2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Toward Understanding State Representation Learning in MuZero: A Case Study in Linear Quadratic Gaussian Control

Yi Tian, Kaiqing Zhang, Russ Tedrake, and Suvrit Sra

***Abstract*— We study the problem of representation learning for control from partial and potentially high-dimensional observations. We approach this problem via *direct latent model learning*, where one directly learns a dynamical model in some latent state space by *predicting costs*. In particular, we establish *finite-sample guarantees* of finding a near-optimal representation function and a near-optimal controller using the directly learned latent model for infinite-horizon time-invariant Linear Quadratic Gaussian (LQG) control. A part of our approach to latent model learning closely resembles *MuZero*, a recent breakthrough in empirical reinforcement learning, in that it learns latent dynamics *implicitly* by predicting *cumulative costs*. A key technical contribution of this work is to prove persistency of excitation for a new stochastic process that arises from the analysis of quadratic regression in our approach.**

I. Introduction

Control with a learned latent model is state-of-the-art in several reinforcement learning (RL) benchmarks, including board games, Atari games, and visuomotor control [18, 25, 6]. To better understand this modern machinery, we introduce it to a classical optimal control problem, namely Linear Quadratic Gaussian (LQG) control, and study its theoretical, finite-sample performance. Essential to this approach is the learning of two components: a *state representation* function that maps an observed history to some latent state, and a *latent model* that predicts the transition and cost in the latent state space. The latent model is usually a Markov decision process, using which we obtain a policy in the latent space or execute online planning.

What is the correct *objective* to optimize for learning a latent model? One popular choice is to learn a function that *reconstructs the observation* from the latent state [3–6]. A latent model learned this way is *agnostic to control tasks* and retains all the information about the environment. This class of approaches can achieve satisfactory performance, but are prone to background distraction and control-irrelevant information [2]. The second class of methods learn an *inverse model* that infers actions from latent states at different time steps [16, 11]. A latent model learned with this methodology is also task agnostic but extracts control-relevant information. In contrast, *task-relevant* representations can be learned by *predicting costs* in the control task [14, 26, 18]. The concept that a good latent state should be able to predict costs is intuitive, and the costs are directly relevant to optimal control. Hence, [24] refers to this class of methods as "direct latent model learning", which is the focus of this work.

The direct latent model learning method of particular interest to us is that of MuZero [18]. Announced by DeepMind in 2019, MuZero extends the line of works including AlphaGo [19] and AlphaZero [20] by not requiring knowledge of the game rules. MuZero matches the superhuman performance of AlphaZero in Go, shogi and chess, while outperforming model-free RL algorithms in Atari games. MuZero builds upon the powerful planning procedure of Monte Carlo Tree Search, with the major innovation being *learning a latent model*. The latent model replaces the rule-based simulator during planning, and avoids the burdensome planning in pixel space for Atari games.

The latent model learning of MuZero features three ingredients: 1) stacking frames, i.e., observations, as input to the representation function; 2) predicting costs, "optimal" values, and "optimal" actions from latent states; and 3) implicit learning of latent dynamics by predicting these quantities from latent states at future time steps. These are the defining characteristics of the MuZero-style algorithm that we shall consider. In MuZero, the "optimal" values and actions are found by the powerful online planning procedure. In this work, we simplify the setup by considering data collected using random actions, which are known to suffice for identifying a partially observable linear dynamical system [15]. In this setup, the values become those associated with this trivial policy and we do not predict actions since they are random noises anyway. Note that although our study of the above ingredients is directly motivated by MuZero, previous empirical works have also explored them. For example, frame stacking has been a widely used technique to handle partial observability [12, 13]; predicting values for learning a latent model has been studied in [14], which also learns the latent state transition implicitly.

Closely related to our work, [24] also considers provable direct latent model learning in LQG, but for the finite-horizon time-varying setting. Our work builds upon it and complements it in two ways: 1) we extend their algorithm to the time-invariant setting with a *stationary* representation function and latent model, which is closer to what has been deployed in practice; 2) we present and analyze a new, MuZero-style latent model learning algorithm. Both 1) and 2) introduce new technical challenges to be addressed. We summarize our contributions as follows.

This manuscript is a shorter version of the technical report `https://yi-t.github.io/files/srl_lti.pdf`. Please refer to the technical report for the missing details. Yi Tian, Russ Tedrake, and Suvrit Sra are with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, {yitian,russt,suvrit}@mit.edu. Kaiqing Zhang is with the Department of Electrical and Computer Engineering, University of Maryland, College Park, kaiqing@umd.edu.

