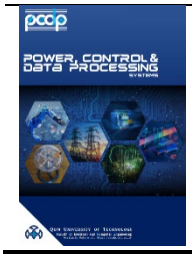
DOI: 10.30511/pcdp.2025.2065658.1034 - ISSN: 3060-8376 - Vol 3(1), 2026



POWER, CONTROL AND DATA PROCESSING SYSTEMS

Available Online at: https://pcdp.qut.ac.ir/

The Formulation of a Linear Quadratic Regulator (LQR) for a System Characterized by Partial Differential Equations (PDE) Utilizing Reinforcement Learning Technique

ARTICLE INFO

Article Type
Original Research

Authors
M. Rezvanitabar[1]
J. Sharifi[2]
M. Yadegar[3]

[1] Department of Electrical and computer engineering, Qom University of Technology, Qom, Iran, mrezvani44@gmail.com
[2] Department of electrical and computer engineering, Qom University of Technology, Qom, Iran, jv.sharifi@gmail.com
[3] Department of electrical and computer engineering, Qom University of Technology, Qom, Iran, yadegar@qut.ac.ir

**** Correspondence***
Address: Department of electrical and computer engineering, Qom University of Technology, Khodakaram Boulevard, Qom, Iran. Postal Code: 71551313.
Phone: +98-25-36169731
jv.sharifi@gmail.com

Article History
Received: July 12, 2025
Accepted: December 02, 2025
ePublished: March 01, 2026

ABSTRACT

Reinforcement learning has emerged as a valuable tool in control theory and related areas, such as robotics and process control, facilitating prediction, identification, and management of complex systems. A key advantage of using reinforcement learning is its ability to derive optimal control policies without requiring a comprehensive understanding of the underlying system dynamics. In essence, RL develops control strategies through interaction with the environment. A common approach in controller design for these systems involves the use of approximation methods. Given the complexities associated with solving PDEs analytically, numerical techniques like the finite element method are typically employed for approximation. In this research, we begin by discretizing the PDE that characterizes the system's dynamics using an appropriate discretization technique. Following this step, we extract the discrete dynamics of the system. Since these discrete dynamics exhibit the Markov property where the future state depends only on the current state and the action taken the next phase involves designing a controller based on these derived dynamics. Recognizing that reinforcement learning focuses on optimizing future actions through data analysis, feedback from the optimal mode can be utilized as a viable option for controller design. This study will further explore LQR controller design for one-dimensional heat equation heat flow as a system whose dynamic is described with PDE.

Keywords: Partial Differential Equation (PDE); Heat Flow; Reinforcement learning; Q-learning; Controller; Linear Quadratic Regulator; Model free, Finite Element Method (FEM)

1 Introduction

The Linear Quadratic Regulator (LQR) represents a crucial control methodology extensively utilized in the fields of engineering and applied mathematics. The standard approach to its implementation generally requires solving the Algebraic Riccati Equation (ARE) to ascertain the optimal feedback gains. The design of a Linear Quadratic Regulator (LQR) for systems governed by partial differential equations (PDEs) is an attractive field of research in control theory. Merging classical optimal control principles with contemporary machine learning techniques adds to the attractiveness of this area. The LQR framework is well-established for finite-dimensional systems, allowing for the minimization of a quadratic cost function while ensuring system stability. However, extending LQR methodologies to infinite-dimensional systems characterized by PDEs presents unique challenges due to the complexity of their dynamics and the necessity for precise feedback control across spatial domains. Recent advancements in reinforcement learning (RL) offer promising solutions to these challenges. Reinforcement learning is characterized by learning through interaction with an environment. In this approach, the learner engages with a dynamic environment that provides feedback based on their actions. This feedback typically comes in the form of rewards or penalties, which guide the learner toward optimal behavior over time. In reinforcement learning, an agent interacts with its environment at discrete time steps. At each time step t, the agent observes the current state of the environment s_t, selects an action a_t, and receives a reward r_t based on its action and the resulting new state s_{t+1}. The agent's objective is to maximize its cumulative reward over time by developing a policy—a strategy that dictates which actions to take in various states.

When designing controllers using reinforcement learning, interaction with this environment becomes crucial. It is posited that understanding the environment's specifics may not be essential; rather, what matters is how effectively an agent learns from its interactions. The learning algorithm applies an action (control signal) to the environment (plant) through an agent (controller) at a given time t. This control signal alters the state of the environment, which then responds by providing a reward based on the new state. Using this new state and reward information, the agent generates subsequent control signals. The agent's objective is to produce signals that optimize (minimize) rewards according to the target problem. Rewards are defined as scalar functions of state and action that indicate how beneficial or detrimental a particular action is in a given state. These scalar functions represent step costs—the cost incurred to transition from state x_k to state x_{k+1} using action u_k.

