp. 2916–2925.
[5] I. Fatkhullin and B. Polyak, "Optimizing static linear feedback: Gradient method," 2020, *arXiv:2004.09875*.
[6] A. R. Kumar and P. J. Ramadge, "DiffLoop: Tuning PID controllers by differentiating through the feedback loop," in *Proc. 55th Annu. Conf. Inf. Sci. Syst. (CISS)*, 2021, pp. 1–6.
[7] M. Mehndiratta, E. Camci, and E. Kayacan, "Can deep models help a robot to tune its controller? A step closer to self-tuning model predictive controllers." *Electronics*, vol. 10, no. 18, p. 2187, 2021.
[8] A. Loquercio, A. Saviolo, and D. Scaramuzza, "AutoTune: Controller tuning for high-speed flight," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 4432–4439, Apr. 2022.
[9] S. Cheng, L. Song, M. Kim, S. Wang, and N. Hovakimyan, "DiffTune$^+$: Hyperparameter-free auto-tuning using auto-differentiation," in *Proc. Mach. Learn. Res.*, vol. 211, 2023, pp. 1–14.
[10] Y. Zheng, C. F. Pai, and Y. Tang, "Benign nonconvex landscapes in optimal and robust control, part I: Global optimality," 2023, *arXiv:2312.15332*.
[11] H. Mohammadi, A. Zare, M. Soltanolkotabi, and M. R. Jovanović, "Convergence and sample complexity of gradient methods for the model-free linear–quadratic regulator problem," *IEEE Trans. Autom. Control*, vol. 67, no. 5, pp. 2435–2450, May 2022.
[12] X. Guo, D. Keivan, G. Dullerud, P. Seiler, and B. Hu, "Complexity of derivative-free policy optimization for structured $\mathcal{H}_\infty$ control," in *Proc. 37th Conf. Neural Inf. Proc. Syst.*, 2023, pp. 1–29
[13] H. K. Venkataraman and P. J. Seiler, "Recovering robustness in model-free reinforcement learning," in *Proc. Amer. Control Conf. (ACC)*, 2019, pp. 4210–4216.
[14] R. Sánchez Peña and M. Sznaier, *Robust Systems Theory and Applications*. Hoboken, NJ. USA: Wiley, 1998.
[15] H. Khalil, *Nonlinear Systems*, 3rd ed. London, U.K.: Pearson, 2002.
[16] M. Sznaier and M. Bozdag, "Challenges in model agnostic controller learning for unstable systems," 2025, *arXiv:2505.11641*.
[17] *MATLAB Version: 23.2.0.2485118 (r2023b)*, MathWorks, Inc., Natick, MA, USA, 2023.
[18] B. Francis and W. Wonham, "The internal model principle of control theory," *Automatica*, vol. 12, no. 5, pp. 457–465, 1976.
[19] W. Cheney and W. Light, *A Course in Approximation Theory*. Providence, RI, USA: Amer. Math. Soc., 2009.
[20] R. Singh and M. Sznaier, "Certified control-oriented learning: A coprime factorization approach," in *Proc. IEEE 61st Conf. Decision Control (CDC)*, 2022, pp. 6012–6017.