- We show that two direct latent model learning methods provably solve infinite-horizon time-invariant LQG control by establishing finite-sample guarantees. Both methods only need a single trajectory; one resembles the method in [24], and the other resembles MuZero.
- By analyzing the MuZero-style algorithm, we notice the potential issue of coordinate misalignment; that is, costs can be invariant to certain transformations of the latent states, and implicit dynamics learning by predicting *one-step* transition may not recover the latent state coordinates consistently. This insight suggests the need of predicting *multi-step* transition or other coordinate alignment procedures in implicit dynamics learning.
- Technically, we overcome the difficulty of having *dependent* data samples in a single trajectory for latent model learning, by proving a new result about the persistency of excitation for a stochastic process that arises from the analysis of the quadratic regression subroutine in both of our methods.

Notation. Let $a \wedge b$ denote the minimum between scalars a and b. Given vector $v \in \mathbb{R}^d$, let $\|v\|$ denote its ℓ_2 norm and $\|v\|_P := (v^\top P v)^{1/2}$ for positive semidefinite $P \in \mathbb{R}^{d\times d}$. Semicolon ";" denotes stacking vectors or matrices vertically. For a collection of d-dimensional vectors $(v_t)_{t=i}^j$, let $v_{i:j} := [v_i; v_{i+1}; \ldots; v_j] \in \mathbb{R}^{d(j-i+1)}$ denote the concatenation along the column. For random variable x, let $\|x\|_{\psi_1}$ denote its subexponential norm. Let $\mathrm{svec}(\cdot)$ denote the operator of flattening a symmetric matrix by stacking its columns; it does not repeat the off-diagonal elements, but scales them by $\sqrt{2}$ [17].

II. Problem setup

A partially observable linear time-invariant (LTI) dynamical system is described by

$$x_{t+1} = A^* x_t + B^* u_t + w_t, \quad y_t = C^* x_t + v_t, \tag{1}$$

with state $x_t \in \mathbb{R}^{d_x}$, observation $y_t \in \mathbb{R}^{d_y}$, and control $u_t \in \mathbb{R}^{d_u}$ for all $t \geq 0$. Process noises $(w_t)_{t\geq 0}$ and observation noises $(v_t)_{t\geq 0}$ are i.i.d. zero-mean Gaussian random vectors with covariance matrices Σ_w and Σ_v, respectively, and the two sequences are mutually independent. Let initial state x_0 be sampled from $\mathcal{N}(0, \Sigma_0)$. The quadratic cost function is given by

$$c(x,u) = \|x\|^2_{Q^*} + \|u\|^2_{R^*}, \tag{2}$$

where $Q^* \succcurlyeq 0$ and $R^* \succ 0$.

A policy/controller π determines an action/control input u_t at time step t based on the history $[y_{0:t}; u_{0:(t-1)}]$ up to this time step. For $t \geq 0$, $c_t := c(x_t, u_t)$ denotes the cost at time step t. Given a policy π, let

$$J^\pi := \limsup_{T\to\infty} \ \mathbb{E}\left[\frac{1}{T}\sum_{t=0}^{T-1} c_t\right] \tag{3}$$

denote the time-average expected cost. The objective of LQG control is to find a policy π such that J^π is minimized.

In the fully observable setting, known as the linear quadratic regulator (LQR), $y_t = x_t$. A linear controller with feedback gain $K \in \mathbb{R}^{d_u \times d_x}$ determines action $u_t = K x_t$ at time step t. Let $J^K(A^*, B^*, Q^*, R^*)$ denote the time-average expected cost (3) in the LQR problem (A^*, B^*, Q^*, R^*) under feedback gain K and define $J^*(A^*, B^*, Q^*, R^*) := \min_K J^K(A^*, B^*, Q^*, R^*)$.

We make the following standard assumptions.

Assumption 1: System dynamics (1) and cost (2) satisfy:

1) The system is stable, that is, $\rho(A^*) < 1$.
2) (A^*, B^*) is ν-controllable for some $\nu > 0$, that is, the controllability matrix $\Phi_c(A^*, B^*) := [B^*, A^*B^*, \ldots, (A^*)^{d_x-1}B^*]$ has rank d_x and $\sigma_{\min}(\Phi_c(A^*, B^*)) \geq \nu$.
3) (A^*, C^*) is ω-observable for some $\omega > 0$, that is, the observability matrix $\Phi_o(A^*, C^*) := [C^*; C^*A^*; \ldots; C^*(A^*)^{d_x-1}]$ has rank d_x and $\sigma_{\min}(\Phi_o(A^*, C^*)) \geq \omega$.
4) $(A^*, \Sigma_w^{1/2})$ is η-controllable for some $\eta > 0$.
5) $(A^*, (Q^*)^{1/2})$ is μ-observable for some $\mu > 0$.
6) $\Sigma_v \succcurlyeq \sigma_v^2 I$ for some $\sigma_v > 0$; this can always be achieved by inserting Gaussian noises with full-rank covariance matrices to the observations.
7) $R^* \succcurlyeq r^2 I$ for some $r > 0$.
8) The operator norms of A^*, B^*, C^*, Q^*, R^*, Σ_w, Σ_v, Σ_0 are $\mathcal{O}(1)$ and the singular value lower bounds ν, ω, ν, η, σ_v, r are $\Omega(1)$.

