Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

RESEARCH ARTICLE

Iterative learning control with discrete-time nonlinear non-minimum phase models via stable inversion

Isaac A. Spiegel*[1] | Nard Strijbosch[2] | Tom Oomen[2] | Kira Barton[1]

[1]Mechanical Engineering Department, University of Michigan Ann Arbor, Michigan, USA

[2]Mechanical Engineering Department, Eindhoven University of Technology, North Brabant, The Netherlands

Correspondence

*Isaac A. Spiegel, Mechanical Engineering Department, University of Michigan Ann Arbor. Email: ispiegel@umich.edu

Summary

Output reference tracking can be improved by iteratively learning from past data to inform the design of feedforward control inputs for subsequent tracking attempts. This process is called iterative learning control (ILC). This article develops a method to apply ILC to systems with nonlinear discrete-time dynamical models with unstable inverses (i.e. discrete-time nonlinear non-minimum phase models). This class of systems includes piezoactuators, electric power converters, and manipulators with flexible links, which may be found in nanopositioning stages, rolling mills, and robotic arms, respectively. As these devices may be required to execute fine transient reference tracking tasks repetitively in contexts such as manufacturing, they may benefit from ILC. Specifically, this article facilitates ILC of such systems by presenting a new ILC synthesis framework that allows combination of the principles of Newton's root finding algorithm with stable inversion, a technique for generating stable trajectories from unstable models. The new framework, called Invert-Linearize ILC (ILILC), is validated in simulation on a cart-and-pendulum system with model error, process noise, and measurement noise. Where preexisting Newton-based ILC diverges, ILILC with stable inversion converges, and does so in less than one third the number of trials necessary for the convergence of a gradient-descent-based ILC technique used as a benchmark.

KEYWORDS:
Iterative Learning Control, Stable Inversion, Non-minimum Phase, Newton's Method

1 | INTRODUCTION

Iterative learning control (ILC) is the process of learning an optimal feedforward control input over multiple trials of a repetitive process based on feedback measurements from previous trials. Compared to real-time-feedback and/or feedforward control techniques, many case studies of ILC have shown a substantial reduction in tracking error. Relevant applications include robot-assisted stroke rehabilitation[1], high speed train control[2], laser additive manufacturing[3], and vehicle-mounted manipulators[4], all of which use nonlinear models. In fact, while the majority of ILC literature focuses on linear systems, the prevalence of nonlinear dynamics in real-world systems has motivated the development of numerous ILC theories for discrete-time nonlinear models[5–10].

Abbreviations: ILC, iterative learning control; NILC, Newton iterative learning control; ILILC, invert-linearize iterative learning control

In addition to the state nonlinearities most commonly treated by nonlinear systems literature, many real-life systems exhibit dynamics well-represented by models with at least one of the following properties: (P1) relative degree ≥ 1, (P2) input nonlinearities, (P3) time-variation, and (P4) instability of the model inverse. For example, (P1) may be exhibited in the position control of myriad systems including piezoactuators[11], motors[12], robotic manipulators[13], and vehicles[14]. (P2) may be exhibited by piezoactuators[11], electric power converters[15], wind energy systems[16], magnetic levitation systems[17], and flexible-link manipulators[13]. (P3) may be exhibited by any feedforward-input-to-output model of systems using both feedforward and feedback control, as is often done for robotic manipulation[18]. Finally, and of primary concern in this work, (P4) may be exhibited by piezoactuators[19], electric power converters[15], wind energy systems[16], DC motor and tachometer assemblies[20], and flexible-link manipulators[13]. However, published discrete-time-nonlinear-model-based ILC theories exclude at least one of properties (P1)-(P4) from consideration. While the prior art makes important contributions such as foundational nonlinear ILC theory[5–7], relaxation of process repetitiveness assumptions[8], robustness to packet dropout in measurement and controller signals[9], and integration of ILC with adaptive control[10], these studies' analyses are limited to specific system structures. As a consequence, (P1) is not addressed by references 6, 7, 9, (P2) is not addressed by references 5–7, 9, 10, (P3) is not addressed by references 5, 8, and (P4) is not addressed by references 5–9[†].

The fact that many of the above example systems exhibit multiple properties and many of the above ILC theories exclude multiple properties from consideration illustrates that it can be challenging to find a model-based ILC synthesis scheme appropriate for many real-world applications. Indeed, flexible-link manipulators exhibit all four properties, and they are relevant to the fast and cost-effective automation of pick-and-place and assembly tasks as well as to the control of large structures such as cranes[21, ch. 6]. Such application spaces would benefit from having a versatile ILC scheme compatible with (P1)-(P4).

Additionally, while ILC seeks to converge to a satisfactorily low error, this learning is not immediate, and trials executed before the satisfactory error threshold is passed may be seen as costly failures from the perspective of the process specification. It is thus desirable to develop ILC schemes that converge as quickly as possible.

One ILC scheme that comes close to meeting these needs for versatility and speed is that of Avrachenkov[22], called Newton ILC (NILC) here. NILC is the application of Newton's root finding algorithm to a complete finite time series (as opposed to individual points in time). NILC's synthesis procedure and convergence analysis are unusually broad in that they admit discrete-time nonlinear models with properties (P1)-(P3)[22,23]. Additionally, Newton's method has been shown to deliver faster convergence in ILC than schemes such as P-type ILC[24, ch. 5], upon which much of the relevant prior art on the ILC of discrete-time nonlinear systems is founded[5–9]. However, this work demonstrates that when synthesized from models with unstable inverses, i.e. non-minimum phase models, NILC typically generates control signals that diverge towards very large magnitudes. In other words NILC may be incompatible with models exhibiting (P4). This article presents a new ILC framework inheriting the benefits of NILC while surmounting this shortcoming.

For linear models with unstable inverses, a common way to obtain feedforward control signals is to systematically synthesize approximate dynamical models with stable inverses by individually changing the model zeros and poles, e.g. the work of Tomizuka[25]. However, it is difficult to prescribe analogous systematic approximation methods for nonlinear models because the poles and zeros do not necessarily manifest as distinct binomial factors in the system transfer function that can be individually inverted or modified.

An alternative is to harness the fact that a scalar difference equation that is unstable when evolved forward in time from an initial condition is stable if evolved backwards in time from a terminal condition. If the stable and unstable modes of a system are decoupled and evolved in opposite directions, a stable total trajectory can be obtained. This process is called stable inversion. For linear systems on a bi-infinite timeline, with boundary conditions at time $\pm\infty$, stable inversion gives an exact solution to the output tracking problem posed by the unstable inverse model. In practice on a finite timeline, a high-fidelity approximation is obtained by ensuring the reference is designed with sufficient room for pre- and post-actuation, i.e. with a "flat" beginning and end. However, unlike ILC, stable inversion alone cannot account for model error. To address this, Zundert et al[26] details stable inversion and presents an ILC scheme for linear systems that incorporates a process similar to stable inversion.

Extension of stable inversion to nonlinear models involves additional complexities. Some of these challenges, e.g. the difficulty of completely decoupling the stable and unstable parts of a nonlinear system, have been addressed by works such as those of Devasia et al[27,28] for continuous-time systems and Zeng et al[29] for discrete-time systems. However, the following challenges remain. First, this prior art assumes that if the state and input are both zero at a particular time step, then the state will be zero at the next time step. This is not true for most representations of systems employing both feedback and feedforward control

[†]References 5–9 do not explicitly discuss inverse instability issues, but the appendix shows that the ILC schemes they present may fail to converge for systems with unstable inverses.

because if the reference is nonzero it drives state change via the feedback controller despite the initial state and feedforward input being zero. Stable inversion erroneously based on this assumption can have poor performance, and stable inversion has not been proven to converge when this assumption is relaxed. Secondly, Zeng et al[29] does not translate from the theoretical solution on a bi-infinite timeline to an implementable solution on a finite timeline. This work addresses these challenges.

In short, although the work to date on NILC and stable inversion has made great strides, gaps remain between the prior art and a synthesis scheme for ILC that is fast and applicable to a wide variety of models—including nonlinear non-minimum phase models. This leads to the main contribution of the present article: an ILC framework enabling controller synthesis from models satisfying all of (P1)-(P4). The key elements of this framework are

- reversing the order of the linearization and model inversion processes in NILC to circumvent issues associated with matrix inversion,
- reformulation of the model inversion in NILC as stable inversion,
- proof of stable inversion convergence with relaxed assumptions on state dynamics, enabling treatment of a wider array of feedback control and other time-varying models, and
- development of a structured method for implementing the stable inversion technique proposed in this work.

The proposed framework is validated in simulation on a nonlinear, relative degree 2, time-varying, non-minimum phase cart-and-pendulum system with model error and process and measurement noise.

The remainder of the paper is organized as follows. Section 2 provides technical details from the prior art in NILC[23] and stable inversion[29] necessary to present the novel contributions of the present work. Section 3 presents analysis that justifies the attribution of a class of NILC failures to inverse instability, and provides a new ILC framework that enables the circumvention of this failure mechanism by incorporating stable inversion. Section 4 provides proof of convergence of stable inversion for an expanded class of systems and provides improved methods for practical implementation. Section 5 details and discusses the validation of the new ILC framework with stable inversion through benchmark simulations on a non-minimum phase cart-and-pendulum system. This includes demonstration of conventional NILC's divergence when applied to the same system. Section 6 presents conclusions and areas for future work.

2 | BACKGROUND

2.1 | Newton ILC

Consider SISO, discrete-time, nonlinear, time-varying models

$$\hat{x}_\ell(k+1) = \hat{f}\left(\hat{x}_\ell(k), u_\ell(k), k\right) \qquad \hat{x}_\ell(0) = x_0 \quad \forall \ell \tag{1a}$$

$$\hat{y}_\ell(k) = \hat{h}(\hat{x}_\ell(k)) \tag{1b}$$

$$k \in \{0, 1, \dots, N\} \tag{2}$$

where $\hat{x} \in \mathbb{R}^{n_x}$ is the state vector, $u \in \mathbb{R}$ is the control input, $\hat{y} \in \mathbb{R}$ is the output, and k is the discrete time index. The system is made to perform repeated trials of a reference tracking task, where $N \in \mathbb{Z}_{>0}$ is the number of time steps in a trial (i.e. the number of samples minus 1), and $\ell \in \mathbb{Z}_{\geq 0}$ is the trial index[‡]. Additionally, consider the trial-invariant reference $r(k) \in \mathbb{R}$. Hats, $\hat{}$, are used to emphasize that (1) is an imperfect model of some true system. It is assumed that the control input and trial-invariant initial condition are perfectly known.

A classical ILC structure is given by

$$\mathbf{u}_{\ell+1} = \mathbf{u}_\ell + L_\ell \mathbf{e}_\ell \tag{3}$$

where $\mathbf{u} \in \mathbb{R}^{N-\mu+1}$ and $\mathbf{e} \in \mathbb{R}^{N-\mu+1}$ are input and error time series vectors, μ is the relative degree of (1), and $L \in \mathbb{R}^{N-\mu+1 \times N-\mu+1}$ is the learning matrix, which must be designed by a human or generated by an automatic synthesis procedure.

[‡]ℓ is used for the trial index because i and j will be used for matrix element indexing, k is used for the discrete time index, t is avoided to prevent confusion with continuous time, and ℓ is the next letter in the alphabet and thus commonly used for indexing.