RL enables adaptive learning of optimal control policies through interaction with the system, making it particularly suitable for real-time applications where system dynamics may be uncertain or partially known. By integrating RL with LQR design, researchers aim to enhance controller performance and robustness, facilitating effective regulation of complex systems governed by PDEs. This integration not only addresses the limitations of traditional LQR approaches but also opens new avenues for research in optimal control.

In the context of the importance of the Subject, it could be stated that the significance of designing LQR controllers for PDE-governed systems is underscored by their widespread applicability across various engineering disciplines, including fluid dynamics, thermal management, and structural control. Effective regulation of such systems is crucial for optimizing performance and ensuring safety in critical applications. The ability to apply RL methods within this context enhances the potential for developing adaptive controllers that can respond to dynamic changes in system behavior, thereby improving overall system efficiency and reliability.

The motivation behind combining LQR with reinforcement learning stems from the limitations encountered in traditional control approaches. Classical LQR requires complete knowledge of system dynamics and often struggles with real-time adaptability. In contrast, RL provides a framework that learns from experience, allowing for the development of control strategies that can adapt to changing conditions without requiring full prior knowledge of the system dynamics [1]. This capability is particularly valuable in environments where model uncertainties are prevalent.

Previous research has laid a solid foundation for understanding the integration of RL with LQR methodologies. Notably, studies have demonstrated that RL can effectively solve LQR problems without requiring explicit solutions to the Algebraic Riccati Equation (ARE), which is traditionally essential in LQR design [2]. For instance, Bradtke's work highlighted the application of dynamic programming-based RL algorithms to LQR problems, emphasizing their potential to bridge theoretical gaps in continuous state and action spaces [3].

Furthermore, recent advancements have explored model-free approaches that utilize RL techniques to derive optimal feedback controls directly from observed data [1]. In addition, various studies have successfully implemented RL strategies in linear time-invariant (LTI) systems, showcasing their effectiveness in optimizing controller performance under noisy conditions and time-varying parameters [4]. These experiences underline the practicality and versatility of combining RL with traditional control frameworks.

In the context of the contributions, it could be stated that this research article aims to provide a comprehensive framework for designing LQR controllers using reinforcement learning techniques specifically tailored for systems governed by PDEs. By addressing these objectives, this work aspires to contribute valuable insights into both optimal control theory and machine learning applications in engineering. The proposed approach will leverage reduced-order modeling techniques alongside iterative learning strategies to enhance computational efficiency while maintaining controller performance. Furthermore, this research will explore the implications of transferring learned policies across different PDE scenarios, thereby broadening the applicability of the developed

methodologies [4]. By investigating these aspects, this study seeks to pave the way for future innovations in adaptive control systems that can effectively manage complex dynamic behaviors inherent in PDE-governed environments. In conclusion, the intersection of linear quadratic regulation and reinforcement learning offers a fertile ground for advancing control methodologies applicable to complex dynamic systems governed by partial differential equations. As research continues to evolve, this integrated approach promises significant improvements in controller design and implementation across various engineering applications.

The integration of machine learning algorithms into systems governed by PDEs raises critical questions: How can we effectively apply these algorithms in such contexts? What contributions can machine learning methods like reinforcement learning make to address these challenges?

In this paper we introduce a method for designing a controller for a PDE system by the RL. In this regard first a PDE system will be transferred to the ODE by discretization. Since the result of discretization is so closed to Markov decision process (MDP), it is possible to define the problem to the RL. A one-dimensional heat equation has been chosen as a case study to effectively illustrate and communicate the principles of controller design utilizing the reinforcement learning approach, and the controller design has been implemented for this system.

2 Literature Review

LQR seeks to reduce a quadratic cost function while adhering to linear dynamic systems. The conventional approach utilizes state-space representation, wherein the dynamics of the system are articulated through linear equations. Nevertheless, numerous real-world systems are influenced by partial differential equations (PDEs), which present challenges that conventional LQR methodologies struggle to address effectively. Recent research has investigated model-based strategies and reinforcement learning techniques to broaden the applicability of LQR to systems governed by PDEs.

In the study [5], the researchers addressed the problem of boundary control for partial differential equations (PDEs) through the application of operator learning techniques. The study [6] focuses on the reduction of PDE system models and the design of stabilizers utilizing reinforcement learning techniques. In a paper, the researchers employed a data-driven approach utilizing reinforcement learning to manage partial differential equations (PDEs) [7]. In the context of predictive control design for partial differential equation (PDE) systems, one can reference applied research [8]. The researchers of [9] developed a predictive control framework utilizing machine learning techniques for systems governed by nonlinear parabolic partial differential equations. In [10], their research examined the development of Economic Model Predictive Control (EMPC) tailored for systems characterized by hyperbolic and parabolic partial differential equations, which exhibit static spatial distribution properties. In relation to alternative approaches to controller design, including the backward step method, it is important to reference the [11]. The study [12] is notable for its contributions to the adaptive control of systems characterized by partial differential equations (PDEs). Researchers in [13] have made significant progress in applied research related to oil well drilling by utilizing backstep infinite-dimensional transformations and Lyapunov functions in the context of partial differential equation (PDE) systems. In the domain of fuzzy control, the study [14] is noteworthy. They also developed a nonlinear ordinary differential equation model to describe the dynamics of the PDE system, using the modal decomposition technique, and expressed it through a Takagi-Sugeno fuzzy model.