If the system parameters $(A^*, B^*, C^*, Q^*, R^*, \Sigma_w, \Sigma_v)$ are known, the optimal policy is obtained by combining the Kalman filter

$$z^*_{t+1} = A^* z^*_t + B^* u_t + L^*(y_{t+1} - C^*(A^* z^*_t + B^* u_t)) \tag{4}$$

with the optimal feedback gain K^* of the linear quadratic regulator (LQR) such that $u_t = K^* z^*_t$, where L^* is the Kalman gain, and at the initial time step, we can set, e.g., $z^*_0 = L^* y_0$. This fact is known as the *separation principle*, and the Kalman gain and optimal feedback gain are given by

$$L^* = S^*(C^*)^\top (C^* S^* (C^*)^\top + \Sigma_v)^{-1}, \tag{5}$$

$$K^* = -((B^*)^\top P^* B^* + R)^{-1}(B^*)^\top P^* A^*, \tag{6}$$

where S^* and P^* are determined by their respective discrete-time algebraic Riccati equations (DAREs):

$$S^* = A^*\big(S^* - S^*(C^*)^\top (C^* S^* (C^*)^\top + \Sigma_v)^{-1} C^* S^*\big)(A^*)^\top + \Sigma_w, \tag{7}$$

$$P^* = (A^*)^\top \big(P^* - P^* B^* ((B^*)^\top P^* B^* + R^*)^{-1}(B^*)^\top P^*\big) A^* + Q^*. \tag{8}$$

Assumptions 1.1 guarantees that the system does not explode under the excitation of random control inputs. Assumptions 1.2 to 1.7 guarantee the existence and uniqueness of positive definite solutions S^* and P^*; Assumption 1.8 further guarantees that their operator norms are $\mathcal{O}(1)$ and minimum singular values are $\Omega(1)$.

We consider the data-driven control setting, where the LQG model $(A^*, B^*, C^*, Q^*, \Sigma_w, \Sigma_v)$ is unknown. For simplicity, we assume R^* is known, though our approaches can be readily extended to the case where it is unknown by learning it from predicting costs.

A. Latent model of LQG

The stationary Kalman filter (4) asymptotically produces the optimal *state estimation* in the sense of minimum mean squared errors. With a finite horizon, however, the optimal state estimator is time-varying, given by

$$z^*_{t+1} = A^* z^*_t + B^* u_t + L^*_{t+1}(y_{t+1} - C^*(A^* z^*_t + B^* u_t)), \quad (9)$$

where L^*_t is the time-varying Kalman gain, converging to L^* as $t \to \infty$. This convergence is equivalent to that of error covariance matrix $\mathbb{E}[(x_t - z^*_t)(x_t - z^*_t)^\top]$, which happens exponentially fast [8]. Hence, for simplicity, we assume this error covariance matrix is stationary at the initial time step by the choice of z^*_0 so that $L^*_t = L^*$ for $t \geq 1$; this assumption is common in the literature [9, 10, 7]. The *innovation* term $i_{t+1} := y_{t+1} - C^*(A^* z^*_t + B^* u_t)$ is independent of the history $(y_0, u_0, y_1, u_1, \ldots, y_{t+1})$ and $(i_t)_{t\geq 1}$ are mutually independent. The following proposition taken from [24, Proposition 1] represents the system in terms of the state estimates obtained by the Kalman filter, which we refer to as the *latent model*.

Proposition 1: Let $(z^*_t)_{t\geq 1}$ be state estimates given by the time-varying Kalman filter. Then, for $t \geq 0$,

$$z^*_{t+1} = A^* z^*_t + B^* u_t + L^* i_{t+1},$$

where $L^* i_{t+1}$ is independent of z^*_t and u_t, i.e., the state estimates follow the same linear dynamics with noises $L^* i_{t+1}$. The cost at step t can be reformulated as functions of the state estimates by

$$c_t = \|z^*_t\|^2_{Q^*} + \|u_t\|^2_{R^*} + b^* + \gamma_t + \eta_t,$$

where $b^* > 0$, and $\gamma_t = \|z^*_t - x_t\|^2_{Q^*} - b^*$, $\eta_t = \left\langle z^*_t, x_t - z^*_t \right\rangle_{Q^*}$ are both zero-mean subexponential random variables. Moreover, $b^* = \mathcal{O}(1)$ and $\|\gamma_t\|_{\psi_1} = \mathcal{O}(d_x^{1/2})$; if control $u_t \sim \mathcal{N}(0, \sigma_u^2 I)$ for $t \geq 0$, then we have $\|\eta_t\|_{\psi_1} = \mathcal{O}(d_x^{1/2})$.