The time series vectors, also called lifted vectors, are explicitly given by

$$\mathbf{u}_\ell = \left[u_\ell(0) \ \cdots \ u_\ell(N-\mu)\right]^T \tag{4}$$

$$\mathbf{e}_\ell = \mathbf{r} - \mathbf{y}_\ell = \left[r(\mu) - y_\ell(\mu) \ \cdots \ r(N) - y_\ell(N)\right]^T \tag{5}$$

where $y \in \mathbb{R}$ is the measured output of the true, but unknowable, system. These unknown system dynamics are represented as the function $\mathbf{g} : \mathbb{R}^{N-\mu+1} \to \mathbb{R}^{N-\mu+1}$, which takes in $\mathbf{u}_\ell$ and outputs $\mathbf{y}_\ell$.

The work of Avrachenkov[22] analyzes the convergence of (3) within a ball around the solution input $\mathbf{u}_\mathrm{d}$ ("solution" meaning that $\mathbf{g}\left(\mathbf{u}_\mathrm{d}\right) = \mathbf{r}$). In the present context this ball can be defined as $S\left(\mathbf{u}_\mathrm{d}, \rho\right) = \{\mathbf{u} \in \mathbb{R}^{N-\mu+1} | \left\|\mathbf{u} - \mathbf{u}_\mathrm{d}\right\|_2 < \rho\}$ with $\rho > 0$ and $\|\cdot\|_2$ being the Euclidean norm. Three conditions are posited:

(C1) The true dynamics $\mathbf{g}$ are continuously differentiable with respect to $\mathbf{u}$ in $S\left(\mathbf{u}_\mathrm{d}, \rho\right)$ and their Jacobian $\frac{\partial \mathbf{g}}{\partial \mathbf{u}}$ is Lipschitz continuous with respect to $\mathbf{u}$ in $S\left(\mathbf{u}_\mathrm{d}, \rho\right)$.

(C2) The learning matrix always has a bounded norm: $\left\|L_\ell\right\|_2 < \varepsilon_1 \in \mathbb{R}_{>0} \ \forall \ell$

(C3) The learning matrix is sufficiently similar to the inverse of the true lifted system Jacobian: $\left\|I - L_\ell \frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\mathbf{u}_\ell)\right\|_2 < 1 \ \forall \ell$

Avrachenkov proves[22] that if (C1)-(C3) are satisfied within $S\left(\mathbf{u}_\mathrm{d}, \rho\right)$, then there exists a ball $S(\mathbf{u}_\mathrm{d}, \varepsilon_2)$ with $\varepsilon_2 > 0$ such that if the initial guess $\mathbf{u}_0$ is an element of $S(\mathbf{u}_\mathrm{d}, \varepsilon_2)$ then (3) converges to $\mathbf{e} = 0_{N-\mu+1}$ as $\ell \to \infty$.

NILC is the use of the Newton-Raphson root finding algorithm to derive an automatic synthesis formula for the trial-varying learning matrix L_ℓ. The learning matrix is derived from the lifted representation of (1)-(2), $\hat{\mathbf{y}}_\ell = \hat{\mathbf{g}}(\mathbf{u}_\ell)$, which is defined as follows. Elements of $\hat{\mathbf{y}}_\ell$ output by $\hat{\mathbf{g}}$ are given via

$$\hat{y}_\ell(k) = \hat{h}\left(\hat{f}^{(k-1)}(\mathbf{u}_\ell)\right) \qquad k \in \{\mu, \mu+1, \cdots, N\} \tag{6}$$

where the parenthetical superscript notation indicates function composition of the form

$$\hat{f}^{(k)}(\mathbf{u}_\ell) = \hat{f}(\hat{x}_\ell(k), u_\ell(k), k) \tag{7}$$

$$= \hat{f}\left(\hat{f}\left(\cdots, u_\ell(k-1), k-1\right), u_\ell(k), k\right) \tag{8}$$

Because $\hat{x}_\ell(0) = x_0$ is known in advance and the time argument is determined by the element index of the lifted representation, $\hat{\mathbf{y}}_\ell$ is a function of only $\mathbf{u}_\ell$. Note that because the first element of $\hat{\mathbf{y}}_\ell$ is $\hat{y}_\ell(\mu)$ it explicitly depends on $u_\ell(0)$.

Using Newton's method to find the root of the error time series

$$\mathbf{e} = \mathbf{r} - \mathbf{y} = \mathbf{r} - \mathbf{g}(\mathbf{u}) \tag{9}$$

yields

$$L_\ell = \left(\frac{\partial \mathbf{g}}{\partial \mathbf{u}}(\mathbf{u}_\ell)\right)^{-1} \tag{10}$$

where $\frac{\partial \mathbf{g}}{\partial \mathbf{u}}$ is the Jacobian of $\mathbf{g}$ with respect to $\mathbf{u}$ as a function of $\mathbf{u}$. This learning matrix formula is impossible to evaluate because of its dependence on the unknown dynamics $\mathbf{g}$. Thus, $\hat{\mathbf{g}}$ is used as an approximation of $\mathbf{g}$ to yield the implementable NILC learning matrix formula

$$L_\ell = \left(\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}_\ell)\right)^{-1} \tag{11}$$

When NILC was originally developed, large Jacobians such as $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}$ were prohibitively difficult to derive and store as functions of $\mathbf{u}_\ell$, necessitating the definition of additional approximation techniques. However, with automatic differentiation tools such as CasADi[30], the barrier to Jacobian computation is vastly reduced, and can be done directly in many cases.

2.2 | Stable Inversion

The first step of stable inversion is deriving the conventional inverse. To synthesize a minimal inverse system representation, first assume (1) is in the normal form

$$\hat{x}^i(k+1) = \hat{x}^{i+1}(k) \qquad i < \mu \tag{12a}$$

$$\hat{x}^i(k+1) = \hat{f}^i\left(\hat{x}(k), u(k), k\right) \qquad i \geq \mu \tag{12b}$$

$$\hat{y}(k) = \hat{x}^1 \tag{12c}$$

where $\hat{x}(0) = 0$, and the superscripts i indicate the vector element index, starting from 1. Note the ILC trial index subscript ℓ is omitted in this section, as stable inversion on its own does not involve incrementing ℓ. Equation (12a) captures the time delay arising from the system relative degree, while equation (12b) captures the remaining system dynamics. One method of deriving this normal form from a system not in normal form is given in Eksteen et al[31]. Note that this coordinate transformation is performed in advance of any stable inversion or ILC analysis or synthesis. Thus the coordinate transform does not interfere with satisfaction of the identical initial condition assumption in (1a).

Given this normal form, use (12c) to replace the first μ state variables with output variables via

$$\hat{x}^i(k) = \hat{y}(k+i-1) \qquad i \leq \mu \tag{13}$$

Similarly, replace the μth state variable incremented by one time step (i.e. the left side of (12b) for $i = \mu$) with an output variable via

$$\hat{x}^\mu(k+1) = \hat{y}(k+\mu) \tag{14}$$

These substitutions are made to facilitate the inversion of system (12), as the inverse of a system with relative degree $\mu \geq 1$ is necessarily acausal with dependence on some subset of $\{\hat{y}(k), \hat{y}(k+1), \cdots, \hat{y}(k+\mu)\}$ at each time step k. For notational compactness, define the $\hat{y}$-preview vector $\hat{\mathcal{Y}}(k) \equiv [\hat{y}(k), \cdots, \hat{y}(k+\mu)]^T$. Then inverting (12b) with $i = \mu$ yields the conventional inverse output function

$$u(k) = \hat{f}^{\mu^{-1}}\left(\left[\hat{x}^{\mu+1}, \ \cdots, \ \hat{x}^{n_x}\right]^T, \hat{\mathcal{Y}}(k), k\right) \tag{15}$$

where $\hat{f}^{\mu^{-1}}$ is the inverse of $\hat{f}^\mu$, i.e. (12b, $i = \mu$) solved for $u(k)$. This output equation is substituted into (12b, $i > \mu$) along with (13)-(14) to yield the entire inverse system dynamics

$$\hat{\eta}(k+1) = \hat{f}_\eta\left(\hat{\eta}(k), \hat{\mathcal{Y}}(k), k\right) \tag{16a}$$

$$u(k) = \hat{f}^{\mu^{-1}}\left(\hat{\eta}(k), \hat{\mathcal{Y}}(k), k\right) \tag{16b}$$

where $\hat{\eta} \in \mathbb{R}^{n_\eta}$ ($n_\eta = n_x - \mu$) is the inverse state vector defined

$$\hat{\eta}^i(k) \equiv \hat{x}^{\mu+i}(k) \tag{17}$$

and $\hat{f}_\eta : \mathbb{R}^{n_\eta} \times \mathbb{R}^{\mu+1} \times \mathbb{Z} \to \mathbb{R}^{n_\eta}$ is the inverse state dynamics

$$\hat{f}^i_\eta(\hat{\eta}(k), \hat{\mathcal{Y}}(k), k) \equiv \hat{f}^{i+\mu}(\hat{x}(k), u(k), k) \tag{18}$$

Next, a similarity transform is to be applied to this inverse system to decouple the stable and unstable modes of its linearization about the initial condition. Consider the Jacobian

$$A = \frac{\partial \hat{f}_\eta}{\partial \hat{\eta}}\left(\hat{\eta} = 0, \hat{\mathcal{Y}} = \hat{\mathcal{Y}}^\dagger, k = 0\right) \tag{19}$$

where $\hat{\mathcal{Y}}^\dagger$ is the solution to $\hat{f}_\eta(0, \hat{\mathcal{Y}}^\dagger, 0) = 0$. Then let V be the similarity transform matrix such that

$$\tilde{A} = V^{-1}AV = \begin{bmatrix} \tilde{A}_s & 0 \\ 0 & \tilde{A}_u \end{bmatrix} \tag{20}$$

where $\tilde{A}_s \in \mathbb{R}^{\upsilon\times\upsilon}$ has all eigenvalues inside the unit circle, and $\tilde{A}_u \in \mathbb{R}^{n_\eta-\upsilon\times n_\eta-\upsilon}$ has all eigenvalues outside the unit circle. This can be satisfied by deriving the real block Jordan form of A. The corresponding inverse system state dynamics are

$$\tilde{\eta}(k+1) = \tilde{f}_\eta\left(\tilde{\eta}(k), \hat{\mathcal{Y}}(k), k\right) \equiv V^{-1}\hat{f}_\eta\left(V\tilde{\eta}(k), \hat{\mathcal{Y}}(k), k\right) \tag{21}$$

where the tilde on $\tilde{f}_\eta$ indicates application to $\tilde{\eta}$ rather than $\hat{\eta}$. Note that despite using a linearization-derived linear similarity transform, (21) describes the same nonlinear time-varying dynamics as (16a), but with the linear parts of the stable and unstable modes decoupled.

If (1) has an unstable inverse, then (21) is unstable and $\tilde{\eta}(k)$ will be unbounded as k increases. However, given an infinite timeline in the positive and negative direction, the equation

$$\tilde{\eta}(k) = \sum_{i=-\infty}^{\infty} \phi(k-i)\left(\tilde{f}_\eta\left(\tilde{\eta}(i-1), \hat{\mathcal{Y}}(i-1), i-1\right) - \tilde{A}\tilde{\eta}(i-1)\right) \tag{22}$$

where

$$\phi(k) = \begin{cases} \begin{bmatrix} \tilde{A}_s^k & 0_{v\times n_\eta - v} \\ 0_{n_\eta - v\times v} & 0_{n_\eta - v\times n_\eta - v} \end{bmatrix} & k > 0 \\ \begin{bmatrix} I_{v\times v} & 0_{v\times n_\eta - v} \\ 0_{n_\eta - v\times v} & 0_{n_\eta - v\times n_\eta - v} \end{bmatrix} & k = 0 \\ \begin{bmatrix} 0_{v\times v} & 0_{v\times n_\eta - v} \\ 0_{n_\eta - v\times v} & -\tilde{A}_u^k \end{bmatrix} & k < 0 \end{cases} \tag{23}$$

is an exact, bounded solution to (21) provided the right hand side of (22) exists for all $k \in \mathbb{Z}$. However, (22) is implicit, and thus cannot be directly evaluated. A fixed-point problem solver—Zeng et al[29] uses Picard iteration—must be used to find $\tilde{\eta}$, and sufficient conditions for the solver convergence and solution uniqueness must be determined.

