Controlgym: Large-Scale Control Environments for Benchmarking Reinforcement Learning Algorithms

Xiangyuan Zhang  
Weichao Mao  
ECE & CSL, University of Illinois Urbana-Champaign  
Saviz Mowlavi  
Mouhacine Benosman  
Mitsubishi Electric Research Laboratories  
Tamer Başar  
ECE & CSL, University of Illinois Urbana-Champaign

Editors: A. Abate, M. Cannon, K. Margellos, A. Papachristodoulou

Abstract

We introduce controlgym, a library of thirty-six industrial control settings, and ten infinite-dimensional partial differential equation (PDE)-based control problems. Integrated within the OpenAI Gym/Gymnasium (Gym) framework, controlgym allows direct applications of standard reinforcement learning (RL) algorithms like stable-baselines3. Our control environments complement those in Gym with continuous, unbounded action and observation spaces, motivated by real-world control applications. Moreover, the PDE control environments uniquely allow the users to extend the state dimensionality of the system to infinity while preserving the intrinsic dynamics. This feature is crucial for evaluating the scalability of RL algorithms for control. This project serves the learning for dynamics & control (L4DC) community, aiming to explore key questions: the convergence of RL algorithms in learning control policies; the stability and robustness issues of learning-based controllers; and the scalability of RL algorithms to high- and potentially infinite-dimensional systems. We open-source the controlgym project at https://github.com/xiangyuan-zhang/controlgym.

Keywords: Reinforcement learning, control, high-dimensional systems, PDE, benchmark.

Figure 1: Environments included in this work, motivated by industrial control applications

This manuscript is a shorter version of the technical report (Zhang et al., 2023b). We refer the readers to (Zhang et al., 2023b) for full details.

1. Introduction

The intersection of machine learning (ML), reinforcement learning (RL), and control theory has garnered significant attention in recent years, giving rise to the learning for dynamics & control (L4DC) research community (Recht, 2019; Vamvoudakis et al., 2021; Brunke et al., 2022; Hu et al., 2023). L4DC has the naturally driven mission to unlock the power of learning-based methods for control and establish a rigorous theoretical foundation (L4DC, 2023). This mission could only be fulfilled with joint forces and close collaboration between theorists and practitioners from ML, control theory, and optimization.

Theorists are keen to validate their algorithms and theories in real-world scenarios but encounter challenges with OpenAI Gym/Gymnasium (Gym) environments (Brockman et al., 2016; Towers et al., 2023). Specifically, most Gym environments feature highly nonlinear dynamics, often involving contacts, and offer very limited parameter customization options, making them ill-suited testbeds for control theory research. Meanwhile, control textbook examples lack the complexity for cutting-edge ML/RL research that prioritizes efficiency and scalability.

To address these requirements, we introduce controlgym, a lightweight and versatile Python library that offers a spectrum of environments spanning from linear systems to chaotic, large-scale systems governed by partial differential equations (PDEs). Specifically, controlgym features thirty-six linear industrial control environments, encompassing sectors like aerospace, cyber-physical systems, ground and underwater vehicles, and power systems. Additionally, controlgym includes ten large-scale control environments governed by fundamental PDEs in fluid dynamics and physics. These PDEs are discretized in space by custom solvers, yielding user-tunable state-space dimensions without affecting the dynamics of the environment, a key aspect for assessing the scalability of RL algorithms. All environments comply with Gym and support standard RL algorithms (Sutton et al., 2000; Kakade, 2002; Schulman et al., 2015, 2017; Mnih et al., 2016; Sutton and Barto, 2018), e.g., as seen in stable-baselines3 (Raffin et al., 2021).

Our primary contribution is the introduction of a diverse array of control environments characterized by continuous and unbounded action-observation spaces, designed for large-scale systems. These environments, detailed in Tables 1 and 2, enhance Gym’s collection and are highly customizable to support theoretical advancement in L4DC. For example, users can manipulate the open-loop dynamics of PDEs by adjusting physical parameters, with explicit formulas relating parameters and eigenvalues available in linear PDE environments. Moreover, our PDE environments uniquely allow the users to extend system dimensionality to infinity while preserving the intrinsic dynamics. The PDE solvers implemented to power controlgym are innovative, employing state-of-the-art schemes with exponential spatial convergence and high-order temporal accuracy masked behind a user-friendly discrete-time state-space formulation. Specifically for linear PDE environments, we have developed novel state-space models to evolve the PDE dynamics.