Proposition 1 shows that the dynamics of the state estimates computed by the time-varying Kalman filter is the same as the original system up to noises; the costs are also the same, up to constants and noises. Hence, a latent model can be parameterized by (A, B, Q, R^*), with the constant b^* and noises neglected due to their irrelevance to planning. A stationary latent policy is a linear controller $u_t = K z_t$ on latent state z_t, parameterized by feedback gain $K \in \mathbb{R}^{d_u \times d_x}$.

The latent model enables us to find a good latent policy. To learn such a latent model and to deploy a latent policy in the original partially observable system, we need a representation function. Let $\overline{A}^* := (I - L^* C^*)A^*$ and $\overline{B}^* := (I - L^* C^*)B^*$. Then, the Kalman filter can be written as $z^*_{t+1} = \overline{A}^* z^*_t + \overline{B}^* u_t + L^* y_{t+1}$. For $t \geq 0$, unrolling the recursion gives

$$\begin{aligned} z^*_t &= \overline{A}^*(\overline{A}^* z^*_{t-2} + \overline{B}^* u_{t-2} + L^* y_{t-1}) + \overline{B}^* u_{t-1} + L^* y_t \\ &= [(\overline{A}^*)^{t-1} L^*, \ldots, L^*] y_{1:t} + [(\overline{A}^*)^{t-1}\overline{B}^*, \ldots, \overline{B}^*] u_{0:(t-1)} + (\overline{A}^*)^t z^*_0 \\ &=: M^*_t [y_{1:t}; u_{0:(t-1)}; z^*_0], \end{aligned}$$

where $M^*_t \in \mathbb{R}^{d_x \times (t d_y + t d_u + d_x)}$. This means the representation function can be parameterized as linear mappings for full histories (with y_0 replaced by z^*_0). Despite the simplicity, the input dimension of the function grows linearly in time, making it intractable to estimate the state using the full history for large t; nor is it necessary, since the impact of old data decreases exponentially. Under Assumption 1, $\rho(\overline{A}^*) < 1$ [1, Appendix E.4]. With an H-step truncated history, the state estimate can be written as

$$\begin{aligned} z^*_t &= [(\overline{A}^*)^{H-1} L^*, \ldots, L^*] y_{(t-H+1):t} \\ &\quad + [(\overline{A}^*)^{H-1}\overline{B}^*, \ldots, \overline{B}^*] u_{(t-H):(t-1)} + \delta_t \\ &=: M^* [y_{(t-H+1):t}; u_{(t-H):(t-1)}] + \delta_t, \end{aligned}$$

where $\delta_t = (\overline{A}^*)^H z^*_{t-H}$, whose impact decays exponentially in H and can be neglected for sufficiently large H, since z^*_{t-H} converges to a stationary distribution and its norm is bounded with high probability. Hence, the representation function that we aim to recover is $M^* \in \mathbb{R}^{d_x \times H(d_y + d_u)}$, which takes as input the H-step history $h_t = [y_{(t-H+1):t}; u_{(t-H):(t-1)}]$. Henceforth, we let $d_h := H(d_y + d_u)$. Then, a representation function is parameterized by matrix $M \in \mathbb{R}^{d_x \times d_h}$.

Overall, a policy is a combination of a representation function M and a feedback gain K in the latent model, denoted by $\pi = (M, K)$. Learning to solve LQG control in this framework can thus be achieved by: 1) learning representation function M; 2) extracting latent model (A, B, Q, R^*); and 3) finding the optimal K by planning in the latent model. Next, we introduce our approach following this pipeline.

III. Method

In practice, latent model learning methods collect trajectories by interacting with the system online using some policy; the trajectories are used to improve the learned latent model, which in turn improves the policy. In LQG control, it is known that the simple setup allows us to learn a good latent model from a single trajectory, collected using zero-mean Gaussian inputs; see e.g., [15]. This is also how we assume the data are collected. We note that our results also apply to data from multiple independent trajectories using the same zero-mean Gaussian inputs.

In direct latent model learning, state representations are learned by predicting costs. To learn the transition function in the latent model, two approaches are explored in the literature. The first approach explicitly minimizes transition prediction errors [23, 3]. Algorithmically, the overall loss is a combination of cost and transition prediction errors. The second approach, which MuZero takes, learns transition *implicitly*, by minimizing *cost prediction errors* at *future states* generated from the transition function [14, 18]. Algorithmically, the overall loss aggregates the cost prediction errors across multiple time steps. In both approaches, the coupling of different terms in the loss makes finite-sample analysis difficult. As observed in [24], the structure of LQG allows us to learn the representation function independently of learning the transition function. This allows us to formulate both approaches under the same direct latent model learning framework (Algorithm 1).