The Picard iterative solver[32, ch. 9] for (22) is

$$\tilde{\eta}_{(m+1)}(k) = \sum_{i=-\infty}^{\infty} \phi(k-i)\left(\tilde{f}_\eta\left(\tilde{\eta}_{(m)}(i-1), \hat{y}(i-1), i-1\right) - \tilde{A}\tilde{\eta}_{(m)}(i-1)\right) \tag{24}$$

where the parenthetical subscript $(m) \in \mathbb{Z}_{\geq 0}$ is the Picard iteration index.

To prove that (24) converges to a unique solution,[29] makes the assumptions that

(Z1) $\hat{f}(0,0,k) = 0 \;\forall\; k$, and

(Z2) $\tilde{\eta}_{(0)}(k) = 0 \;\forall\; k$.

Note that the continuous time literature also makes these assumptions[27,28].

The first assumption is violated for many representations of systems incorporating both feedback and feedforward control. An example of such a system is given in Section 5, where u is the feedforward control input and the feedback control is part of the time-varying dynamics of $\hat{f}$. This feedback control influences $\hat{x}$ regardless of whether or not $u(k) = 0$. While there may often be a change of variables that enables satisfaction of (Z1), (12) already imposes constraints on the states and outputs, and for many systems it is unlikely for there to exist a change of variables satisfying both assumptions.

Furthermore, while for systems satisfying (Z1), (Z2) may be the zero-input state trajectory, this is untrue for systems violating (Z1). For these systems, the zero state trajectory (Z2) is essentially arbitrary, and may degrade the quality of low-*m* Picard iterates if far from the solution trajectory. This jeopardizes convergence because the computational complexity of the Picard iteration solution grows exponentially with the number of iterations. It is thus desirable to reach a satisfactory solution in as few iterations as possible, i.e. it is desirable to have high-quality low-*m* iterates.

Section 4 addresses these limitations by proving a new set of sufficient conditions for the unique convergence of (24) that relaxes (Z1), (Z2).

3 | ILC ANALYSIS AND DEVELOPMENT

In order to develop a new ILC framework for non-minimum phase models, it is necessary to concretely identify the failure mechanism of Section 2.1's NILC. Such analysis is absent in the literature, and is thus provided in Section 3.1. Section 3.2 then presents a new learning matrix formula overcoming this failing.

3.1 | Failure for Models with Unstable Inverses

The NILC scheme (3), (11) provides convergence of $\mathbf{e}_\ell$ to 0 in theory. However, this assumes perfect computation of the matrix inversion in (11). In practice, the precision to which $\left(\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}_\ell)\right)^{-1}$ can be accurately computed is directly dependent on the condition number of $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}_\ell)$. If the condition number of a matrix is large enough, the values computed for its inverse may become arbitrary, and their order of magnitude may grow directly with the order of magnitude of the condition number[33, 34, ch. 3.2]. This "blowing up" of the matrix inverse can cause divergence of (3), (11).

In studies unrelated to NILC, large $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}$ condition numbers have been observed for non-minimum phase linear systems, both time-invariant[35, 36, ch. 5.3-5.4] and time-varying[37, 38, ch. 4.1.1]. The fact that the minimum singular value of $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}_\ell)$ decreases with increases in the system frequency response function magnitude at the Nyquist frequency[39] may contribute to this ill-conditioning. For linear systems, this magnitude is directly dependent on the zero magnitudes, and thus on the inverse systems' stability.

If inverse instability degrades the conditioning of $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}$ for linear models, it is guaranteed to do so for nonlinear models. This is because the Jacobian evaluated at a particular input trajectory, $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}^*)$, is equal to the constant matrix $\frac{\partial \bar{\mathbf{g}}}{\partial \mathbf{u}}$ where $\bar{\mathbf{g}}$ is the lifted input-output model of the linearization of (1) about the trajectory $\mathbf{u}^*$.

To illustrate this equality, first consider that the elements of $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}^*)$ are given by (6) and the chain rule as

$$\frac{\partial \hat{y}(k)}{\partial u(j)}(\mathbf{u}^*) = \frac{\partial \hat{h}}{\partial \hat{x}}\left(\hat{f}^{(k-1)}(\mathbf{u}^*)\right)\frac{\partial \hat{f}^{(k-1)}}{\partial \mathbf{u}}(\mathbf{u}^*)\frac{\partial \mathbf{u}}{\partial u(j)} \tag{25}$$

where

$$\frac{\partial \mathbf{u}}{\partial u(j)} = \begin{bmatrix} 0_{1\times j} & 1 & 0_{1\times N-\mu+1-j}\end{bmatrix}^T \tag{26}$$

and $\frac{\partial \hat{h}}{\partial \hat{x}}$ is a row vector.

Then consider the linearization of (1) about $\mathbf{u}^*$:

$$\delta\hat{x}(k+1) = \bar{f}\left(\delta\hat{x}(k), \delta u(k), k\right) = \frac{\partial \hat{f}}{\partial \hat{x}}\left(\hat{x}^*(k), u^*(k), k\right)\,\delta\hat{x}(k) + \frac{\partial \hat{f}}{\partial u}(\hat{x}^*(k), u^*(k), k)\,\delta u(k) \tag{27a}$$

$$\delta\hat{y}(k) = \bar{h}(\delta\hat{x}(k)) = \frac{\partial \hat{h}}{\partial \hat{x}}(\hat{x}^*(k))\,\delta\hat{x}(k) \tag{27b}$$

where $\hat{x}^*(k) = \hat{f}^{(k-1)}(\mathbf{u}^*)$ and the δ notation denotes $\delta\hat{x}(k) = \hat{x}(k) - \hat{x}^*(k)$ for $\hat{x}$ and similar for u.

Lifting (27) in the same manner as (1) yields the output perturbation as a function of the input perturbation time series $\delta\mathbf{u}$ via

$$\delta\hat{y}(k) = \frac{\partial \hat{h}}{\partial \hat{x}}\left(\hat{f}^{(k-1)}(\mathbf{u}^*)\right)\bar{f}^{(k-1)}(\delta\mathbf{u}) \tag{28}$$

Because of (27)'s linearity, $\bar{f}^{(k-1)}(\delta\mathbf{u})$ can be explicitly expanded as

$$\bar{f}^{(k-1)}(\delta\mathbf{u}) = \left(\prod_{\kappa=0}^{k-1}\frac{\partial \hat{f}^{(\kappa)}}{\partial \hat{f}^{(\kappa-1)}}(\mathbf{u}^*)\right)\delta\hat{x}(0) + \frac{\partial \hat{f}^{(k-1)}}{\partial \mathbf{u}}(\mathbf{u}^*)\delta\mathbf{u} \tag{29}$$

where $\prod$ is ordered with the factor of least κ on the right and the factor of greatest κ on the left. The terminal condition of the recursive function composition is $\hat{f}^{(-1)} = \hat{x}(0)$. From (28) and (29) it is clear that the elements of $\frac{\partial \bar{\mathbf{g}}}{\partial \delta\mathbf{u}}$ are given by

$$\frac{\partial \delta\hat{y}(k)}{\partial \delta u(j)} = \frac{\partial \hat{h}}{\partial \hat{x}}\left(\hat{f}^{(k-1)}(\mathbf{u}^*)\right)\frac{\partial \hat{f}^{(k-1)}}{\partial \mathbf{u}}(\mathbf{u}^*)\frac{\partial \delta\mathbf{u}}{\partial \delta u(j)}, \tag{30}$$

which is equal to (25) because $\frac{\partial \delta\mathbf{u}}{\partial \delta u(j)} = \frac{\partial \mathbf{u}}{\partial u(j)}$ due to the identical structures (4) of $\mathbf{u}$ and $\delta\mathbf{u}$ with respect to u and δu time indexing. Thus, if (1) is such that its linearization (27) is unstable, $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}_\ell)$ will suffer ill-conditioning and attempts to compute the learning matrix (11) may yield a matrix with large arbitrary elements. Such a learning gain matrix may in turn cause $\mathbf{u}_{\ell+1}$ to contain large arbitrary elements, causing the learning law to diverge.

Therefore, for the learning law (3) to converge for a system with an unstable inverse in practice, a learning matrix synthesis that does not require matrix inversion of $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}_\ell)$ is desired.

3.2 | Alternative Learning Matrix Synthesis

To circumvent issues associated with inverting $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}_\ell)$ a new learning matrix definition seeking to satisfy the requirements (C2)-(C3) in the spirit of Newton's method, but without the matrix inversion requirement of (11), is given by

$$L_\ell = \frac{\partial \hat{\mathbf{g}}^{-1}}{\partial \hat{\mathbf{y}}}(\mathbf{y}_\ell) \tag{31}$$

where $\hat{\mathbf{g}}^{-1} : \mathbb{R}^{N-\mu+1} \to \mathbb{R}^{N-\mu+1}$ is a lifted model of the inverse of (1). This makes $\frac{\partial \hat{\mathbf{g}}^{-1}}{\partial \hat{\mathbf{y}}}$ a function of the output of (1), namely $\hat{\mathbf{y}}_\ell$. As stated in Section 2.1, $\hat{\mathbf{y}}_\ell$ is the output of a necessarily erroneous model, and thus is merely a prediction of the accessible, measured output $\mathbf{y}_\ell$. Hence $\mathbf{y}_\ell$ is used as the input to $\frac{\partial \hat{\mathbf{g}}^{-1}}{\partial \hat{\mathbf{y}}}$. In short, this work proposes using the linearization of the inverse of (1) rather than the inverse of the linearization, and thus the new framework (3), (31) will be referred to as "Invert-Linearize ILC" (ILILC).

The first step in deriving $\hat{\mathbf{g}}^{-1}$, and thus in deriving (31) is the inversion of the original model (1) . A direct method of inverting (1) is to solve

$$\hat{y}_\ell(k+\mu) = \hat{h}(\hat{f}^{(k+\mu-1)}(\mathbf{u}_\ell)) \tag{32}$$

for $u_\ell(k)$, and substitute the resulting function of $\left\{\hat{y}_\ell(k),\ \hat{y}_\ell(k+1), \cdots, \hat{y}_\ell(k+\mu)\right\}$ into (1a). However, if (1) has an unstable inverse, this method of inversion will yield unbounded states $\hat{x}_\ell(k)$ as k increases. Thus, $\hat{\mathbf{g}}^{-1}$ is derived via the stable inversion procedure described in Section 4 rather than direct inversion. Note, though, that (31) also admits the use of other stable approximate inverse models for $\hat{\mathbf{g}}^{-1}$ should they be available.

4 | STABLE INVERSION DEVELOPMENT

This section proves a relaxed set of sufficient conditions for the convergence of Picard iteration to the unique solution to the stable inversion problem, i.e. the unique solution to (22) from Section 2.2. This enables stable inversion—and thus ILC—for a new class of system representations capturing simultaneous feedback and feedforward control. Additionally, a new initial Picard iterate prescription is given to suit the broadened scope of stable inversion, and a procedure for practical implementation is described. This procedure enables the derivation of $\hat{\mathbf{g}}^{-1}$.

4.1 | Fixed-Point Problem Solution

Several definitions are needed to prove the relaxed set of sufficient conditions for convergence of the fixed-point problem solver used for stable inversion.