A novel dynamic neural programming control approach was introduced in reference [15], resulting in a subsystem of ordinary differential equations that capture the underlying dynamics of the PDE system. [16] examined and compared reinforcement learning with backward learning, with a key focus on creating strategies to reduce traffic congestion on highways. Traffic dynamics on highways are modelled using two partial differential equations, which describe traffic density and speed, and reformulate control as a reinforcement learning issue, requiring stabilization through observation of system state values without prior knowledge of the system's movement patterns. An optimization of policy is accomplished by using a neural network-based algorithm in conjunction with a numerical simulator that solves partial differential equations. The controller's performance was evaluated by comparing it in two different operating conditions: one with a thorough understanding of traffic flow dynamics and another with limited information, to assess the recovery capabilities of backward, proportional, and proportional integral control methods. The focus of [17] was on managing dynamic systems that are determined by partial differential equations through distributed reinforcement learning techniques. A key aspect of this investigation involves reducing the environmental dimensions for agents by applying a convolution operation to the state space, thus resolving the dimensionality challenges encountered in deep reinforcement learning. In the [18], a stochastic partial differential equation control issue as a reinforcement learning challenge is characterized by utilizing a distributed control due to the system's infinite domain. This proposed RL control method can be readily adapted to boundary control techniques for managing finite systems. A model-based reinforcement learning approach has been formulated in [19], concentrating on affine control systems characterized by quadratic objective functions and constrained horizon optimal control challenges involving variable reference trajectories. To achieve this, deep neural networks are employed. The methodology is exemplified through applications such as a chemical reactor and a one-dimensional diffusion-convection-reaction system, which serve to highlight the fundamental features of the proposed technique. The [20] focused on the design of control mechanisms and the experimental validation of a flexible robotic system. It introduced a control strategy grounded in reinforcement

learning, utilizing an actor-critic framework. This approach was meticulously crafted to minimize vibrations while ensuring accurate trajectory tracking. Subsequent findings demonstrated that the closed-loop system, employing the proposed reinforcement learning control algorithm through the direct Lyapunov method, exhibits semi-universal finiteness. To validate the effectiveness of the proposed control strategy, a series of experiments were conducted on a laboratory platform, allowing for a comparative analysis with traditional PD control methods. An optimal control approach utilizing adaptive reinforcement learning for collaborative robotic systems is introduced in [21]. This method combines hybrid motion and force control strategies, leveraging adaptive reinforcement learning principles. The implementation involves a simultaneous actor-critic algorithm alongside Bellman optimization, which is derived from the discrepancies in the control policy and the ideal control input. The effectiveness of the adaptive reinforcement learning-based optimal control strategy in closed-loop systems is confirmed through the application of a proposed Lyapunov candidate function. Ultimately, simulation results are provided for a constrained system involving three robots, demonstrating the practical application of the optimal hybrid motion and force control strategy.

some approaches for control of PDE systems have been developed over the past decades such as the linear quadratic optimal control [22, 23], model predictive control [24, 25], geometric control [26], robust control with spatiotemporal norm bounded uncertainties and Markov jumping parameters [27], and sliding-mode control [28, 29].

It can be inferred from the previously discussed literature on analogous studies that there exists a scarcity of research focused on the utilization of reinforcement learning as a model-free methodology in systems governed by partial differential equations, especially in relation to the discretization of such equations.

3 Conceptual Subjects

To establish a theoretical framework and foundational concepts, this section discusses essential definitions, research hypotheses, and mathematical topics pertinent to the research area. The research background will highlight the innovation of the current study and its position within existing literature, as well as identify gaps related to the overarching research theme. The definitions and mathematical concepts associated with the topic are divided into two sections. The first section discusses the theoretical principles related to partial differential equations and their significance in control theory. The second section provides a selective overview of the reinforcement learning literature. Given the extensive theoretical discussions surrounding reinforcement learning and partial differential equations, we aim to focus solely on the aspects that are relevant to other sections of this research, thereby enhancing the thematic and conceptual connections with both the literature and mathematics essential for this study.

This research diverges from complex mathematical structures, prioritizing the design of controllers utilizing Q-learning techniques for a specified system. A notable characteristic of this investigation, in contrast to similar studies in the field, is this particular focus.