Leveraging its strengths, controlgym is a testbed for exploring three essential aspects of applying RL to continuous control. First, it aims to probe whether RL algorithms can consistently converge in learning control policies. Second, it examines the stability and robustness of the policy and training process, motivated by real-world safety-critical applications. Lastly, it assesses the scalability of RL algorithms in high-dimensional and potentially infinite-dimensional systems. With controlgym, we bridge the theoretical development and practical applicability of L4DC by providing a research platform that supports the establishment of a rigorous foundation. Initial deployments of controlgym include RL for PDE control (Zhang et al., 2024b; Botteghi and Fasel, 2023).
2024) and toward a foundational control transformer (Zhang et al., 2024a). We have provided code examples of controlgym in (Zhang et al., 2023b) and our GitHub repository.

1.1. Related Works

The COMPlibi project (Leibfritz, 2004; Leibfritz and Lipinski, 2004) pioneered in offering standard control tasks, as MATLAB files, for analyzing model-based control algorithms. In the era of ML and RL, Gym (Brockman et al., 2016; Towers et al., 2023) has become the standard platform for developing and benchmarking RL algorithms for continuous control, offering a variety of environments such as cart pole, inverted pendulum, and robotic tasks powered by Mujoco (Todorov et al., 2012). Numerous follow-up projects that implement RL algorithms on Gym environments include rllab/garage (Duan et al., 2016), RLlib (Liang et al., 2018), dm_control (Tunyasuvunakool et al., 2020), deluca (Gradu et al., 2020), stable-baselines3 (Raffin et al., 2021), safe-control-gym (Brunke et al., 2022), realworldrl-suite (Dulac-Arnold et al., 2020), tianshou (Weng et al., 2022), and TorchRL (Bou et al., 2023). Very recently, HydroGym (Callaham et al., 2023) provided fluid dynamics environments for testing RL algorithms for flow control.

Complying with Gym’s framework, we offer a spectrum of control environments designed to support the foundational theoretical developments in RL for linear optimal control (Fazel et al., 2018; Bu et al., 2019a; Tu and Recht, 2019; Mohammadi et al., 2021; Yang et al., 2019; Dean et al., 2020; Malik et al., 2020; Furieri et al., 2020; Simchowitz and Foster, 2020; Simchowitz et al., 2020; Hambly et al., 2021; Chen and Hazan, 2021; Perdomo et al., 2021; Li et al., 2021; Zhao and You, 2021; Jansch-Porto et al., 2022; Ozaslan et al., 2022; Ju et al., 2022; Lale et al., 2022; Duan et al., 2022; Zhang and Başar, 2023; Tang et al., 2023; Tsiamis et al., 2022; Ziemann et al., 2022; Duan et al., 2023), linear robust control and dynamic games (Agarwal et al., 2019; Gravell et al., 2019; Bu et al., 2019b; Zhang et al., 2019, 2021a; Yang et al., 2020; Zhang et al., 2021c, 2020, 2021b; Keivan et al., 2022; Guo and Hu, 2022; Cui et al., 2023), estimation and filtering (Umenberger et al., 2022; Zhang et al., 2023a,c,d), and PDE control (Pan et al., 2018; Bucci et al., 2019; Liu and Wang, 2021; Degrave et al., 2022; Zeng et al., 2022; Vignon et al., 2023; Mowlavi et al., 2023; Werner and Peitz, 2023; Peitz et al., 2023).