Algorithm 1 consists of three main steps. Lines 3 to 5 correspond to cost-driven representation learning. Lines 6 to 8 correspond to latent model learning, where the system dynamics can be identified either explicitly, by ordinary least

squares (SysID), or implicitly, by future cost prediction (CoSysID, Algorithm 2). Line 8 corresponds to latent policy optimization; in LQG this amounts to solving a DARE. Below we elaborate on cost-driven representation learning, SysID, and CoSysID in order.

A. Cost-driven representation learning

The procedure of cost-driven representation learning is almost identical to that in [24]. The main idea is to perform quadratic regression (11) to the d_x-step cumulative costs; these correspond to the value prediction in MuZero. By the μ-observability of $(A^*, (Q^*)^{1/2})$ (Assumption 1.5), the cost observability Gramian $\overline{Q}^* := \sum_{t=0}^{d_x-1}((A^*)^t)^\top Q^*(A^*)^t \succcurlyeq \mu^2 I$. Under zero control and zero noise, starting from x, the d_x-step cumulative cost is precisely $\|x\|^2_{\overline{Q}^*}$. Hence, $\hat{N}$ estimates $N^* = (M^*)^\top \overline{Q}^* M^*$; up to an orthonormal transformation, $\hat{M}$ recovers $M^{*\prime} := (\overline{Q}^*)^{1/2} M^*$, the representation function under an equivalent parameterization, termed as the *normalized parameterization* in [24], where

$$A^{*\prime} = (\overline{Q}^*)^{1/2} A^* (\overline{Q}^*)^{-1/2},\ B^{*\prime} = (\overline{Q}^*)^{1/2} B,\ C^{*\prime} = C^* (\overline{Q}^*)^{-1/2},$$
$$w'_t = (\overline{Q}^*)^{1/2} w_t,\ Q^{*\prime} = (\overline{Q}^*)^{-1/2} Q^* (\overline{Q}^*)^{-1/2}.$$

Due to the following proposition, the algorithm does not need to know the dimension d_x of the latent model; it can discover d_x from the eigenvalues of $\hat{N}$.

Proposition 2: Under i.i.d. control inputs $u_t \sim \mathcal{N}(0, \sigma_u^2 I)$ for $t \geq 0$, $\lambda_{\min}(\mathrm{Cov}(z_t^*)) = \Omega(\nu^2)$ for $t \geq d_x$, where ν is defined in Assumption 1.3. As long as $H \geq a$ for some dimension-free constant $a > 0$, M^* has rank d_x and $\sigma_{\min}(M^*) \geq \Omega(\nu H^{-1/2})$.

Proposition 2 is an adaption of [24, Proposition 2] to the infinite-horizon LTI setting. Necessarily, this implies that by our choice of H, $d_h = H(d_y + d_u) \geq d_x$. Moreover, since $\overline{Q}^* \succcurlyeq \mu^2 I$, $N^* = (M^*)^\top \overline{Q}^* M^*$ is a $d_h \times d_h$ matrix with rank d_x, and $\lambda^+_{\min}(N^*) \geq \lambda_{\min}(\overline{Q}^*)\lambda^2_{\min}(M^*) = \Omega(\mu^2\nu^2 H^{-1})$. Hence, if $\hat{N}$ is sufficiently close to N^*, by setting an appropriate threshold on the eigenvalues of $\hat{N}$, the dimension of the latent model equals the number of eigenvalues above it.

To find an approximate factorization of $\hat{N}$, let $\hat{N} = U\Lambda U^\top$ be its eigenvalue decomposition, where the diagonal elements of Λ are listed in a descending order, and U is an orthonormal matrix. Let Λ_{d_x} be the left-top block of Λ and U_{d_x} be the left d_x columns of U. By the Eckart-Young-Mirsky theorem, $\hat{M} = \max(\Lambda_{d_x}, 0)^{1/2} U_{d_x}^\top$, where "max" applies elementwise, is the solution to Line 4 of Algorithm 1, that is, the best approximate factorization of $\hat{N}$ among $d_x \times d_h$ matrices in terms of the Frobenius norm approximation error.

In the next two subsections, we move on to discuss learning latent dynamics, including the explicit approach SysID and the implicit approach CoSysID.