Definition 1 (Lifted Matrices and Third-Order Tensors)**.** Given the vector and matrix functions of time $a(k) \in \mathbb{R}^n$ and $B(k) \in \mathbb{R}^{n\times n}$, the corresponding lifted matrix and third order tensor are given by upright bold notation: $\mathbf{a} \in \mathbb{R}^{n\times\mathcal{K}}$ and $\mathbf{B} \in \mathbb{R}^{n\times n\times\mathcal{K}}$. $\mathcal{K}$ is the time dimension, and may be ∞. Elements of the lifted objects are $\mathbf{a}^{i,k} \equiv a^i(k)$ and $\mathbf{B}^{i,j,k} \equiv B^{i,j}(k)$.

Definition 2 (Matrix and Third-Order Tensor Norms)**.** $\|\cdot\|_\infty$ refers to the ordinary ∞-norm when applied to vectors, and is the matrix norm induced by the vector norm when applied to matrices (i.e. the maximum absolute row sum). Additionally, the entry-wise $(\infty, 1)$-norm is defined for the matrices and third-order tensors $\mathbf{a}$ and $\mathbf{B}$ from Definition 1 as

$$\|\mathbf{a}\|_{\infty,1} \equiv \sum_{k\in\mathcal{K}} \|a(k)\|_\infty \qquad \|\mathbf{B}\|_{\infty,1} \equiv \sum_{k\in\mathcal{K}} \|B(k)\|_\infty \tag{33}$$

Definition 3 (Local Approximate Linearity[27,29])**.** $\tilde{f}_\eta$ is locally approximately linear in $\tilde{\eta}(k)$ and its $\tilde{\eta}(k) = 0$ dynamics, in a closed s-neighborhood around $(\tilde{\eta}(k) = 0, \tilde{f}_\eta(0, \hat{\mathscr{Y}}(k), k) = 0)$, with Lipschitz constants $K_1, K_2 > 0$ if $\exists s > 0$ such that for any vectors

- $a(k), b(k) \in \mathbb{R}^{n_\eta}$ with $\|\cdot\|_\infty \le s\ \forall k$, and
- $\mathscr{a}(k), \mathscr{b}(k) \in \mathbb{R}^{\mu+1}$ such that $\left\|\tilde{f}_\eta(0, \mathscr{a}(k), k)\right\|_\infty, \left\|\tilde{f}_\eta(0, \mathscr{b}(k), k)\right\|_\infty \le s\ \forall k$

the following is true $\forall k$

$$\begin{aligned}&\left\|\left(\tilde{f}_\eta(a(k), \mathscr{a}(k), k) - Aa(k)\right) - \left(\tilde{f}_\eta(b(k), \mathscr{b}(k), k) - Ab(k)\right)\right\|_\infty \\ &\qquad\qquad \le K_1 \|a(k) - b(k)\|_\infty + K_2 \left\|\tilde{f}_\eta(0, \mathscr{a}(k), k) - \tilde{f}_\eta(0, \mathscr{b}(k), k)\right\|_\infty\end{aligned} \tag{34}$$

With these definitions a new set of sufficient conditions for Picard iteration convergence is established. Proof of this theorem shares the approach of Zeng et al[29] in establishing the Cauchy nature of the Picard sequence. It is also influenced by the proofs of Picard iterate local approximate linearity for continuous-time systems in Devasia et al[27].

Theorem 1. The Picard iteration (24) converges to a unique solution to (21) if the following sufficient conditions are met.

(C4) $\left\|\tilde{\boldsymbol{\eta}}_{(0)}\right\|_{\infty,1} \le s$

(C5) $\forall k\ \exists \hat{\mathscr{Y}}(k) = \hat{\mathscr{Y}}^\dagger(k)$ such that $\tilde{f}_\eta(0, \hat{\mathscr{Y}}^\dagger(k), k) = 0$

(C6) $\tilde{f}_\eta$ is locally approximately linear in the sense of (34)

(C7) $K_1 \|\boldsymbol{\varphi}\|_{\infty,1} < 1$

(C8) $\frac{\|\boldsymbol{\varphi}\|_{\infty,1} K_2 \left\|\tilde{\mathbf{f}}_\eta(0,\hat{\mathscr{y}})\right\|_{\infty,1}}{1-\|\boldsymbol{\varphi}\|_{\infty,1} K_1} \leq s$

where $\boldsymbol{\varphi}$ and $\left\|\tilde{\mathbf{f}}_\eta(0,\hat{\mathscr{y}})\right\|_{\infty,1} = \sum_{k=-\infty}^{\infty} \left\|\tilde{f}_\eta(0,\hat{\mathscr{y}}(k),k)\right\|_\infty$ are defined by Definition 1.

Proof: Proof that (24) converges to a unique fixed point begins with an induction showing that $\tilde{\eta}_{(m)}(k)$ remains in the locally approximately linear neighborhood $\forall\ k, m$. The base case of this induction is given by (C4). Then under the premise

$$\left\|\tilde{\boldsymbol{\eta}}_{(m)}\right\|_{\infty,1} \leq s \tag{35}$$

the induction proceeds as follows. Here, ellipses indicate the continuation of a line of mathematics.

By the Picard iterative solver (24):

$$\left\|\tilde{\boldsymbol{\eta}}_{(m+1)}\right\|_{\infty,1} = \sum_{k=-\infty}^{\infty} \left\| \sum_{i=-\infty}^{\infty} \phi(k-i)\left(\tilde{f}_\eta\left(\tilde{\eta}_{(m)}(i-1),\hat{\mathscr{y}}(i-1),i-1\right) - A\tilde{\eta}_{(m)}(i-1)\right)\right\|_\infty \cdots \tag{36}$$

By the triangle inequality:

$$\cdots \leq \sum_{k=-\infty}^{\infty}\sum_{i=-\infty}^{\infty} \left\|\phi(k-i)\left(\tilde{f}_\eta\left(\tilde{\eta}_{(m)}(i-1),\hat{\mathscr{y}}(i-1),i-1\right) - A\tilde{\eta}_{(m)}(i-1)\right)\right\|_\infty \cdots \tag{37}$$

By the fact that for matrix norms induced by vector norms $\|Ba\| \leq \|B\|\,\|a\|$ for matrix B and vector a:

$$\cdots \leq \sum_{k=-\infty}^{\infty}\sum_{i=-\infty}^{\infty} \|\phi(k-i)\|_\infty \left\|\tilde{f}_\eta\left(\tilde{\eta}_{(m)}(i-1),\hat{\mathscr{y}}(i-1),i-1\right) - A\tilde{\eta}_{(m)}(i-1)\right\|_\infty \cdots \tag{38}$$

$$\cdots = \sum_{i=-\infty}^{\infty} \left\|\tilde{f}_\eta\left(\tilde{\eta}_{(m)}(i-1),\hat{\mathscr{y}}(i-1),i-1\right) - A\tilde{\eta}_{(m)}(i-1)\right\|_\infty \sum_{k=-\infty}^{\infty} \|\phi(k-i)\|_\infty \cdots \tag{39}$$

By the fact that $\sum_{k=-\infty}^{\infty} \|\phi(k-i)\|_\infty$ has the same value $\forall i$

$$\cdots = \|\boldsymbol{\varphi}\|_{\infty,1} \sum_{i=-\infty}^{\infty} \left\|\tilde{f}_\eta\left(\tilde{\eta}_{(m)}(i-1),\hat{\mathscr{y}}(i-1),i-1\right) - A\tilde{\eta}_{(m)}(i-1)\right\|_\infty \cdots \tag{40}$$

By (C5):

$$\cdots = \|\boldsymbol{\varphi}\|_{\infty,1} \sum_{i=-\infty}^{\infty} \left\|\left(\tilde{f}_\eta\left(\tilde{\eta}_{(m)}(i-1),\hat{\mathscr{y}}(i-1),i-1\right) - A\tilde{\eta}_{(m)}(i-1)\right) - \left(\tilde{f}_\eta\left(0,\hat{\mathscr{y}}^\dagger(i-1),i-1\right) - A(0)\right)\right\|_\infty \cdots \tag{41}$$

By (C6):

$$\cdots \leq \|\boldsymbol{\varphi}\|_{\infty,1} \sum_{i=-\infty}^{\infty} K_1 \left\|\tilde{\eta}_{(m)}(i-1)\right\|_\infty + K_2 \left\|\tilde{f}_\eta\left(0,\hat{\mathscr{y}}(i-1),i-1\right)\right\|_\infty \cdots \tag{42}$$

$$\cdots = \|\boldsymbol{\varphi}\|_{\infty,1}\left(K_1 \left\|\tilde{\boldsymbol{\eta}}_{(m)}\right\|_{\infty,1} + K_2 \left\|\tilde{\mathbf{f}}_\eta(0,\hat{\mathscr{y}})\right\|_{\infty,1}\right) \cdots \tag{43}$$

By (C7), the denominator of (C8) is positive. Thus both sides of (C8) can be multiplied by this denominator without changing the inequality direction. Thus by (35) and algebraic rearranging of (C8)

$$\cdots \leq \|\boldsymbol{\varphi}\|_{\infty,1}\left(K_1 s + K_2 \left\|\tilde{\mathbf{f}}_\eta(0,\hat{\mathscr{y}})\right\|_{\infty,1}\right) \leq s \tag{44}$$

$\therefore \left\|\tilde{\boldsymbol{\eta}}_{(m)}\right\|_{\infty,1} \leq s\ \forall m$. Because $\left\|\tilde{\boldsymbol{\eta}}_{(m)}\right\|_{\infty,1} \geq \left\|\tilde{\eta}_{(m)}(k)\right\|_\infty\ \forall k$, this implies that $\tilde{\eta}_{(m)}(k)$ is within the locally approximately linear neighborhood $\forall\ m, k$.

To show that (24) converges to a unique fixed point, define

$$\Delta\tilde{\eta}_{(m)}(k) \equiv \tilde{\eta}_{(m+1)}(k) - \tilde{\eta}_{(m)}(k) \tag{45}$$

Then, by a nearly identical induction

$$\left\|\Delta\tilde{\boldsymbol{\eta}}_{(m+1)}\right\|_{\infty,1} \leq \|\boldsymbol{\varphi}\|_{\infty,1} K_1 \left\|\Delta\tilde{\boldsymbol{\eta}}_{(m)}\right\|_{\infty,1} \tag{46}$$

By (C7)

$$\lim_{m\to\infty} \left\| \Delta\tilde{\boldsymbol{\eta}}_{(m)} \right\|_{\infty,1} = 0 \tag{47}$$

which implies

$$\lim_{m\to\infty} \left\| \Delta\tilde{\eta}_{(m)}(k) \right\|_{\infty} = 0 \ \forall k \tag{48}$$

$\therefore \forall k$ the sequence $\{\tilde{\eta}_m(k)\}$ is a Cauchy sequence, and thus the fixed point $\tilde{\eta}(k) = \lim_{m\to\infty} \tilde{\eta}_{(m)}(k)$ is unique. ■

Remark 1. Neither the preceding presentation nor the nonlinear stable inversion prior art[29] explicitly discusses the intuitive foundation of stable inversion: evolving the stable modes of an inverse system forwards in time from an initial condition and evolving the unstable modes backwards in time from a terminal condition. Unlike for linear time invariant (LTI) systems, this intuition is not put into practice directly for nonlinear systems because the similarity transforms that completely decouple the stable and unstable modes of linear systems do not necessarily decouple the stable and unstable modes of nonlinear systems. However, the same principle underpins this work. This is evidenced by the fact that the intuitive LTI stable inversion is recovered from (22) when $\hat{f}$ is LTI, as illustrated briefly below.