2. Control Environments

2.1. Linear Control Environments

We incorporate 36 linear control environments from various industries, as detailed in Table 1. We select and organize these continuous-time linear systems from the pioneering COMPlibi project (Leibfritz, 2004; Leibfritz and Lipinski, 2004), and provide them as standard Gym environments. These environments span control applications ranging from aircraft, helicopters, jet engines, reactor models, decentralized cyber-physical systems, binary distillation towers, ground and underwater autonomous vehicles, power systems, compact disk (CD) players, and large space structures. Additionally, the scope of our environments extends to control problems within projects such as the International Space Station and the Los Angeles Hospital.

With the user-selected sampling time $\Delta t$, we assume the control input is constant over each $\Delta t$ and generate the discrete-time system dynamics as

$$ s_{k+1} = A_s s_k + B_1 w_k + B_2 a_k, \quad z_k = C_1 s_k + D_{11} w_k + D_{12} a_k, \quad y_k = C s_k + D_{21} w_k, $$

where $A_s$, $B_1$, $B_2$, $C_1$, $D_{11}$, $D_{12}$, $C$, and $D_{21}$ are system matrices.
Table 1: List of the linear control environments in controlgym

<table>
<thead>
<tr>
<th>ID</th>
<th>(n_s)</th>
<th>(n_a)</th>
<th>(n_y)</th>
<th>Task</th>
<th>ID</th>
<th>(n_s)</th>
<th>(n_a)</th>
<th>(n_y)</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac1</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>aircraft</td>
<td>cm5</td>
<td>480</td>
<td>1</td>
<td>2</td>
<td>cable-mass model</td>
</tr>
<tr>
<td>ac2</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>aircraft</td>
<td>dis1</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td>decentralized system</td>
</tr>
<tr>
<td>ac3</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>aircraft</td>
<td>dis2</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>decentralized system</td>
</tr>
<tr>
<td>ac4</td>
<td>9</td>
<td>1</td>
<td>2</td>
<td>aircraft</td>
<td>dlr</td>
<td>40</td>
<td>2</td>
<td>2</td>
<td>space structure</td>
</tr>
<tr>
<td>ac5</td>
<td>9</td>
<td>1</td>
<td>5</td>
<td>aircraft</td>
<td>he1</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>helicopter</td>
</tr>
<tr>
<td>ac6</td>
<td>10</td>
<td>4</td>
<td>5</td>
<td>aircraft</td>
<td>he2</td>
<td>8</td>
<td>4</td>
<td>6</td>
<td>helicopter</td>
</tr>
<tr>
<td>ac7</td>
<td>55</td>
<td>2</td>
<td>2</td>
<td>aircraft</td>
<td>he3</td>
<td>8</td>
<td>4</td>
<td>6</td>
<td>helicopter</td>
</tr>
<tr>
<td>ac8</td>
<td>40</td>
<td>3</td>
<td>4</td>
<td>aircraft</td>
<td>he4</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>helicopter</td>
</tr>
<tr>
<td>ac9</td>
<td>40</td>
<td>3</td>
<td>4</td>
<td>aircraft</td>
<td>he5</td>
<td>20</td>
<td>4</td>
<td>6</td>
<td>helicopter</td>
</tr>
<tr>
<td>ac10</td>
<td>10</td>
<td>2</td>
<td>2</td>
<td>aircraft</td>
<td>he6</td>
<td>20</td>
<td>4</td>
<td>6</td>
<td>helicopter</td>
</tr>
<tr>
<td>bdt1</td>
<td>11</td>
<td>3</td>
<td>3</td>
<td>distillation tower</td>
<td>iss</td>
<td>270</td>
<td>3</td>
<td>3</td>
<td>International Space Station</td>
</tr>
<tr>
<td>bdt2</td>
<td>82</td>
<td>4</td>
<td>4</td>
<td>distillation tower</td>
<td>je1</td>
<td>30</td>
<td>3</td>
<td>5</td>
<td>jet engine</td>
</tr>
<tr>
<td>cm1</td>
<td>348</td>
<td>1</td>
<td>1</td>
<td>clamped beam model</td>
<td>je2</td>
<td>24</td>
<td>3</td>
<td>6</td>
<td>jet engine</td>
</tr>
<tr>
<td>cm2</td>
<td>120</td>
<td>2</td>
<td>2</td>
<td>CD player</td>
<td>lah</td>
<td>48</td>
<td>1</td>
<td>1</td>
<td>L.A. University Hospital</td>
</tr>
<tr>
<td>cm3</td>
<td>120</td>
<td>1</td>
<td>2</td>
<td>cable-mass model</td>
<td>pas</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>piezoelectric bimorph actuator</td>
</tr>
<tr>
<td>cm4</td>
<td>240</td>
<td>1</td>
<td>2</td>
<td>cable-mass model</td>
<td>psm</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>power system</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>rea</td>
<td>8</td>
<td>1</td>
<td>1</td>
<td>chemical reactor</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>umv</td>
<td>8</td>
<td>2</td>
<td>2</td>
<td>underwater vehicle</td>
</tr>
</tbody>
</table>