3.1 Discretization of Partial Differential Equations (PDE)

Numerous physical phenomena, including fluid dynamics, electricity, magnetism, mechanics, and optics, can be mathematically represented and modeled using partial differential equations. Many fundamental laws of physics, such as Maxwell's equations, Newton's law of cooling, the Navier-Stokes equations, and Schrödinger's equations, are described as partial differential equations. The infinite-dimensional nature of partial differential equation (PDE) systems makes it challenging to directly apply conventional control methods associated with ordinary differential equation (ODE) systems for controller design in PDE contexts. If we can transform a system described by a PDE into one represented by an ODE, we can then utilize the control strategies available in ODE theory. This raises the question: what tools can we use to convert a PDE system into an ODE system? What mathematical relationships facilitate this transformation? The answer lies in the discretization of PDE equations. Common methods for converting partial differential equations into ordinary differential equations include:

- Finite Element Method
- Finite Difference Method
- Finite Volume Method

Thus, we can convert the desired PDE equation into an ODE equation using one of these techniques and subsequently design a controller for the ODE system, which serves as an approximation of the original PDE system. In this study, the finite element method was employed as the tool for conversion from PDE to ODE.

3.2 Reinforcement Learning and Elements

Reinforcement learning is a type of machine learning that has emerged from the fields of artificial intelligence, computer science, and engineering. It is closely associated with optimal and adaptive control [30]. Reinforcement learning provides a framework to address optimal control problems which are described as Markov decision processes. Reinforcement learning comprises several key elements, each of which is essential for developing and expressing the reinforcement learning problem. Without these components, interpreting and solving a problem becomes impossible. Here, we define each component.

Environment: In reinforcement learning, the environment is defined as described in Eq 1.

$$P_{x,x'}^{u,r} = \Pr(x', r|x, u) \quad \text{Eq 1}$$

The relationship between the inputs and outputs is represented in the continuous interval [0,1] as follows:

$$P: x \times u \times x \to [0,1] \quad \text{Eq 2}$$

This relationship provides a probabilistic description of the likelihood of being in state x, taking action u, receiving reward r, and transitioning to state x'.

Agent: From an artificial intelligence perspective, an agent is any entity that can perceive its environment through sensors and take actions based on stimuli

Reward: The reward function establishes the objective in a reinforcement learning scenario. Essentially, it assigns a scalar value to each observed state of the environment, indicating how desirable it is to be in that state.

Policy: The policy outlines the actions taken by the agent at any given moment. Essentially, it serves as a mapping between the states of the environment and possible actions that can be taken in those states. Typically, the policy is represented as a stochastic function, denoted by Eq 3, which indicates the probability of selecting action u when in state x.

$$\pi(x, u) = Pr\ (u|x) \quad \text{Eq 3}$$

If the policy is deterministic and not random:

$$\mu(x): x \to u \quad \text{Eq 4}$$

u is a function of x, which represents the action that taken in each situation.

Therefore, if the policy is clear-cut, then the relationship represented by Eq 4 will become Eq 5, which is a scalar value.

$$\pi(x, u) = \mu(x) \quad \text{Eq 5}$$

Markov decision process (MDP):

In the MDP framework, the action taken at any given time impacts both the reward obtained in the following moment and the subsequent state that results. MDP is generally represented by a set of four elements denoted as (X, U, P, R), where:

The set X includes all possible states in the environment, U consists of the available actions, P accounts for the system's dynamics, and R denotes the rewards or penalties related to these actions. Stochastic optimal control is also an area of study for MDPs, and within this field, adaptive optimal control methods have a close connection to reinforcement learning [31].

Value functions: In contrast to a reward function, which focuses on what is advantageous in the current moment, a value function considers what is ultimately beneficial over a prolonged time. The value of a state essentially represents the total anticipated rewards that an agent can acquire beginning from that state in the future.

Policy evaluation: The process involves evaluating the value of states following a specific policy framework.

Policy improvement: Involves creating a more efficient policy that is based on core principles. The process entails moving from a basic policy, identified as, to a more refined policy denoted by π'.

Agent-Environment Interaction: a specific time point t; the agent selects action A as the control signal for that instant. When action A is applied to the environment, it shifts to a new state. Following this, a reward R is generated and then provided to the agent.