Algorithm 1 Direct latent model learning for LQG control

1: **Input:** length T, history length H, noise magnitude σ_u
2: Collect a trajectories of length $T+H$ using $u_t \sim \mathcal{N}(0, \sigma_u^2 I)$, for $t \geq 0$, to obtain $\mathscr{D}_{\text{raw}}$:
$$(y_0, u_0, c_0, y_1, u_1, c_1, \ldots, y_{T+H}) \tag{10}$$
3: Estimate the state representation function and cost constants by solving
$$\hat{N}, \hat{b}_0 \in \operatorname*{argmin}_{N=N^\top, b_0} \sum_{t=H}^{T+H-1} \left(\|h_t\|_N^2 + b_0 - \bar{c}_t\right)^2, \tag{11}$$
where $\bar{c}_t := \sum_{\tau=t}^{t+d_x-1}(c_\tau - \|u_\tau\|^2_{R^*})$
4: Find $\hat{M} \in \operatorname{argmin}_{M \in \mathbb{R}^{d_x \times H(d_y+d_u)}} \|M^\top M - \hat{N}\|_F$
5: Compute $\hat{z}_t = \hat{M}[y_{(t-H+1):t}; u_{(t-H):(t-1)}]$ for all $t \geq H$, so that the data are converted to $\mathscr{D}_{\text{state}}$:
$$(\hat{z}_H, u_H, c_H, \ldots, \hat{z}_{T+H-1}, u_{T+H-1}, c_{T+H-1}, \hat{z}_{T+H})$$
6: Run SysID (12) or CoSysID (Algorithm 2) to obtain dynamics matrices $(\hat{A}, \hat{B})$
7: Estimate the cost function by solving
$$\widetilde{Q}, \hat{b} \in \operatorname*{argmin}_{Q=Q^\top, b} \sum_{t=H}^{T+H-1} (\|\hat{z}_t\|_Q^2 + b - c_t)^2,$$
8: Truncate negative eigenvalues of $\widetilde{Q}$ to 0 to obtain $\hat{Q} \succcurlyeq 0$
9: Find feedback gain $\hat{K}$ from $(\hat{A}, \hat{B}, \hat{Q}, R^*)$ by (8) and (6)
10: **Return:** policy $\hat{\pi} = (\hat{M}, \hat{K})$

B. Explicit learning of system dynamics

Explicit learning of the system dynamics simply minimizes the transition prediction error in the latent space [23], or more generally, the statistical distances between the predicted and estimated distributions of the next latent state, like the KL divergence [3]. In linear systems, it suffices to use the ordinary least squares as the SysID procedure, that is, to solve

$$(\hat{A}, \hat{B}) \in \operatorname*{argmin}_{A,B} \sum_{t=H}^{T+H-1} \|A\hat{z}_t + Bu_t - \hat{z}_{t+1}\|^2. \tag{12}$$

In this linear regression, if $(\hat{z}_t)_{t \geq H}$ are the optimal state estimates $(z_t^*)_{t \geq H}$ (9), then [21] has shown finite-sample guarantees for $(\hat{A}, \hat{B})$. Here, $\hat{z}_t$ contains errors resulting from the representation function $\hat{M}$ and the residual error δ_t, but as long as T and H are large enough, SysID still has a finite-sample guarantee. We refer to the algorithm that instantiates Algorithm 1 with SysID as CoReL (COst-driven state REpresentation Learning). As the time-varying counterpart in [24], it provably solves learning for LQG control, as will be shown in Theorem 1.

C. Implicit learning of system dynamics (MuZero-style)

An important ingredient of latent model learning in MuZero [18] is to *implicitly* learn the transition function by minimizing the cost prediction error at *future latent states* generated from the transition function. Let $z_t = Mh_t$ denote the latent state given by representation function M at step t. Let $z_{t,0} = z_t$ and $z_{t,i} = Az_{t,i-1} + Bu_{t+i-1}$ for $i \geq 1$ be the future latent state predicted by dynamics (A, B) from z_t after i steps of transition. For a trajectory of length $T+H$ like (10), the loss that considers ℓ steps into the future is given by

$$\sum_{t=H}^{T+H-K-1} \sum_{i=0}^{\ell} (\|z_{t,i}\|_Q^2 + \|u_t\|^2_{R^*} + b - c_t)^2.$$

This loss involves powers of A up to A^ℓ; with the squared norm, the powers double, making the minimization over A hard to solve and analyze for $\ell \geq 2$. In LQG control, our finding is that it suffices to take $\ell = 1$. As mentioned in §I, MuZero also predicts optimal values and optimal actions; in LQG, to handle $Q^* \not\succ 0$, like cost-driven representation learning (see §III-A), we adopt the *cumulative costs* and use the normalized parameterization. Thus, the optimization problem we aim to solve is given by

$$\min_{M,A,B,b} \sum_{t=H}^{T+H-1} \big((\|Mh_t\|^2 + b - \bar{c}_t)^2 + (\|AMh_t + Bu_t\|^2 + b - \bar{c}_{t+1})^2\big). \tag{13}$$