where \(s_k \in \mathbb{R}^{n_s}\) is the state, \(a_k\) is the control/action input, \(w_k\) is the disturbance input that could be either stochastic or adversarial, \(z_k \in \mathbb{R}^{n_y}\) is the observation, and \(A, B_1, B_2, C_1, C, D_{11}, D_{12}, D_{21}\) are the discretized system matrices with appropriate dimensions. These linear control environments are directly applicable to support theoretical research of RL for the fundamental linear control, game, and estimation tasks.

**Linear control objectives.** For each linear control task in Table 1, we define a regulation task whose primary objective is to steer the system’s dynamics toward the zero vector. The reward function (to be maximized) is formulated as the negative sum of the linear-quadratic (LQ) stage cost

\[
J(a_k) = -\mathbb{E}\left\{ \sum_{k=0}^{K-1} (s_k^T Q s_k + a_k^T R a_k + 2 s_k^T S a_k) \right\},
\]

where \(Q = C_1^T C_1, R = D_{12}^T D_{12}\), and \(S = C_1^T D_{12}\) aim to balance regulation performance and control efforts, and \(K\) is the total number of discrete time steps.

### 2.2. PDE Control Environments

In this section, we describe one-dimensional PDE control environments with periodic boundary conditions and spatially distributed control inputs. We first define a spatial domain \(\Omega = [0, L] \subset \mathbb{R}\) and a continuous field \(u(x, t) : \Omega \times \mathbb{R}^+ \to \mathbb{R}\), where \(x\) and \(t\) represent spatial and temporal coordinates, respectively, and \(L\) is the length of the domain. Each PDE control task listed in Table 2 then takes the general continuous form

\[
\frac{\partial u}{\partial t} - \mathcal{F}\left( \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}, \ldots \right) = a(x, t),
\]

where \(\mathcal{F}\) is a linear or nonlinear differential operator (see Sections 2.2.1-2.2.10 for specific definitions for each PDE) that contains spatial derivatives of various orders and depends on various...
an internal integration time step

2005

differencing Runge-Kutta (ETDRK4) scheme (using, respectively, a pseudo-spectral method (To compute the mapping each specific PDE and forcing support functions where nonlinear PDEs can be approximated by a discrete-time finite-dimensional nonlinear system Numerical solver for nonlinear PDEs. a discrete-time vector the total number of discrete-time steps. We also assume that the scalar control inputs time of \( t = t_n \). The total simulation time is \( K \Delta t \), where \( K \in \mathbb{N} \) is an input parameter specifying the total number of discrete-time steps. We also assume that the scalar control inputs \( a_i(t) \) are piecewise constant over each discrete-time step of duration \( \Delta t \) so that they can be concatenated into a discrete-time vector \( a_k \in \mathbb{R}^{n_a} \) for all \( k \in \{0, \cdots, K-1\} \).

Discretization of space and time. To solve the PDEs listed in Table 2, we first need to discretize space and time in the continuous form (2.1). For a state dimension \( n_s \) that is even and a sampling time \( \Delta t \in \mathbb{R}^+ \), both selected by the user, we define a state vector \( s_k \in \mathbb{R}^{n_s} \) that contains the values of \( u \) at \( n_s \) equally-spaced points in \( \Omega \) and at discrete time \( k \in \mathbb{N} \) corresponding to the simulation time \( t = k \Delta t \). The total simulation time is \( K \Delta t \), where \( K \in \mathbb{N} \) is an input parameter specifying the total number of discrete-time steps. We also assume that the scalar control inputs \( a_i(t) \) are piecewise constant over each discrete-time step of duration \( \Delta t \) so that they can be concatenated into a discrete-time vector \( a_k \in \mathbb{R}^{n_a} \) for all \( k \in \{0, \cdots, K-1\} \).