For LTI $\hat{f}$, $\tilde{f}$ takes the form

$$\tilde{\eta}(k+1) = \tilde{A}\tilde{\eta}(k) + \tilde{B}\hat{\mathcal{Y}}(k) \tag{49}$$

$$\begin{bmatrix} \tilde{\eta}_s(k+1) \\ \tilde{\eta}_u(k+1) \end{bmatrix} = \begin{bmatrix} \tilde{A}_s & 0 \\ 0 & \tilde{A}_u \end{bmatrix} \begin{bmatrix} \tilde{\eta}_s(k) \\ \tilde{\eta}_u(k) \end{bmatrix} + \begin{bmatrix} \tilde{B}_s \\ \tilde{B}_u \end{bmatrix} \hat{\mathcal{Y}}(k) \tag{50}$$

Then the implicit solution (22) becomes the explicit solution

$$\tilde{\eta}(k) = \sum_{i=-\infty}^{\infty} \phi(k-i)\tilde{B}\hat{\mathcal{Y}}(i-1) \tag{51}$$

$$\begin{bmatrix} \tilde{\eta}_s(k) \\ \tilde{\eta}_u(k) \end{bmatrix} = \begin{bmatrix} \sum_{i=-\infty}^{k} \tilde{A}_s^{k-i}\tilde{B}_s\hat{\mathcal{Y}}(i-1) \\ -\sum_{i=k+1}^{\infty} \tilde{A}_u^{k-i}\tilde{B}_u\hat{\mathcal{Y}}(i-1) \end{bmatrix} = \begin{bmatrix} \tilde{A}_s\tilde{\eta}_s(k-1) + \tilde{B}_s\hat{\mathcal{Y}}(k-1) \\ \tilde{A}_u^{-1}\tilde{\eta}_u(k+1) - \tilde{A}_u^{-1}\tilde{B}_u\hat{\mathcal{Y}}(k) \end{bmatrix} \tag{52}$$

which is the forward evolution of the stable modes and backward evolution of the unstable modes where the initial and terminal conditions at $k = \pm\infty$ are zero.

4.2 | Initial Picard Iterate $\tilde{\eta}_{(0)}$ Selection and Implementation

This subsection addresses the need to select a new initial Picard iterate $\tilde{\eta}_{(0)}(k)$ in the absence of (Z2). Also addressed is the fact that (24) is a purely theoretical, rather than implementable, solution because it contains infinite sums along an infinite timeline.

In the context of ILC, the learned feedforward control action is often intended to be a relatively minor adjustment to the primary action of the feedback controller. Thus, choosing $\tilde{\eta}_{(0)}(k)$ to be the feedback-only trajectory, i.e. the zero-feedforward-input trajectory, is akin to warm-starting the fixed-point solving process. This trajectory is given by

$$\begin{aligned} \hat{x}(k+1) &= \hat{f}\left(\hat{x}(k), 0, k\right) \qquad \hat{x}(0) = 0_{n_x} \\ \tilde{\eta}_{(0)}(k) &= V^{-1} \begin{bmatrix} 0_{\mu\times\mu} & 0_{\mu\times n_\eta} \\ 0_{n_\eta\times\mu} & I_{n_\eta\times n_\eta} \end{bmatrix} \hat{x}(k) \end{aligned} \tag{53}$$

for $k \in \{0, \cdots, N-\mu\}$.

An implementable version of (24) is given by

$$\tilde{\eta}_{(m+1)}(k) = \sum_{i=1}^{N-\mu+1} \phi(k-i) \left(\tilde{f}_\eta \left(\tilde{\eta}_{(m)}(i-1), \hat{\mathcal{Y}}(i-1), i-1 \right) - \tilde{A}\tilde{\eta}_{(m)}(i-1) \right) \tag{54}$$

for $k \in \{1, ..., N-\mu\}$, fixing the initial condition $\tilde{\eta}_{(m)}(0) = 0_{n_\eta}$ $\forall m$.

Note that (54) is equivalent to assuming $\tilde{\eta}_{(m)}(k) = 0$, $\hat{\mathcal{Y}}(k) = 0$, and $\tilde{f}_\eta(0, 0, k) = 0$ for $k \in (-\infty, -1] \cup [N-\mu+1, \infty)$ and extracting the $k \in [1, N-\mu]$ elements of $\tilde{\eta}_{(m+1)}(k)$ generated by (24). These assumptions correspond to a lack of control action prior to $k = 0$ and a reference trajectory that brings the system back to its zero initial condition with enough trailing zeros for the system to settle by $k = N-\mu$. This is typical of repetitive motion processes, but admittedly may preclude some other ILC applications.

Furthermore, for the first Picard iteration ($m+1 = 1$) these assumptions yield identical (54)-generated and (24)-generated $\tilde{\eta}_{(1)}(k)$ on $k \in [0, N-\mu]$. Because output tracking of systems with unstable inverses typically requires preactuation, for this range of k to contain a practical control input trajectory there must be sufficient leading zeros in the reference starting at $k = 0$. For the following Picard iterates the theoretical and implementable trajectories are unlikely to be equal, but can be made closer the more leading zeros are included in the reference.

Ultimately, applying (54) for any number of iterations $m_{\text{final}} \geq 1$ yields an expression for each time step of $\tilde{\eta}_{(m_{\text{final}})}(k)$ whose only variable parameters are the elements of $\hat{\mathbf{y}}$. This is because the recursion calling $\tilde{\eta}_{(m_{\text{final}})}(k)$ terminates at the known trajectory $\tilde{\eta}_{(0)}(k)$, and because $\hat{y}(k) = 0$ for $k \in \{0, ..., \mu - 1\}$ due to the known initial condition $\hat{x}(0) = 0$. The concatenation of these expressions plugged into the inverse output function (16b) yields the lifted inverse system model

$$\hat{\mathbf{g}}^{-1}(\hat{\mathbf{y}}) = \begin{bmatrix} \hat{f}^{\mu^{-1}}\left(V\tilde{\eta}_{(m_{\text{final}})}(0), \hat{\mathcal{Y}}(0), 0\right) \\ \hat{f}^{\mu^{-1}}\left(V\tilde{\eta}_{(m_{\text{final}})}(1), \hat{\mathcal{Y}}(1), 1\right) \\ \vdots \\ \hat{f}^{\mu^{-1}}\left(V\tilde{\eta}_{(m_{\text{final}})}(N-\mu), \hat{\mathcal{Y}}(N-\mu), N-\mu\right) \end{bmatrix} \tag{55}$$

which enables the synthesis of the ILILC learning matrix (31). With this, the complete synthesis of ILILC with stable inversion—starting from a model in the normal form (12)—can be summarized by Procedure 1.

Procedure 1 ILILC Synthesis with Stable Inversion

```
1: Derive the minimal state space representation $\hat{f}_\eta$ and $\hat{f}^{\mu^{-1}}$ (from (16)) of the conventional inverse of (12).
2: Apply similarity transform $V$ (from (20)) to derive the inverse state dynamics representation $\tilde{f}_\eta$ (from (21)) with decoupled stable and unstable linear parts.
3: Use the fixed-point problem solver (53)-(54) to derive the inverse system state $\tilde{\eta}_{(m_{\text{final}})}$ as a function of $\hat{\mathbf{y}}$ at each point in time $k \in \{0, \cdots, N-\mu\}$.
4: Derive the lifted inverse model $\hat{\mathbf{g}}^{-1}$ via (55).
5: Use an automatic differentiation tool to derive $\frac{\partial \hat{\mathbf{g}}^{-1}}{\partial \hat{\mathbf{y}}}$ as a function of $\mathbf{y}$, i.e. the learning matrix $L_\ell$ from (31).
6: Compute $L_\ell = \frac{\partial \hat{\mathbf{g}}^{-1}}{\partial \hat{\mathbf{y}}}(\mathbf{y}_\ell)$ at each trial for the ILC law (3).
   // Steps 3-4 are greatly facilitated by using a computer algebra system. CasADi can provide this functionality in addition to automatic differentiation.
```

Remark 2. The computation time required to synthesize ILILC with stable inversion grows with the number of time steps N in the time series, and can become relatively long. However, the overwhelming majority of this computation is performed before the execution of the zeroth trial and need not be repeated. This allows for minimal computation time—i.e. minimal downtime—between trials.

More specifically, Steps 1-5 are all performed before trial zero execution, with Step 5 being the most computationally intensive. These steps yield a function $\frac{\partial \hat{\mathbf{g}}^{-1}}{\partial \hat{\mathbf{y}}}(\cdot)$ that arithmetically produces a learning matrix L_ℓ given an output time series. Step 6—the only step featuring intertrial computation—merely needs to call this function and the simple matrix-vector multiplication of (3). The fixed-point problem solving and automatic differentiation does not need to be redone.

For reference, the validation system's computation times for each step of Procedure 1 are given in Section 5.4, Table 2.

5 | VALIDATION

This section presents validation of the fundamental claim that the original NILC fails for models with unstable inverses and that the newly proposed ILILC framework—when used with stable inversion—succeeds. Additionally, while the intent of ILC is to account for model error, overly erroneous modeling can cause violation of (C3), which may cause divergence of the ILC law. Thus this section also probes the performance and robustness of ILILC with stable inversion over increasing model error in physically motivated simulations.

The ILILC law (3), (31) is applied as a reference shaping tool to a feedback control system (sometimes called "series ILC"). This represents the common scenario of applying a higher level controller to "closed source" equipment. The resultant system (1) is a nonlinear time-varying system with relative degree $\mu = 2$.

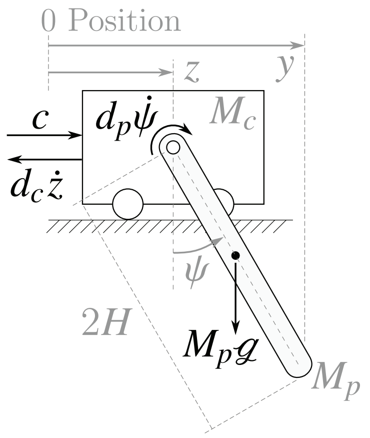

FIGURE 1 Cart and pendulum system. Dimension, position, and mass annotations are in grey. Force and torque annotations are in black.

Modeling error is simulated by synthesizing the ILC laws from a nominal "control model" of the example system, and applying the resultant control inputs to a set of "truth models" featuring random parameter errors and the injection of process and measurement noise. Finally, to give context to the results for ILILC with stable inversion, identical simulations are run with a benchmark technique that does not require modification for models with unstable inverses: gradient ILC.

5.1 | Benchmark Technique: Gradient ILC

Gradient ILC is gradient descent applied to the optimization problem

$$\arg\min_{\mathbf{u}} \frac{1}{2}\mathbf{e}^T\mathbf{e} \tag{56}$$

which yields the ILC law

$$\mathbf{u}_{\ell+1} = \mathbf{u}_\ell + \gamma \frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}} \left(\mathbf{u}_\ell\right)^T \mathbf{e}_j \tag{57}$$

where $\gamma > 0$ is the gradient descent step size. Note that (57) is free of the matrix inversion that inhibits the application of NILC to systems with unstable inverses. Past work on gradient ILC[40] has been limited to linear systems due in part to the difficulty of synthesizing $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}$ for nonlinear systems. This article extends gradient ILC to nonlinear systems by using the automatic differentiation tool CasADi to synthesize $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}$.

The tuning parameter γ influences the performance-robustness trade off of (57). Reducing γ improves the probability that (57) will converge for some unknown model error, but may also reduce the rate of convergence. For the sake of comparing the convergence rates between gradient ILC and ILILC, here we choose γ such that the two methods have comparable probabilities of convergence over the set of random model errors tested: $\gamma = 1.1$.

5.2 | Example System

Consider the system pictured in Figure 1, consisting of a pendulum fixed to the mass center of a cart on a rail. This subsection presents the first-principles continuous-time equations of motion for this plant, the method for converting these dynamics to the discrete-time normal form (12), and the control architecture of the system.