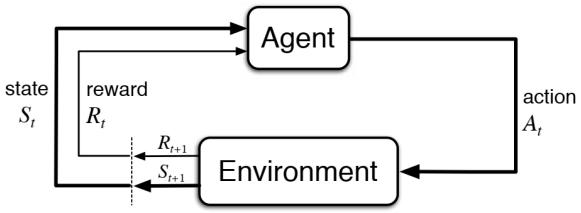


Figure 1: The agent–environment interaction in a Markov decision process [31]

4 Reinforcement Learning in The Control Theory

The following section will initially provide the rationale for integrating reinforcement learning into control theory, and subsequently establish a conceptual connection between the two, laying the groundwork for further discussion in subsequent sections. Highlighting the complex Riccati equations in optimal control serves as a compelling justification for applying reinforcement learning algorithms in control theory. Typically, these equations are solved offline and in reverse chronological order, rendering them less suitable for real-time online applications, and they necessitate a precise understanding of the system's underlying dynamics. In practice, our understanding of these dynamics is often constrained, requiring the development of an optimal policy through engagement with the environment. One of the key advantages of reinforcement learning is its capacity to efficiently handle systems with poorly specified underlying mechanisms. The primary objective in reinforcement learning involves the agent selecting actions that either raise or lower the overall reward totals acquired. The characteristic of the reward applies to both states and actions, and is commonly referred to as a penalty function when attempting to minimize total rewards due to its negative impact. In control theory, the primary goal is to reduce tracking discrepancies, which are viewed as either an incentive or a disincentive, with the aim of minimizing this difference. An accurate definition of the reward can be instrumental in attaining the desired control objectives. The reward function was specifically designed to align with the objectives we are currently working to accomplish. The equation in question is a well-known illustration of a reward function, specifically as presented in Eq 6.

$$r = f(x, u) = r(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k \quad \text{Eq 6}$$

This corresponds to the incentive linked to the state x_k resulting from the action u_k being taken. In the context of applying reinforcement learning to control theory, the primary goal is to reduce the total cumulative rewards $r(x_k, u_k)$, as opposed to solely concentrating on decreasing the cost at each stage. The primary objective is to lower the strategic cost. The stage cost is determined by the current state, the action taken in that state, and the resulting next state, while the strategic cost is calculated

by summing the discounted rewards over time, as described in Eq 7.

$$J_{k,t} = \sum_{i=k}^{k+T} \lambda^{i-k} r_i \implies J_{k,t} = r_k + \lambda r_{k+1} + \cdots \lambda^T r_{k+T} \quad \text{Eq 7}$$

4.1 State Value

When the reward is randomly variable, as specified in Eq 8, the r parameter will have varying values, making it impossible to derive an exact value for J parameter. Instead of reducing a particular outcome of incentives, we should aim to reduce the anticipated value of those incentives. In light of this specific context, it is appropriate to establish the value of a state within a particular policy framework:

$$V_k^{\pi}(x) = E_{\pi}\{J_{k,T} | x_k = x\} = E_{\pi}\{\sum_{i=k}^{k+T} \lambda^{i-k} r_i \, | x_k = x\} \quad \text{Eq 8}$$

4.2 Optimal Policy and Optimal Value

The Eq 9 is applicable to a wide range of random environments and reward systems. Taking into account the deterministic policy and its associated costs, and understanding that the expected value remains unchanged, we can remove the expected value from the equation. As a result, the relationship defining the value of a state will be described as follows:

$$V^{\pi}(\mathrm{x}) = \sum_{i=k}^{k+T} \lambda^{i-k} r_i \quad \text{Eq 9}$$

Based on the statement that in a deterministic setting, the policy π functions as a controller (Eq 10), it follows that:

$$u = \mu(x) \quad \text{Eq 10}$$

We will encounter an infinite form for μ, and the optimal policy is the one that yields the lowest possible value. In essence:

$$\pi^*(x,u) = \underset{\pi}{\operatorname{argmin}} V_k^{\pi}(x) = \underset{\pi}{\operatorname{argmin}} E_{\pi}\{\sum_{i=k}^{k+T} \lambda^{i-k} r_i \, | x_k = x\} \quad \text{Eq 11}$$

The optimal values for the state under the optimal policy are denoted as the optimal values, as shown in Eq 12.

$$V_k^*(x) = \min_{\pi} V_k^{\pi}(x) = \min_{\pi} E_{\pi}\{\sum_{i=k}^{k+T} \lambda^{i-k} r_i \, | x_k = x\} \quad \text{Eq 12}$$

4.3 Bellman Equations

The Bellman Equation will be employed to solve for the optimal control of systems described by Eq 11 and 12, based on its recursive form of the state value as outlined in Eq 13.

$$V_k^{\pi}(x) = E_{\pi}\{r_k + \lambda \sum_{i=k+1}^{k+T} \lambda^{i-(k+1)} r_i \, | x_k = x\} \quad \text{Eq 13}$$

We calculate the mathematical expectation from Eq 28, taking into account that both the subsequent state x' and action u are stochastic variables, necessitating computation of the mathematical expectation with respect to x' and u. The probability of choosing action u in state x is represented as $\pi(x,u)$. Additionally, when in state x and performing action u, the likelihood of moving to state x′ is represented by $P_{xx'}^u$. This transition yields a reward of $R_{xx'}^u$. Utilizing the principle of linearity allows us to calculate the expected value mathematically:

$$V_k^{\pi}(x) = \sum_u \pi(x,u) \sum_{x'} P_{xx'}^u \left[R_{xx'}^u + \lambda V_{k+1}^{\pi}(x')\right] \quad \text{Eq 14}$$

The concluding equation (Eq 14) is a recursive formulation recognized as the Bellman Equation. This equation is associated with the policy π and is employed to ascertain the value of states in accordance with the π policy.