Numerical solver for nonlinear PDEs. After discretizing space and time, the dynamics of the nonlinear PDEs can be approximated by a discrete-time finite-dimensional nonlinear system

\[
s_{k+1} = f(s_k, a_k; w_k),
\]

where \( f : \mathbb{R}^{n_s} \times \mathbb{R}^{n_a} \rightarrow \mathbb{R}^{n_s} \) is a time-invariant mapping contingent on the physical parameters of each specific PDE and forcing support functions \( \Phi_i \), and \( w_k \) is an optional stochastic process noise. To compute the mapping \( f \), we numerically approximate the space and time derivatives in (2.1) using, respectively, a pseudo-spectral method (Trefethen, 1996) and a fourth-order exponential time differencing Runge-Kutta (ETDRK4) scheme (Cox and Matthews, 2002; Kassam and Trefethen, 2005). We then integrate numerically the dynamics over one discrete-time step of duration \( \Delta t \) using an internal integration time step \( dt \). The mapping \( f \) is therefore not obtained explicitly; rather, its

<table>
<thead>
<tr>
<th>ID</th>
<th>( n_s )</th>
<th>( n_a )</th>
<th>( n_y )</th>
<th>Linearity</th>
</tr>
</thead>
<tbody>
<tr>
<td>convection_diffusion_reaction</td>
<td>(50, +\infty)</td>
<td>(1, +\infty)</td>
<td>(1, +\infty)</td>
<td>Linear</td>
</tr>
<tr>
<td>wave</td>
<td>(100, +\infty)</td>
<td>(1, +\infty)</td>
<td>(1, +\infty)</td>
<td>Linear</td>
</tr>
<tr>
<td>schrodinger</td>
<td>(100, +\infty)</td>
<td>(1, +\infty)</td>
<td>(1, +\infty)</td>
<td>Linear</td>
</tr>
<tr>
<td>burgers</td>
<td>(50, +\infty)</td>
<td>(1, +\infty)</td>
<td>(1, +\infty)</td>
<td>Linear</td>
</tr>
<tr>
<td>kuramoto_sivashinsky</td>
<td>(200, +\infty)</td>
<td>(1, +\infty)</td>
<td>(1, +\infty)</td>
<td>Nonlinear</td>
</tr>
<tr>
<td>fisher</td>
<td>(50, +\infty)</td>
<td>(1, +\infty)</td>
<td>(1, +\infty)</td>
<td>Nonlinear</td>
</tr>
<tr>
<td>allen_cahn</td>
<td>(50, +\infty)</td>
<td>(1, +\infty)</td>
<td>(1, +\infty)</td>
<td>Nonlinear</td>
</tr>
<tr>
<td>korteweg_de_vries</td>
<td>(200, +\infty)</td>
<td>(1, +\infty)</td>
<td>(1, +\infty)</td>
<td>Nonlinear</td>
</tr>
<tr>
<td>cahn_hilliard</td>
<td>(50, +\infty)</td>
<td>(1, +\infty)</td>
<td>(1, +\infty)</td>
<td>Nonlinear</td>
</tr>
<tr>
<td>ginzburg_landau</td>
<td>(50, +\infty)</td>
<td>(1, +\infty)</td>
<td>(1, +\infty)</td>
<td>Nonlinear</td>
</tr>
</tbody>
</table>

Table 2: List of PDE control environments in controlgym

physical constants, and \( a \) is a distributed control force defined as

\[
a(x, t) = \sum_{j=0}^{n_a-1} \Phi_j(x)a_j(t). \tag{2.2}
\]

The control force consists of \( n_a \) scalar control inputs \( a_j(t) \), each acting over a specific subset of \( \Omega \) defined by its corresponding forcing support function \( \Phi_j(x) \). Such a control force can be used to model the addition of energy to the system or other external influences that affect the PDE dynamics. We use periodic boundary conditions in all of our PDE control tasks, meaning that spatial derivatives are equal at both ends of the domain \( \Omega \).
action is evaluated through a numerical integration loop. The sampling time $\Delta t$ may be selected as large as desired, but it should be an integer multiple of the integration time step $dt$.