The cart is subjected to an applied force c, and viscous damping occurs both between the cart and the rail and between the pendulum and the cart. Equations of motion for this plant are given by

$$\ddot{\psi} = -3\Big(HM_p\left(c+\omega_c\right)\cos(\psi) + d_p(M_c+M_p)\dot{\psi} + H^2M_p^2\sin(\psi)\cos(\psi)\dot{\psi}^2 + gH\left(M_cM_p + M_p^2\right)\sin(\psi) \\ - d_cHM_p\cos(\psi)\dot{z}\Big)\frac{1}{H^2M_p\left(4(M_c+M_p) - 3M_p\cos^2(\psi)\right)} \tag{58}$$

$$\ddot{z} = \left(4H\left(c+\omega_c\right) + 3d_p\cos(\psi)\dot{\psi} + 4H^2M_p\sin(\psi)\dot{\psi}^2 + 3gHM_p\sin(\psi)\cos(\psi) - 4d_cH\dot{z}\right)\frac{1}{H\left(4(M_c+M_p) - 3M_p\cos^2(\psi)\right)} \quad (59)$$

where $\psi(k)$ is the pendulum angle, $z(k)$ is the cart's horizontal position, $g = 9.8\,\frac{\mathrm{m}}{\mathrm{s}^2}$ is gravitational acceleration, and the process noise $\omega_c(k)$ is a random sample from a normal distribution with 0 mean and standard deviation 3.15×10^{-2} N. H is the pendulum half-length, M_c and M_p are the cart and pendulum masses, and d_c and d_p are the cart-rail and pendulum-cart damping coefficients, respectively. The time argument of ω_c, ψ, z and their derivatives has been dropped for compactness.

The output to be tracked is the pendulum tip's horizontal position, y. Obtaining a discrete-time state space model of this system in the normal form (12) requires first a change of coordinates such that the desired output is a state, and then discretization. The change of coordinates is

$$\psi = \arcsin\left(\frac{y-z}{2H}\right) \quad (60)$$

with associated derivative substitutions

$$\dot{\psi} = \frac{\dot{y}-\dot{z}}{2H\sqrt{1-\frac{(y-z)^2}{4H^2}}} \quad (61)$$

$$\ddot{\psi} = \frac{\sec(\psi)\left(\ddot{y}-\ddot{z}+2H\sin(\psi)\dot{\psi}^2\right)}{2H} \quad (62)$$

Then the equations of motion are solved for in terms of the new coordinates. In the present case (58)-(62) can be solved for $\ddot{y}(k)$ and $\ddot{z}(k)$ as functions of $y(k)$, $z(k)$, $\dot{y}(k)$, and $\dot{z}(k)$. Next, forward Euler discretization is applied recursively to the equations of motion to reformulate the state dynamics in terms of discrete time increments rather than derivatives, as is required by the normal form. The innermost layer of the recursion is the first derivatives

$$\dot{y}(k) = \frac{y(k+1)-y(k)}{T_s} \qquad \dot{z}(k) = \frac{z(k+1)-z(k)}{T_s} \quad (63)$$

where the sample period $T_s = 0.016$ s in this case. These can be plugged into $\ddot{y}(k)$ and $\ddot{z}(k)$ to eliminate their dependence on derivatives. The next—and in this case final—layer is the forward Euler discretization of the second derivatives. The outermost layer can be rearranged to yield the discrete-time equations of motion

$$\begin{aligned} y(k+2) &= \ddot{y}(k)T_s^2 + 2y(k+1) - y(k) \\ z(k+2) &= \ddot{z}(k)T_s^2 + 2z(k+1) - z(k), \end{aligned} \quad (64)$$

which are directly used to define the state dynamics f in terms of the state vector $x(k) = [y(k),\ y(k+1),\ z(k),\ z(k+1)]^T$. The explicit expressions of (64) are too long to print here, but can be easily obtained in Mathematica, MATLAB symbolic toolbox, etc. via the algebra described in (60)-(64).

The output must track the reference $r(k)$ given in Figure 2. To accomplish this the plant is equipped with a full-state feedback controller modeled as

$$c(k) = \kappa_0 r^*(k) - \begin{bmatrix}\kappa_1 & \kappa_2 & \kappa_3 & \kappa_4\end{bmatrix} x(k) \quad (65)$$

$$r^*(k) = r(k) + u(k) \quad (66)$$

Here, $r^*(k)$ is the effective reference and $u(k)$ is the control input generated by the ILC law. In other words, the ILC law adjusts the reference delivered to the feedback controller to eliminate the error transients inherent to feedback control. Finally, the error signal input to the ILC law is subject to measurement noise $\omega_y(k)$

$$e(k) = r(k) - y(k) - \omega_y(k) \quad (67)$$

where the noise's distribution has 0 mean and standard deviation 5×10^{-5} m.

The ILC law itself is synthesized from a control model that is identical in structure to the truth model presented above, but has $\hat{\omega}_c = \hat{\omega}_y = 0$ and uses the model parameters tabulated in Table 1. Stable inversion for the synthesis of learning matrix (31) is performed with a single Picard iteration, i.e. $m_{\text{final}} = 1$ in (55). To simulate model error, the hatless truth model parameters differ from the behatted control model parameters in a manner detailed in Section 5.3. This ultimately results in the system block diagram given in Figure 3.

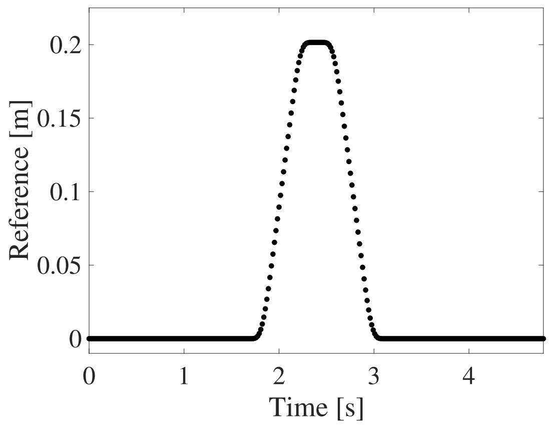

FIGURE 2 Reference

TABLE 1 Cart-Pendulum Control Model Parameters

Parameter	Symbol	Value
Cart Mass	$\hat{M}_c$	0.5 kg
Pendulum Mass	$\hat{M}_p$	0.25 kg
Pendulum Half-Length	$\hat{H}$	0.225 m
Cart-Rail Damping Coefficient	$\hat{d}_c$	$10\ \frac{\text{kg}}{\text{s}}$
Pendulum-Cart Damping Coefficient	$\hat{d}_p$	$0.01\ \frac{\text{kg m}^2}{\text{s}}$
Full State Feedback Gain 0	$\hat{\kappa}_0$	630
Full State Feedback Gain 1	$\hat{\kappa}_1$	−5900
Full State Feedback Gain 2	$\hat{\kappa}_2$	5900
Full State Feedback Gain 3	$\hat{\kappa}_3$	−3700
Full State Feedback Gain 4	$\hat{\kappa}_4$	4300

5.3 | Simulation and Analysis Methods

Let $\hat{\theta} \in \mathbb{R}^{10}$ be a vector of the control model parameters in Table 1. Then a truth model can be specified by the vector θ, generated via

$$\theta = \left(1_{10\times 1} e_\theta^T \odot I + I\right) \hat{\theta} \tag{68}$$

where $\odot$ is the Hadamard product and $e_\theta \in \mathbb{R}^{10}$ is a random sample of a uniform distribution. Under (68), each element of e_θ is the relative error between the corresponding elements of θ and $\hat{\theta}$. Thus, $\left\|e_\theta\right\|_2$ provides a scalar metric for the model error between the control model and a given truth model. The range $\left\|e_\theta\right\|_2 \in [0, 0.1]$ is divided into 20 bins of equal width, and 50 truth models are generated for each bin. Both ILC schemes are applied to each truth model with 50 trials, and $u_0(k) = 0\ \forall k$. A full set of 50 trials of one of the ILC laws applied to a single truth model is referred to as a "simulation." The results of these simulations are used to characterize the probability of convergence and rate of convergence of each ILC law.

For each iteration of a simulation, the normalized root mean square error (NRMSE) is given by

$$\text{NRMSE}_\ell \equiv \frac{\text{RMS}\left(\mathbf{e}_\ell\right)}{\|\mathbf{r}\|_\infty} \tag{69}$$

A simulation is deemed convergent if there exists ℓ^* such that NRMSE_ℓ is less than some tolerance for all $\ell \geq \ell^*$. This work uses a tolerance of 5×10^{-4}, which is close to the NRMSE floor created by noise.

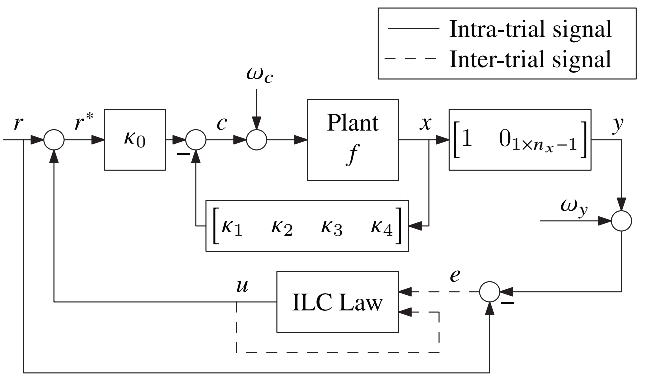
Intra-trial signal
Inter-trial signal
ω_c
r
r^*
κ_0
c
Plant f
x
$\begin{bmatrix}1 & 0_{1\times n_x-1}\end{bmatrix}$
y
$\begin{bmatrix}\kappa_1 & \kappa_2 & \kappa_3 & \kappa_4\end{bmatrix}$
ω_y
u
ILC Law
e

FIGURE 3 System Block Diagram. The control law outputting u is synthesized from the control models defined by the behatted parameters of Table 1 and by $\omega_c(k) = \omega_y(k) = 0$. The plant and controller gain blocks are defined with the truth model parameters generated according to Section 5.3. Inter-trial signals from trial ℓ are stored and used to compute the input for trial $\ell + 1$.

Let $\ell^{\beta,\tau,\lambda}$ be the minimum ℓ^* for truth model $\tau \in [1, 50]$ in bin $\beta \in [1, 20]$ under ILC law $\lambda \in \{$ ILILC , gradient ILC$\}$, and let C be the set of all (β, τ) for which both ILILC and gradient ILC converge. Then the mean transient convergence rate

$$\mathcal{R}_\lambda = \underset{C,\ell\in[1,\ell^{\beta,\tau,\lambda}]}{\text{mean}} \left(\frac{\text{NRMSE}_\ell^{\beta,\tau,\lambda}}{\text{NRMSE}_{\ell-1}^{\beta,\tau,\lambda}} \right) \tag{70}$$

offers a numerical performance metric. Note that Avrachenkov[22] gives a theoretical convergence analysis for the ILC structure (3) in general (covering NILC, ILILC, and gradient ILC). This analysis can be used to lower bound performance (i.e. upper bound convergence rate) via multiple parameters computed from the learning matrix L_ℓ and the true dynamics $\mathbf{g}$. The mean transient convergence rate (70) may thus serve as a specific, measurable counterpart to any theoretical worst-case-scenario analyses performed via the formulas in the work of Avrachenkov[22].

Finally, to verify the fundamental necessity and efficacy of ILILC for systems with unstable inverses, 2 trials of traditional stable-inversion-free NILC (3), (11) are applied to each truth model.

All computations are performed on a desktop computer with a 4 GHz CPU and 16 GB of RAM.