By replacing V^* with V^{π} in Eq 15 and applying the minimization condition, we will obtain the subsequent result:

$$V_k^*(x) = \min_{\pi} \sum_u \pi(x,u) \sum_{x'} P_{xx'}^u \left[R_{xx'}^u + \lambda V_{k+1}^*(x')\right] \quad \text{Eq 15}$$

The policy should be chosen to ensure that V^* denotes the minimum value that can be attained. In other words:

$$\pi^*(x,u) = \underset{\pi}{argmin} \sum_u \pi(x,u) \sum_{x'} P_{xx'}^u \left[R_{xx'}^u + \lambda V_{k+1}^*(x')\right] \quad \text{Eq 16}$$

In Eq 16, $P_{xx'}^u$ functions as the system's transfer function, delineating the dynamics of the environment. Within control systems, we operate under the premise that the environment, rewards, and actions are deterministic, which permits the removal of the sigma (∑) that signifies mathematical expectations. Additionally, given the relationship $x' = x_{k+1} = Ax_k + Bu_k$; if we are in state x_k and perform action u_k, we will deterministically arrive at the subsequent state x'. An alternative formulation of $V_k^*(x)$ can be expressed as shown in Eq 17.

$$V^*(x_k) = r(x_k, u_k) + \lambda V^*(x_{k+1}) \quad \text{Eq 17}$$

In Eq 17, $r(x_k, u_k)$ signifies the reward acquired from performing action u_k in state x_k, whereas $V^*(x_{k+1})$ indicates the optimal value associated with the next state.

4.4 Bellman Equations for Both Policy Evaluating and Improvement

An application of Bellman equations can be found in the processes of policy evaluation and policy improvement. According to the Bellman relation, when action u is dictated by the policy π, the relationship between the value of state x and the resulting state is articulated in Eq 18.

$$V^{\pi}(x) = \sum_u \pi(x,u) \sum_{x'} P_{xx'}^u \left[R_{xx'}^u + \lambda V^{\pi}(x')\right] \quad \text{Eq 18}$$

By solving the Bellman Equation, it is possible to ascertain $V^{\pi}(x)$, which signifies the value associated with each state. This procedure is referred to as *policy evaluation*. In scenarios where the number of states is finite, the value of each state can be obtained by formulating a system of equations. Conversely, in control problems characterized by an infinite number of states, this approach is not applicable, necessitating the use of a continuous representation for $V^{\pi}(x)$. For policy improvement, we establish the relationship outlined in Eq 33 through the Bellman equations.

$$\pi'(x,u) = \underset{\pi}{argmin} \sum_{x'} P_{xx'}^u \left[R_{xx'}^u + \lambda V^{\pi}(x')\right] \quad \text{Eq 19}$$

In reinforcement learning, the process consists of two distinct phases: policy evaluation and policy improvement. We begin with an initial policy denoted as π_0 and conduct an evaluation to determine the value V_0. Utilizing this value, we formulate a new policy, π_1. Subsequently, we evaluate policy π_1 to obtain V_1. With the value V_1 in hand, we proceed to create another policy, π_2. Through multiple iterations of this cycle, we ultimately converge on the optimal policy π^* and the corresponding optimal value V^*.

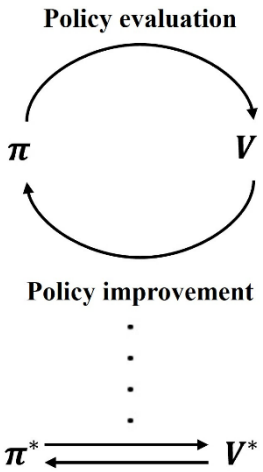


Figure 2: Cycle of policy evaluation and policy improvement to achieve the optimal policy and value [2]

4.5 Reinforcement Learning Algorithms

Reinforcement learning includes a range of algorithms, among which those that make use of existing model dynamics are classified as dynamic programming methods. Dynamic programming encompasses three distinct types of algorithms: *policy iteration*, *value iteration*, and *generalized policy iteration*. In contrast, there exists a separate category known as model-free methods, which operate independently of model access. The algorithms of *temporal difference* and *Q-learning*, which are the focus of this paper, belong to this model-free category. This paper specifically employs the Q-learning method for the design of a Linear Quadratic Regulator for a system governed by partial differential equations.