In general, one should choose $n_s$ to be sufficiently large and $dt$ to be sufficiently small to ensure the accuracy of the discretized system (2.3). Due to the exponential convergence rate of the pseudospectral method as well as the high convergence rate of the ETDRK4 scheme, $n_s$ larger than about 50 is sufficient in most cases, except the Kortweg de Vries and Kuramoto-Sivashinsky PDEs that require $n_s$ larger than about 200 for accurate solutions. For all PDEs, the presence of small-scale (i.e., high wavenumber) spatial features in the initial condition may necessitate higher values of $n_s$.

**Explicit state-space model for linear PDEs.** After space and time discretization, the linear PDEs listed in Table 2 can be approximated by a discrete-time linear state-space model of the form

$$s_{k+1} = As_k + B_2a_k + w_k,$$

where $A \in \mathbb{R}^{n_s \times n_s}$ is a time-invariant transition matrix contingent on the physical parameters of each specific PDE, $B_2 \in \mathbb{R}^{n_s \times n_u}$ is a time-invariant control matrix, and $w_k \sim \mathcal{N}(0, \Sigma_w)$ is an optional process noise. The $A$ matrix in (2.4) is constructed from a spectral approximation of the space derivatives in (2.1) combined with an analytical temporal integration of the continuous-time linear dynamics over one discrete-time step of duration $\Delta t$ (see (Zhang et al., 2023b) for detailed treatments of each case). Contrary to the case of nonlinear PDEs where evaluating the mapping $f$ in (2.3) requires an internal numerical integration loop, the availability of matrix $A$ in explicit form for linear PDEs allows for the direct application of model-based linear controllers.

Similar to nonlinear PDEs, due to the numerical approximation of the spatial derivatives, one should choose $n_s$ to be sufficiently large to ensure the accuracy of the state-space model (2.4), with $n_s$ greater than about 50 sufficient in most cases. Due to the analytical temporal integration of the dynamics, there is no internal integration time step $dt$ to select. As in the nonlinear case, the sampling time $\Delta t$ may be chosen as large as desired.

Since the state-space model (2.4) is derived from a PDE, the eigenvalues and eigenvectors of the $A$ matrix can be analyzed explicitly. This method allows for a clearer understanding of the impact of the PDE’s physical parameters on the system dynamics, such as open-loop stability. By adjusting these parameters, users can tailor the system dynamics to better assess their algorithms; we refer readers to (Zhang et al., 2023b) for full details.

**Observation process.** For all PDEs listed in Table 2, we place $n_y$ sensors uniformly throughout the domain $\Omega$, where each sensor measures the unscaled value of the state at its location, perturbed by additive zero-mean Gaussian white noise. That is, the observation $y_k$ at time $k$ is computed by $y_k = C s_k + v_k$, where $C \in \mathbb{R}^{n_y \times n_s}$ is structured with a single 1 per row and zeros elsewhere, and $v_k \sim \mathcal{N}(0, \Sigma_v)$. Both $n_y$ and $\Sigma_v$ are user-configurable parameters.