5.4 | Results and Discussion

The condition number of $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}_0)$ is 1×10^{17}. Attempted inversion of this matrix in MATLAB yields an inverse matrix with average nonzero element magnitude of 4×10^{13} and max element magnitude of 3×10^{16}. Consequently, $\mathbf{u}_1$ generated by (3), (11) has an average element magnitude of 2×10^{10} m and a max element magnitude of 8×10^{11} m, which is so large that $\mathbf{y}_1$ and $\frac{\partial \hat{\mathbf{g}}}{\partial \mathbf{u}}(\mathbf{u}_1)$ contain `NaN` elements for all simulations. Conversely, while some simulations using ILILC , i.e. (3), (31), diverge due to excessive model error, the majority converge. Additionally, the computation times given in Table 2 show that Procedure 1 successfully front-loads almost all of the required computation; intertrial computation time is almost always less than 150 ms. Together, these results validate the fundamental claim that the direct application of Newton's method in NILC is insufficient for systems with unstable inverses, and that the combination of ILILC and stable inversion fills this gap.

To accompany the quantitative metric $\|e_\theta\|_2$, Figure 4 offers a qualitative sense of the degree of model error in this study by comparing two representative ILILC solution trajectories $u_{50}(k)$ with the solution to the $\|e_\theta\|_2 = 0$, $\omega_c(k) = \omega_y(k) = 0$ scenario. The lower-model-error representative solution is from within the range of $\|e_\theta\|_2$ for which all simulations converged, while the higher-model-error solution comes from a bin in which some simulations diverged. A more detailed analysis of the boundaries in θ-space determining convergence or divergence of a simulation is beyond the scope of this work. However, the given trajectories illustrate that even in the conservative subspace defined by the 100% convergent bins learning bridges a visible performance gap, and that beyond this subspace there are far greater performance gains to be had.

Finally, a statistical comparison of the performance and robustness of ILILC with stable inversion and gradient ILC is given in Figure 5. The tuning of gradient ILC indeed yields comparable robustness to ILILC, with ILILC 97% as likely to converge as gradient ILC over all simulations. The convergence rates of the two ILC schemes, however, differ substantially, with gradient

TABLE 2 Computation Times for the Steps of Procedure 1

When Performed	Step	Operation	Time [s]
Once, before execution of trial 0	1-2	Derive the minimal, similarity transformed inverse state space system	6
	3	Fixed-point-problem-solving-based stable inversion to derive the inverse state trajectory as a function of $\hat{\mathbf{y}}$	95
	4	Conversion to lifted input-output model $\hat{\mathbf{g}}(\cdot)$	0.3
	5	Automatic differentiation to produce $\frac{\partial \hat{\mathbf{g}}^{-1}}{\partial \hat{\mathbf{y}}}(\cdot)$	573
Between trials 49,000 samples	6	Update of feedforward input trajectory via $L_\ell = \frac{\partial \hat{\mathbf{g}}^{-1}}{\partial \hat{\mathbf{y}}}(\mathbf{y}_\ell)$ and (3)	Mean: 0.138 Std: 0.006

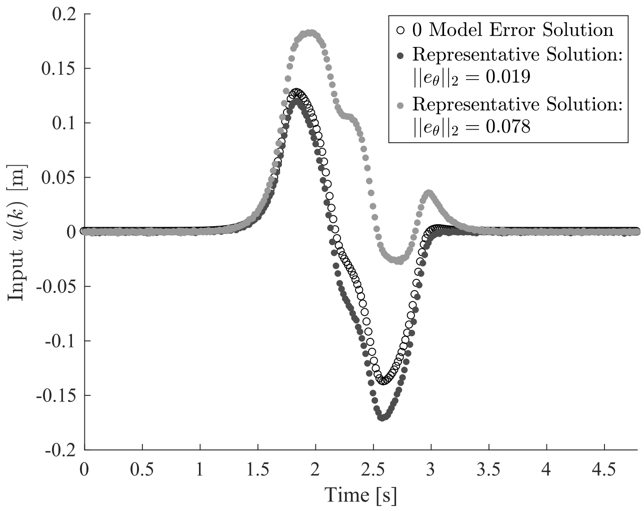

FIGURE 4 Representative input solution trajectories from low- and high-model-error ILILC simulations compared with the solution to the zero-model-error problem. The zero-model-error solution is the input trajectory that would be chosen for feed-forward control in the absence of learning, and differs notably from both minimum-error trajectories found by ILILC with stable inversion.

TABLE 3 Transient Convergence Rates for ILILC and Gradient ILC

ILC Law	Mean	Standard Deviation
Gradient ILC	0.76	0.17
ILILC + Stab. Inv.	0.41	0.27

ILC taking over 3 times as many trials as ILILC to converge on average. The mean transient convergence rate values tabulated in Table 3 give a more portable quantification of ILILC's advantage, having a convergence rate nearly half that of gradient ILC's.

This analysis confirms that ILILC with stable inversion is an important addition to the engineer's toolbox because it enables ILC synthesis from nonlinear non-minimum phase models and delivers the fast convergence characteristic of algorithms based on Newton's method.

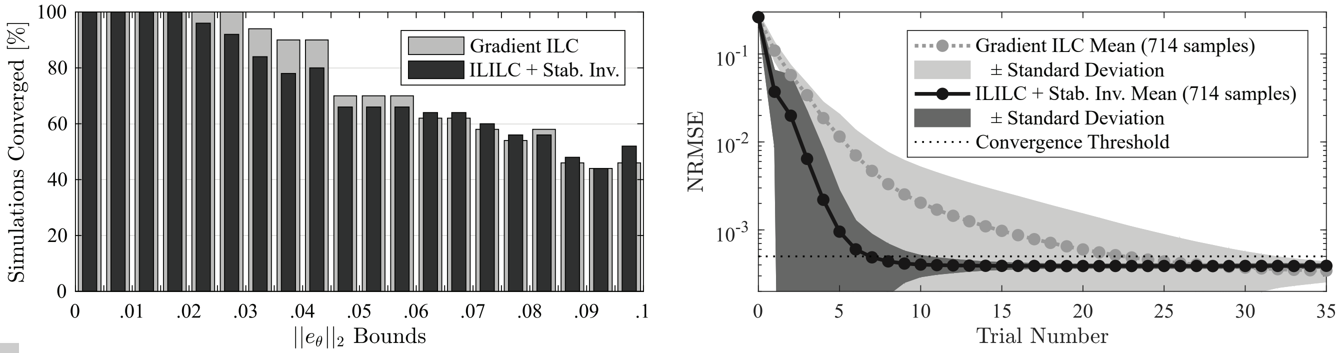

FIGURE 5 *Left:* Histogram giving the percentage of simulations converged in each bin of the model error metric $\|e_\theta\|_2$. *Right:* Mean value of NRMSE for each ILC trial over all simulations that are convergent for both gradient ILC and ILILC with stable inversion. This illustrates that for comparable robustness to model error, ILILC converges substantially faster than gradient ILC.

6 | CONCLUSION

This work introduces and validates a new ILC synthesis scheme applicable to nonlinear time-varying systems with unstable inverses and relative degree greater than 1. This is done with the support of nonlinear stable inversion, which is advanced from the prior art via proof of convergence for an expanded class of systems and methods for improved practical implementation. In all, this results in a new, broadly implementable ILC scheme displaying a competitive convergence speed under benchmark testing.

Future work may focus on further broadening the applicability of ILILC by relaxing reference and initial condition repetitiveness assumptions, and on the extension of ILILC with a potentially adaptive tuning parameter or other means to enable the exchange of some speed for robustness when called for. Levenberg-Marquardt-Fletcher algorithms may offer one source of inspiration for such work.

ACKNOWLEDGMENTS

This work is supported by the U.S. National Science Foundation (CAREER Award #1351469), the U.S. Department of Commerce, National Institute of Standards and Technology (Award 70NANB20H137), and the Netherlands Organization for Scientific Research (research programme VIDI project 15698)

Conflict of interest

The authors declare no potential conflict of interests.

How to cite this article: Spiegel I. A., N. Strijbosch, T. Oomen, and K. Barton (2021), Iterative learning control of discrete-time nonlinear non-minimum phase models via stable inversion, *Int J Robust Nonlinear Control*, *Submitted Manuscript* .

APPENDIX

A FAILURE OF OTHER ILC SCHEMES FOR SYSTEMS WITH UNSTABLE INVERSES

This appendix demonstrates that the sufficient conditions for convergence proposed by past works[5–9] on ILC for discrete-time nonlinear systems are in actuality not sufficient for at least some cases of systems having unstable inverses. This is done by running model-error-free ILC simulations that are guaranteed to converge by the past works, and observing them to diverge instead.

Each of references 5–9 proposes sufficient conditions for the convergence $\lim_{\ell\to\infty} \mathbf{e}_\ell = 0_{N-\mu+1}$ of a particular ILC scheme applied to a particular class of nonlinear dynamics. All of these classes of nonlinear dynamics are supersets of the SISO LTI dynamics

$$x_\ell(k+1) = Ax_\ell(k) + Bu_\ell(k) \tag{A1a}$$

$$y_\ell(k) = Cx_\ell(k) \tag{A1b}$$

with relative degree $\mu = 1$, i.e. $CB \neq 0$. Additionally, assume (A1) is stable and $x_\ell(0)$ is such that $y_\ell(0) = r_\ell(0)\ \forall \ell$. Given a system of this structure, the ILC schemes and convergence conditions of the past works reduce to the following.

From reference 5 the learning law is

$$u_{\ell+1}(k) = u_\ell(k) + L_\ell(k)\left(\gamma_1 e_\ell(k+1) + \gamma_0 e_\ell(k)\right) \tag{A2}$$

where $L \in \mathbb{R}$ is a potentially time-varying and trial-varying part of the learning gain and $\gamma_1, \gamma_0 \in \mathbb{R}$ are trial-invariant, time-invariant learning gains with $\gamma_1 \neq 0$. The learning laws of references 6–9 are special cases of (A2): reference 6 sets $\gamma_1 = 1$, $\gamma_0 = -1$, reference 7 sets L to be trail-invariant, $\gamma_1 = 1, \gamma_0 = 0$, reference 8 sets $\gamma_1 = 1$ and leaves γ_0 free, and reference 9 sets L to be trial-invariant and time-invariant, $\gamma_1 = 1, \gamma_0 = 0$.

Each work presents a different variation of convergence analysis, but all propose a sufficient condition for the convergence $\lim_{\ell\to\infty} \mathbf{e}_\ell = 0_{N-\mu+1}$ under their ILC scheme. References 5, 7, 8 use

(C9) $|1 - L_\ell(k)\gamma_1 CB| < 1\ \forall\, k, \ell$.

Reference 9 uses the stricter condition

(C10) $0 < L_\ell(k)CB < 1\ \forall\, k, \ell$.

Finally, reference 6 uses the combination of (C9) and

(C11) $\|A\| > 1$

where any consistent norm may be chosen for $\|\cdot\|$.

Consider the example system and learning gain

$$A = \begin{bmatrix} -0.3 & -0.79 & 0.53 \\ 0 & 0.5 & 1 \\ 0 & -0.36 & 0.5 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 0 \\ 1.34 \end{bmatrix}$$
$$C = \begin{bmatrix} 0.7 & 1.1 & -0.74 \end{bmatrix} \qquad x_\ell(0) = 0\ \forall \ell \tag{A3}$$

$$L_\ell(k) = 0.5(CB)^{-1} \quad \forall\, k, \ell \tag{A4}$$

with the reference given in Figure 2 and the zeroth control input $u_0(k) = 0\ \forall k$. This system has an unstable inverse.