5 Designing of LQR Using the Q-learning

Q-learning illustrates that a system model is not required in either the policy evaluation or policy improvement stages, facilitating the creation of an adaptive optimal controller. To accomplish this, we introduce a function referred to as the quality function, as specified in Eq 20.

$$Q_h(x_k, u_k) = r(x_k, u_k) + \lambda V_h(x_{k+1}) \quad \text{Eq 20}$$

This function quantifies the value associated with a specific state-action pair. In this context, r signifies the reward received from executing action u_k, while $V_h(x_{k+1})$ represents the value of the resulting state. It is crucial to note that the quality function is influenced by both the state and the action taken, unlike the value function, which is determined exclusively by the state. Essentially, the quality function reflects the value derived from action u_k when situated in state x_k. Eq 21 serves as a general formulation for policy h, and under optimal conditions, it can be expressed using Eq 21.

$$Q^*(x_k, u_k) = r(x_k, u_k) + \lambda V^*(x_{k+1}) \quad \text{Eq 21}$$

V^* is represented by Eq 22, which is exclusively a function of x:

$$V^*(x_k) = \min_u Q^*(x_k, u_k) \quad \text{Eq 22}$$

In a particular state x_k, the optimization of Q^* concerning u results in V^*. From the preceding discussions, it can be inferred that h^*, representing the optimal policy, is equivalent to:

$$h^*(x_k) = \operatorname{argmin}_u Q^*(x_k, u) \quad \text{Eq 23}$$

It is essential to understand, as indicated in Eq 23, that the optimal policy can be determined solely from Q^* and the optimization problem, without the necessity of a system model. By differentiating Q^* with respect to the control variable u and identifying the roots of the resulting derivative, the optimal policy can be derived.

Subsequently, we will introduce Bellman's optimality equation for the quality function to establish a return relationship. While we initially defined Q in relation to r and V, it is also possible to articulate a return relationship for Q as presented in Eq 24.

$$Q^*(x_k, u_k) = r(x_k, u_k) + \lambda Q^*(x_{k+1}, h^*(x_{k+1})) \quad \text{Eq 24}$$

In Eq 24, $r(x_k, u_k)$ denotes the reward acquired at time k, whereas $Q^*(x_{k+1}, h^*(x_{k+1}))$ signifies the value associated with the state-action pair at the subsequent time step.

In the context of the Linear Quadratic Regulator (LQR) problem, it is established that the quality function is represented as a quadratic function of both the state and the action, as indicated in Eq 25.

$$Q_k(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k + x_{k+1}^T P x_{k+1} \quad \text{Eq 25}$$

Substituting the system dynamics represented by the equation ($X_{k+1} = AX_k + BU_k$) into Eq 25 will result in Eq 26.

$$Q_k(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k + (Ax_k + Bu_k)^T P (Ax_k + Bu_k) \quad \text{Eq 26}$$

By streamlining the algebra presented in Eq 26, it can be reformulated into the subsequent matrix representation:

$$Q_k(x_k, u_k) = \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \begin{bmatrix} Q + A^T P A & B^T P A \\ A^T P B & R + B^T P B \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \quad \text{Eq 27}$$

By substituting the variables as specified in Equations 28 and 29, and subsequently reformulating Eq 27, we ultimately derive Eq 30. This equation illustrates that the state-action value is represented as a quadratic function of both the state and the action.

$$\begin{bmatrix} x_k \\ u_k \end{bmatrix} = Z \quad \text{Eq 28}$$

$$\begin{bmatrix} Q + A^T P A & B^T P A \\ A^T P B & R + B^T P B \end{bmatrix} = H \quad \text{Eq 29}$$

$$Q_k(x_k, u_k) = Z_k^T H Z_k \quad \text{Eq 30}$$

The symmetrical and second-order characteristics of Q enable its derivation without the necessity of knowing the matrices A and B; rather, this can be achieved exclusively through engagement with the environment. After Q has been established and optimized concerning u_k, the optimal policy can be realized.