**PDE control objectives.** For all PDEs listed in Table 2, we define a control task whose primary objective is to steer the system’s dynamics toward a user-defined target state $s_{ref} \in \mathbb{R}^{n_s}$. The reward function is formulated as the negative sum of the LQ stage cost

$$\mathcal{J}(a_k) = -\mathbb{E} \left\{ \sum_{k=0}^{K-1} \left[ (s_k - s_{ref})^\top Q (s_k - s_{ref}) + a_k^\top R a_k \right] \right\},$$

where $Q$ and $R$ are positive-definite weighting matrices that balance tracking performance and control effort. When the target state is the zero vector, the tracking problem reduces to the LQ regulation problem.
2.2.1. Convection-Diffusion-Reaction Equation

The convection-diffusion-reaction (CDR) equation models the transfer of particles, energy, or other physical quantities within a system due to convection, diffusion, and reaction processes. The temporal dynamics of the continuous concentration function \( u(x, t) \) in one spatial dimension is given by

\[
\frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} - \nu \frac{\partial^2 u}{\partial x^2} - r u = a(x, t),
\]

(2.5)

where \( c \) is the convection velocity, \( \nu > 0 \) is the diffusivity constant, \( r \) is the reaction constant, and \( a(x, t) \) is a source term defined in (2.2) that models the addition of energy to the system or other external influences that affect the PDE dynamics. The scalar physical parameters of the CDR equation characterize the strength of convection, diffusion, and reaction processes. When \( c = r = 0 \), the CDR equation (2.5) reduces to the heat equation. The CDR equation with \( r = 0 \) has been used to validate the global convergence of RL algorithms in Kalman filtering (Zhang et al., 2023c).

2.2.2. Wave Equation

The wave equation is a fundamental linear PDE in physics and engineering, describing the propagation of various types of waves through a homogeneous medium. The temporal dynamics of the perturbed scalar quantity \( u(x, t) \) propagating as a wave through one-dimensional space is given by

\[
\frac{\partial^2 u}{\partial t^2} - c^2 \frac{\partial^2 u}{\partial x^2} = a(x, t),
\]

(2.6)

where \( c \) is a constant representing the wave’s speed in the medium, and \( a(x, t) \) is a source term defined in (2.2) that models the effect of a force or other external influences acting on the system.

2.2.3. Schrödinger Equation

The Schrödinger equation is fundamental in quantum mechanics, describing how the quantum state of an isolated quantum-mechanical system, a complex-valued wave function, changes over time. For a single non-relativistic particle in a constant potential, the Schrödinger equation for the wave function \( u(x, t) \) is given by

\[
i \hbar \frac{\partial u}{\partial t} + \frac{\hbar^2}{2m} \frac{\partial^2 u}{\partial x^2} - V u = \hbar a(x, t),
\]

(2.7)

where \( i \) is the imaginary unit, \( \hbar \) is the Planck constant, \( V \) is the real potential constant, and \( a(x, t) \) is a real-valued source term defined in (2.2) that models a force acting on the system or other external influences that affect the PDE dynamics.

2.2.4. Burgers’ Equation

Burgers’ equation is a simplified version of nonlinear PDEs arising in fluid dynamics and captures key features of water waves and gas dynamics such as shock formation. The temporal dynamics of the velocity \( u(x, t) \) is

\[
\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} - \nu \frac{\partial^2 u}{\partial x^2} = a(x, t),
\]

(2.8)
where \( \nu > 0 \) is the diffusivity (or viscosity) parameter and \( a(x, t) \) is a source term defined in (2.2) that models a force acting on the system or other external influences that affect the PDE dynamics. At the inviscid limit of \( \nu = 0 \), Burgers’ equation predicts discontinuous shocks; at low \( \nu \) values, Burgers’ equation exhibits shock-like behavior but remains smooth; and with high values of \( \nu \), Burgers’ equation mirrors the dissipative nature of the heat equation.

### 2.2.5. Kuramoto-Sivashinsky Equation

The Kuramoto-Sivashinsky (KS) equation is a nonlinear PDE applied to studying pattern formation and instability in fluid dynamics, combustion, and plasma physics. The temporal dynamics of \( u(x, t) \) in one spatial dimension is provided by

\[
\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + \frac{\partial^2 u}{\partial x^2} + \frac{\partial^4 u}{\partial x^4} = a(x, t),
\]

(2.9)

where \( a(x, t) \) is a source term defined in (2.2) that models a force acting on the system or other external influences that affect the dynamics. The nonlinear convection term, the second-order diffusion term, and the fourth-order dispersion term interact to produce complex spatial patterns and temporal chaos when the domain length \( L \) is large enough (Cvitanović et al., 2010). Due to the chaotic dynamics, the specific choice of initial condition has negligible influence on the qualitative properties of the solution.