To convexify the optimization problem (13), we define $N := M^\top M$ and $N' := [AM, B]^\top [AM, B]$. Then, (13) becomes

$$\min_{N,N_1,b} \sum_{t=H}^{T+H-1} \big((\|h_t\|_N^2 + b - \bar{c}_t)^2 + (\|[h_t; u_t]\|_{N_1}^2 + b - \bar{c}_{t+1})^2\big). \tag{14}$$

This minimization problem is convex in N, N_1 and b, and has a closed-form solution; essentially, it consists of two linear regression problems coupled by b. Since constant b is merely a term accounting for the estimation error and not part of the representation function, we can decouple the two regression problems by allowing b to take different values in them. This further simplifies the algorithm: the first regression problem is exactly cost-driven representation learning (§III-A), and the second is cost-driven system identification (CoSysId, Algorithm 2). The algorithm that instantiates Algorithm 1 with CoSysId is called CoReDyL (COst-driven state REpresentation and DYnamic Learning). Like CoReL, this MuZero-style latent model learning method provably solves LQG control, as we will show in Theorem 1.

CoSysId has similar steps to cost-driven representation learning (§III-A), except that in Line 5, it requires fitting a matrix $\hat{S}_0$. This is because the approximate factorization steps recover M^* and M_1^* up to orthonormal transformations, but there is no guarantee for the two orthonormal matrices to be the same; we need to fit $\hat{S}_0$ to align their coordinates. We note that although CoSysId needs the output $\hat{M}$ from cost-driven representation learning, the two quadratic regressions (11) and (15) are not coupled and can be solved in parallel.

Algorithm 2 CoSysId: Cost-driven system identification

1: **Input:** data $\mathscr{D}_{\text{raw}}$, representation function $\hat{M}$
2: Estimate the system dynamics by

$$\hat{N}_1, \hat{b}_1 \in \operatorname*{argmin}_{N_1 = N_1^\top, b_1} \sum_{t=H}^{T+H-1} \big(\|[h_t; u_t]\|_{N_1}^2 + b_1 - \bar{c}_{t+1}\big)^2 \tag{15}$$

3: Find $\hat{M}_1 \in \operatorname{argmin}_{M_1 \in \mathbb{R}^{d_x \times (Hd_y + (H+1)d_u)}} \|M_1^\top M_1 - \hat{N}_1\|_F$
4: Split $\hat{M}_1$ to $[\widetilde{M}, \widetilde{B}]$ after column $H(d_y + d_u)$ and set $\widetilde{A} = \widetilde{M}\hat{M}^\dagger$.
5: Find alignment matrix $\hat{S}_0$ by

$$\hat{S}_0 \in \operatorname*{argmin}_{S_0 \in \mathbb{R}^{d_x \times d_x}} \sum_{t=H}^{T+H-1} \|S_0 \hat{M}_1 [h_t; u_t] - \hat{z}_{t+1}\|^2$$

6: **Return:** system dynamics estimate $(\hat{A}, \hat{B}) = (\hat{S}_0 \widetilde{A}, \hat{S}_0 \widetilde{B})$

IV. Theoretical guarantees

The following Theorem 1 shows that both CoReL and CoReDyL are guaranteed to solve unknown LQG control with a finite number of samples.

Theorem 1: Given an unknown LQG problem satisfying Assumption 1, let $M^{*\prime}$ and $(A^{*\prime}, B^{*\prime}, Q^{*\prime}, R^*)$ be the optimal state representation function and the true system parameters under the normalized parameterization. For a given $p \in (0,1)$, if we run CoReL (Algorithm 1 with (12)) or CoReDyL (Algorithm 1 with Algorithm 2) for $T \geq \text{poly}(d_x, d_y, d_u, \log(T/p)))$, $H = \Omega(\log(H^2(d_y + d_u)T\log(T/p)))$, and $\sigma_u = \Theta(1)$, then there exists an orthonormal matrix $S \in \mathbb{R}^{d_x \times d_x}$, such that with probability at least $1-p$, the representation function $\hat{M}$ satisfies

$$\|\hat{M} - SM^{*\prime}\|_2 = \mathscr{O}(\text{poly}(H, d_x, d_u, d_y, \log(T/p))T^{-1/2}),$$

and the feedback gain $\hat{K}$ satisfies

$$J^{\hat{K}}(SA^{*\prime}S^\top, SB^{*\prime}, SQ^{*\prime}S^\top, R^*) - J^*(SA^{*\prime}S^\top, SB^{*\prime}, SQ^{*\prime}S^\top, R^*) = \mathscr{O}(\text{poly}(H, d_x, d_u, d_y, \log(T/p))T^{-1}).$$

Compared with common system identification methods based on learning Markov parameters [15, 22], the error bounds of the system parameters produced by CoReDyL (or CoReL) have the same dependence on T, but worse dependence on system dimensions. Moreover, to establish persistency of excitation, CoReDyL (or CoReL) requires a larger burn-in period. These relative sample inefficiencies are the price we pay for direct latent model learning, which is only supervised by *scalar-valued* costs that are quadratic in the history, instead of *vector-valued* observations that are linear in the history. Hence, we have to address the more challenging problem of *quadratic regression*, which lifts the dimension of the optimization problem. On the other hand, direct latent model learning avoids learning the reconstruction function C^* and can learn task-relevant representations in more complex settings, as demonstrated by empirical studies.