The plant (A3) satisfies (C11), and with (A4) it satisfies (C9) and (C10) for $\gamma_1 = 1$. Thus, according to references 5–9 the ILC scheme (A2) is guaranteed to yield tracking error convergence in a model-error-free simulation. However, Figure A1 shows that the tracking error diverges under (A2), meaning that satisfaction of (C9)-(C11) is not actually sufficient for the convergence of all systems (A1) under the learning law (A2) in practice. This illustrates that the failure to account for phenomena arising from inverse instability is not unique to NILC, but rather pervades the literature on ILC with discrete-time nonlinear systems.

In light of the counterexample given by (A3) to the sufficiency of (C9)-(C11) for the convergence of the ILC schemes in references 5–9, it is desirable to formalize an additional condition that precludes systems such as (A3) from consideration for the application of these ILC schemes. Such a condition is given by:

(C12) equation (21) must be asymptotically stable about its solution.

For SISO LTI systems with relative degree $\mu \geq 1$ (i.e. systems of class (A1)), (C12) is equivalent to

$$\mathrm{SpectralRadius}\left(A - B\left(CA^{\mu-1}B\right)^{-1} CA^{\mu}\right) < 1 \tag{A5}$$

where $A - B\left(CA^{\mu-1}B\right)^{-1} CA^{\mu}$ is the state matrix of the inverse system. (A3) violates this condition, but many systems satisfy it, including all damped harmonic oscillators discretized via the forward Euler method. While sufficient, note that (C9)-(C12) might not be necessary conditions. Analysis of necessary conditions for error convergence under past works' ILC schemes is beyond the scope of this work. As shown in Figure A1, the ILC scheme proposed by the present article is capable of solving the problem presented by the given counterexample—(A3)—to past works' ILC schemes.

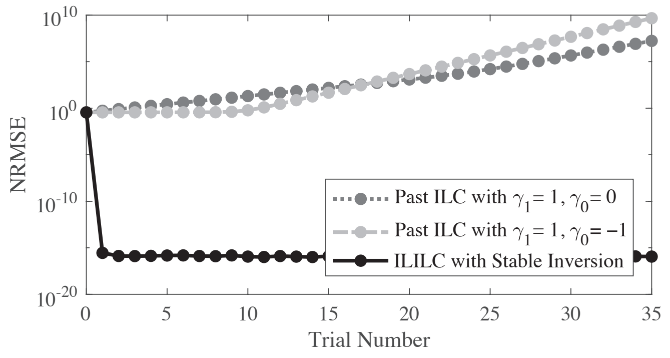

FIGURE A1 NRMSE versus trial number of past works' ILC schemes (A2) applied with learning gain (A4) to the system (A3). These NRMSEs monotonically increase, confirming the inability of the past work on ILC with discrete-time nonlinear systems to account for unstable inverses. The NRMSE trajectory yielded by the stable-inversion-supported ILILC scheme proposed by this article is also displayed. The convergence of this ILC scheme when applied to (A3) reiterates its ability to control such non-minimum phase systems.

References

1. Freeman CT. Upper Limb Electrical Stimulation Using Input-Output Linearization and Iterative Learning Control. *IEEE Transactions on Control Systems Technology* 2015; 23(4): 1546–1554. doi: 10.1109/TCST.2014.2363412

2. Yu Q, Hou Z, Xu JX. D-Type ILC Based Dynamic Modeling and Norm Optimal ILC for High-Speed Trains. *IEEE Transactions on Control Systems Technology* 2018; 26(2): 652–663. doi: 10.1109/TCST.2017.2692730

3. Rafajłowicz W, Jurewicz P, Reiner J, Rafajłowicz E. Iterative Learning of Optimal Control for Nonlinear Processes with Applications to Laser Additive Manufacturing. *IEEE Transactions on Control Systems Technology* 2019; 27(6): 2647–2654. doi: 10.1109/TCST.2018.2865444

4. Xing X, Liu J. Modeling and robust adaptive iterative learning control of a vehicle-based flexible manipulator with uncertainties. *International Journal of Robust and Nonlinear Control* 2019; 29: 2385–2405. doi: 10.1002/rnc.4500

5. Jang TJ, Ahn HS, Choi CH. Iterative learning control for discrete-time nonlinear systems. *International Journal of Systems Science* 1994; 25(7): 1179–1189. doi: 10.1080/00207729408949269

6. Saab SS. Discrete-Time Learning Control Algorithm for a Class of Nonlinear Systems. In: *Proceedings of 1995 American Control Conference*, IEEE; 1995; Seattle: 2793–2743

7. Wang D. Convergence and robustness of discrete time nonlinear systems with iterative learning control. *Automatica* 1998; 34(11): 1445–1448. doi: 10.1016/S0005-1098(98)00098-3

8. Sun M, Wang D. Initial shift issues on discrete-time iterative learning control with system relative degree. *IEEE Transactions on Automatic Control* 2003; 48(1): 144–148. doi: 10.1109/TAC.2002.806668

9. Zhang Y, Liu J, Ruan X. Iterative learning control for uncertain nonlinear networked control systems with random packet dropout. *International Journal of Robust and Nonlinear Control* 2019; 29: 3529–3546. doi: 10.1002/rnc.4568

10. Xing J, Chi R, Lin N. Adaptive iterative learning control for 2D nonlinear systems with nonrepetitive uncertainties. *International Journal of Robust and Nonlinear Control* 2021; 31: 1168–1180. doi: 10.1002/rnc.5347

11. Shieh HJ, Hsu CH. An adaptive approximator-based backstepping control approach for piezoactuator-driven stages. *IEEE Transactions on Industrial Electronics* 2008; 55(4): 1729–1738. doi: 10.1109/TIE.2008.917115

12. Hackl CM, Hopfe N, Ilchmann A, Mueller M, Trenn S. Funnel Control for Systems with Relative Degree Two. *SIAM Journal on Control and Optimization* 2013; 51(2): 1046–1060. doi: 10.1137/100799903

13. Geniele H, Patel RV, Khorasani K. End-Point Control of a Flexible-Link Manipulator: Theory and Experiments. *IEEE Transactions on Control Systems Technology* 1997; 5(6): 556–570. doi: 10.1109/87.641401

14. Münz U, Papachristodoulou A, Allgöwer F. Robust consensus controller design for nonlinear relative degree two multi-agent systems with communication constraints. *IEEE Transactions on Automatic Control* 2011; 56(1): 145–151. doi: 10.1109/TAC.2010.2084150

15. Escobar G, Ortega R, Sira-Ramirez H, Vilain JP, Zein I. An experimental comparison of several nonlinear controllers for power converters. *IEEE Control Systems Magazine* 1999; 19(1): 66–82. doi: 10.1109/37.745771

16. De Battista H, Mantz RJ. Dynamical variable structure controller for power regulation of wind energy conversion systems. *IEEE Transactions on Energy Conversion* 2004; 19(4): 756–763. doi: 10.1109/TEC.2004.827705

17. Gutierrez HM, Ro PI. Magnetic servo levitation by sliding-mode control of nonaffine systems with algebraic input invertibility. *IEEE Transactions on Industrial Electronics* 2005; 52(5): 1449–1455. doi: 10.1109/TIE.2005.855651

18. Khosla PK, Kanade T. Experimental Evaluation of Nonlinear Feedback and Feedforward Control Schemes for Manipulators. *The International Journal of Robotics Research* 1988; 7(1): 18–28. doi: 10.1177/027836498800700102

19. Schitter G, Stark RW, Stemmer A. Sensors for closed-loop piezo control: Strain gauges versus optical sensors. *Measurement Science and Technology* 2002; 13: N47–N48. doi: 10.1088/0957-0233/13/4/404

20. Awtar S, Craig KC. Electromagnetic Coupling in a dc Motor and Tachometer Assembly. *Journal of Dynamic Systems, Measurement, and Control* 2004; 126(3): 684–691. doi: 10.1115/1.1789543

21. Wit dCC, Siciliano B, Bastin G. *Theory of Robot Control*. Springer-Verlag . 1996

22. Avrachenkov KE. Iterative learning control based on quasi-Newton methods. In: *Proceedings of the 37th IEEE Conference on Decision & Control*, No. December in December. IEEE; 1998; Tampa: 170–174

23. Spiegel IA, Barton K. A Closed-Form Representation of Piecewise Defined systems and their Integration with Iterative Learning Control. In: *2019 American Control Conference (ACC)*, IEEE; 2019; Philadelphia, PA: 2327–2333

24. Xu JX, Tan Y. *Linear and Nonlinear Iterative Learning Control*. 404. Berlin: Springer-Verlag . 2003

25. Tomizuka M. Zero Phase Error Tracking Algorithm for Digital Control. *Journal of Dynamic Systems, Measurement and Control* 1987; 109(1): 65–68. doi: 10.1115/1.3143822

26. Zundert vJ, Bolder J, Koekebakker S, Oomen T. Resource-efficient ILC for LTI/LTV systems through LQ tracking and stable inversion: Enabling large feedforward tasks on a position-dependent printer. *Mechatronics* 2016; 38: 76–90. doi: 10.1016/j.mechatronics.2016.07.001

27. Devasia S, Chen D, Paden B. Nonlinear inversion-based output tracking. *IEEE Transactions on Automatic Control* 1996; 41(7): 930–942. doi: 10.1109/9.508898

28. Devasia S, Paden B. Stable Inversion For Nonlinear Nonminimum-phase Time-varying Systems. *IEEE Transactions on Automatic Control* 1998; 43(2): 283–288. doi: 10.1109/9.661082

29. Zeng G, Hunt LR. Stable inversion for nonlinear discrete-time systems. *IEEE Transactions on Automatic Control* 2000; 45(6): 1216–1220. doi: 10.1109/9.863610

30. Andersson JAE, Gillis J, Horn G, Rawlings JB, Diehl M. CasADi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation* 2019; 11: 1–36. doi: 10.1007/s12532-018-0139-4

31. Eksteen JJA, Heyns PS. Improvements in stable inversion of NARX models by using Mann iteration. *Inverse Problems in Science and Engineering* 2016; 24(4): 667–691. doi: 10.1080/17415977.2015.1055262

32. Agarwal RP. *Difference Equations and Inequalities: Theory, Methods, and Applications*. New York: Marcel Dekker, Inc. 2 ed. 2000.

33. Rump SM. Inversion of extremely ill-conditioned matrices in floating-point. *Japan Journal of Industrial and Applied Mathematics* 2009; 26(2-3): 249–277. doi: 10.1007/BF03186534

34. Belsley DA, Kuh E, Welsch RE. *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. Hoboken, NJ: John Wiley & Sons, Inc. . 1980

35. Chu B, Owens D. Singular Value distribution of non-minimum phase systems with application to iterative learning control. In: *Proceedings of the 52nd IEEE Conference on Decision and Control*, IEEE; 2013; Florence: 6700–6705

36. Moore KL. *Iterative Learning Control for Deterministic Systems*. London: Springer-Verlag . 1993

37. Norrlöf M, Gunnarsson S. Time and frequency domain convergence properties in iterative learning control. *International Journal of Control* 2002; 75(14): 1114–1126. doi: 10.1080/00207170210159122

38. Dijkstra BG. *Iterative Learning Control with Application to a Wafer Stage*. PhD thesis. Delft University of Technology, ; 2004.

39. Lee JH, Lee KS, Kim WC. Model-based iterative learning control with a quadratic criterion for time-varying linear systems. *Automatica* 2000; 36(5): 641–657. doi: 10.1016/S0005-1098(99)00194-6

40. Owens DH, Hatonen JJ, Daley S. Robust monotone gradient-based discrete-time iterative learning control. *International Journal of Robust and Nonlinear Control* 2009; 19: 634–661. doi: 10.1002/rnc.1338