### 2.2.6. Fisher Equation

The Fisher equation is a nonlinear PDE employed in biology, ecology, and epidemiology to model gene propagation, invasions, and population dynamics. The temporal dynamics of \( u(x, t) \) in one spatial dimension is described by

\[
\frac{\partial u}{\partial t} - \nu \frac{\partial^2 u}{\partial x^2} - r \cdot u(1 - u) = a(x, t),
\]

(2.10)

where \( \nu > 0 \) is the diffusivity constant, \( r \) is the reaction constant, and \( a(x, t) \) is a source term defined in (2.2) that models external influences affecting the PDE dynamics. The term \( r \cdot u(1 - u) \) captures population expansion limited by carrying capacity, with \( r \) as the intrinsic growth rate.

### 2.2.7. Allen-Cahn Equation

The Allen-Cahn equation is a nonlinear PDE modeling phase separation in binary alloy systems in materials science. The temporal dynamics of \( u(x, t) \) in one spatial dimension, with \( u = \pm 1 \) indicating the presence of one phase or the other, is given by

\[
\frac{\partial u}{\partial t} - \nu^2 \frac{\partial^2 u}{\partial x^2} + V(u^3 - u) = a(x, t),
\]

(2.11)

where \( \nu > 0 \) is the diffusivity constant, \( V \) is the potential constant, and \( a(x, t) \) is a source term defined in (2.2) that models external influences affecting the PDE dynamics.
2.2.8. **Korteweg-de Vries Equation**

The Korteweg-de Vries (KdV) equation is a nonlinear PDE pivotal in understanding nonlinear wave dynamics, modeling solitary wave propagation across shallow water surfaces, with applications extending to plasma physics, nonlinear optics, and quantum mechanics. The temporal dynamics of $u(x,t)$ with an additional source term $a(x,t)$ that models external influences is given by

$$\frac{\partial u}{\partial t} + \frac{\partial^3 u}{\partial x^3} - 6u \frac{\partial u}{\partial x} = a(x,t).$$

(2.12)

2.2.9. **Cahn-Hilliard Equation**

The Cahn-Hilliard equation is a nonlinear PDE modeling phase separation in alloys and polymers in materials science. The temporal dynamics of $u(x,t)$ in one spatial dimension, with $u = \pm 1$ indicating the presence of one phase or the other, is described by

$$\frac{\partial u}{\partial t} - \nu \frac{\partial^2}{\partial x^2} \left( u^3 - u - \Gamma \frac{\partial^2 u}{\partial x^2} \right) = a(x,t),$$

(2.13)

where $\nu > 0$ is the diffusivity constant, $\Gamma$ is the constant surface tensor coefficient, and $a(x,t)$ is a source term defined in (2.2) that models external influences affecting the PDE dynamics.

2.2.10. **Ginzburg-Landau Equation**

The Ginzburg-Landau equation is a nonlinear PDE describing the evolution of disturbances near the onset of instability in various physical systems. The temporal dynamics of the amplitude $u(x,t)$ of a disturbance in one spatial dimension is governed by

$$\frac{\partial u}{\partial t} - u + |u|^2u - \frac{\partial^2 u}{\partial x^2} = a(x,t),$$

(2.14)

where $a(x,t)$ is a source term defined in (2.2) that models external influences affecting the PDE dynamics.

### 3. Conclusion

We have presented controlgym, a library designed to support the research efforts of L4DC. The controlgym project facilitates a deeper investigation into the performance of RL algorithms, particularly focusing on their convergence, the stability and robustness of RL-based controllers, and the scalability of RL algorithms to systems with high and infinite state dimensionality.

### Acknowledgments

The research of XZ, WM, and TB were supported in part by the US Army Research Laboratory (ARL) Cooperative Agreement W911NF-17-2-0181, in part by the Army Research Office (ARO) MURI Grant AG285, and in part by the ARO Grant W911NF-24-1-0085. SM and MB were supported solely by MERL.
References