We refer the reader to the technical report for the full proof. Central to the analysis is the finite-sample characterization of the *quadratic regression* problem. To solve (11), notice that $\|h_t\|_N^2 = \langle \text{svec}(N), \text{svec}(h_t h_t^\top) \rangle$, so this quadratic regression is essentially a linear regression problem in terms of $\text{svec}(N)$. A major difficulty in the analysis is to establish persistency of excitation for $(\text{svec}(h_t h_t^\top))_{t \geq H}$, meaning that the minimum eigenvalue of the design matrix $\sum_{t=H}^{T+H-1} \text{svec}(h_t h_t^\top)\text{svec}(h_t h_t^\top)^\top$ grows linearly in the size T of the data. We overcome the difficulty using the small-ball method [21], originally developed for linear system identification.

Acknowledgements

This work was supported in part by NSF TRIPODS (DMS-2022448), NSF TILOS (CCF-2112665) and the Northrop Grumman – Maryland Seed Grant Program.

References

[1] Dimitri Bertsekas. *Dynamic Programming and Optimal Control: Volume I*, volume 1. Athena Scientific, 2012.

[2] Xiang Fu, Ge Yang, Pulkit Agrawal, and Tommi Jaakkola. Learning task informed abstractions. In *International Conference on Machine Learning*, pages 3480–3491. PMLR, 2021.

[3] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[4] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[5] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[6] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[7] Ali Jadbabaie, Horia Mania, Devavrat Shah, and Suvrit Sra. Time varying regression with hidden linear dynamics. *arXiv preprint arXiv:2112.14862*, 2021.

[8] N Komaroff. Iterative matrix bounds and computational solutions to the discrete algebraic Riccati equation. *IEEE Transactions on Automatic Control*, 39(8): 1676–1678, 1994.

[9] Sahin Lale, Kamyar Azizzadenesheli, Babak Hassibi, and Anima Anandkumar. Logarithmic regret bound in partially observable linear dynamical systems. *Advances in Neural Information Processing Systems*, 33: 20876–20888, 2020.

[10] Sahin Lale, Kamyar Azizzadenesheli, Babak Hassibi, and Anima Anandkumar. Adaptive control and regret minimization in linear quadratic Gaussian (LQG) setting. In *2021 American Control Conference (ACC)*, pages 2517–2522. IEEE, 2021.

[11] Alex Lamb, Riashat Islam, Yonathan Efroni, Aniket Didolkar, Dipendra Misra, Dylan Foster, Lekan Molu, Rajan Chari, Akshay Krishnamurthy, and John Langford. Guaranteed discovery of controllable latent states with multi-step inverse models. *arXiv preprint arXiv:2207.08229*, 2022.

[12] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[14] Junhyuk Oh, Satinder Singh, and Honglak Lee. Value prediction network. *Advances in neural information processing systems*, 30, 2017.

[15] Samet Oymak and Necmiye Ozay. Non-asymptotic identification of LTI systems from a single trajectory. In *2019 American control conference (ACC)*, pages 5655–5661. IEEE, 2019.

[16] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

[17] Kathrin Schacke. On the Kronecker product. *Master's Thesis, University of Waterloo*, 2004.

[18] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[19] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[20] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018.

[21] Max Simchowitz, Horia Mania, Stephen Tu, Michael I Jordan, and Benjamin Recht. Learning without mixing: Towards a sharp analysis of linear system identification. In *Conference On Learning Theory*, pages 439–473. PMLR, 2018.

[22] Max Simchowitz, Ross Boczar, and Benjamin Recht. Learning linear dynamical systems with semi-parametric least squares. In *Conference on Learning Theory*, pages 2714–2802. PMLR, 2019.

[23] Jayakumar Subramanian, Amit Sinha, Raihan Seraj, and Aditya Mahajan. Approximate information state for approximate planning and reinforcement learning in partially observed systems. *arXiv preprint arXiv:2010.08843*, 2020.

[24] Yi Tian, Kaiqing Zhang, Russ Tedrake, and Suvrit Sra. Can direct latent model learning solve linear quadratic Gaussian control? *arXiv preprint arXiv:2212.14511*, 2022.

[25] Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering Atari games with limited data. *Advances in Neural Information Processing Systems*, 34:25476–25488, 2021.

[26] Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *arXiv preprint arXiv:2006.10742*, 2020.