A fundamental component of Q-learning is the implicit representation of system dynamics within matrix H, which is derived from interactions with the environment. In the process of policy evaluation, where matrix H is established through these interactions, we incorporate the second-order formulation of Q^* (as presented in Eq 31) into Bellman's Equation, which is defined for the state-action value (Eq 25). This leads to the reformulation expressed in Eq 31.

$$Z_k^T H Z_k = x_k^T Q x_k + u_k^T R u_k + Z_{k+1}^T H Z_{k+1} \quad \text{Eq 31}$$

By utilizing the definition of Kronecker multiplication, we arrive at Eq 32.

$$Z_k^T H Z_k = \hat{H}^T \hat{Z}_k \quad \text{Eq 32}$$

The expression $\hat{Z}_k$ represents the Kronecker product of the vector Z, while $\hat{H}$ is also classified as a vector. Thus, Eq 32 can be expressed Eq 33.

$$(\hat{Z}_k - \hat{Z}_{k+1})\hat{H}^T = x_k^T Q x_k + u_k^T R u_k \quad \text{Eq 33}$$

Ultimately, we arrive at the equation that can be formulated from the data $\hat{H}$. Upon acquiring $\hat{H}$, enhancements to the policy can be implemented. For the purpose of policy improvement, Eq 30 is reformulated as Eq 34, wherein the matrix H is expressed in a block structure.

$$Q_k(x_k, u_k) = Z_k^T H Z_k = \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \quad \text{Eq 34}$$

To enhance policy improvement, it is crucial to reduce Q_k with respect to u in Eq 34. By streamlining Eq 33 and derivation with respect to u, we obtain Eq 35, which integrates the principle of state feedback.

$$H_{21}x_k + H_{22}x_k = 0 \implies u_k = -(H_{22})^{-1}H_{21}x_k \implies u_k = -\boldsymbol{K}x_k \quad \text{Eq 35}$$

The information presented indicates that an understanding of the system dynamics is not required during both the evaluation and improvement phases. Evaluation is conducted through engagement with the environment, and subsequent to the calculation of H as per Eq 35, the value of K is updated.

5.1 Presentation of the System

This section focuses on defining the problem in order to analyze the dynamics of the system being examined in the case study. we describe the key dynamics of the system that are crucial for our calculations and simulations. A notable illustration of a natural occurrence controlled by partial differential equations (PDEs) is the one-dimensional heat diffusion equation. This article presumes the dynamic equations governing the system at hand are already known, and it goes on to explore simulations and controller design methodologies.

Our objective is to regulate the temperature at a user-specified point by adjusting the heat supplied to the rod from an external source, such as a burner or heater, to keep it at a predetermined reference temperature. The differential equations describing this problem are formulated with partial derivatives and are expressed as in Eq 36.

$$\frac{\partial u(t,x)}{\partial t} = a\frac{\partial^2 u(t,x)}{\partial x^2} + b\,Q(x,t) \quad \text{Eq 36}$$

$u(t,x)$: Represents the temperature at a specific point, x, at a particular time, t.

$Q(x,t)$: Indicates the temperature entering the rod from an external source at time t and point x.

a and b: The parameters in question are influenced by the physical characteristics of the rod, including specific heat, density, and thermal capacity.

As highlighted in section 3.1 on the discretization of partial differential equations, resolving these equations analytically is typically impractical, which results in a scarcity of exact solutions. To resolve this issue, we utilize numerical techniques, including the finite element method, for their implementation. We employ an approximation for the first derivative of temperature with respect to its spatial location, as specified in Eq 37.

$$\frac{\partial u}{\partial x_i} = \frac{u(t,x_{i+1}) - u(t,x_i)}{\delta x} \quad \text{Eq 37}$$

The first-order approximation yields the second derivative of temperature with respect to its spatial location, as described in Eq 38.

$$\frac{\partial^2 u}{\partial x_i^2} = \frac{u(t,x_{i+1}) - 2u(t,x_i) + u(t,x_{i-1})}{\delta^2 x} \quad \text{Eq 38}$$

Substituting an approximation for the spatial variation of temperature's second derivative into heat distribution dynamic (Eq 36) allows us to express the governing partial differential equations for the rod's temperature in a discrete form.

$$\frac{\partial u(t,x_i)}{\partial t} = a\frac{u(t,x_{i+1}) - 2u(t,x_i) + u(t,x_{i-1})}{\delta^2 x} + bQ(t,x_i) \quad \text{Eq 39}$$

We can derive an ordinary differential equation for each element using Eq 39. This process ultimately leads to an n-th order differential equation, which serves as an approximation of the system modelled by the partial differential equations that govern temperature distribution. The temperature $u(t,x)$ is assumed to be constant over each interval, representing conditions at point x and time t. To avoid confusion, we will segment the rod into 10 identical segments.

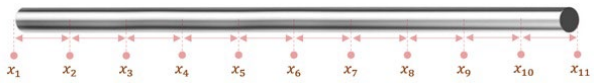


Figure 3: Dividing the rod into ten equal sections

To establish the problem's boundary conditions, the following statements will be defined:

- The temperatures at both ends of the rod are specified by equations (40) and (41).

$$u(t,x_f) = T_f \quad \text{Eq 40}$$

$$u(t,x_e) = T_e \quad \text{Eq 41}$$

Here, T_f represents the temperature at the beginning of the rod, and T_e denotes the temperature at the end of the rod.

- At time zero, the temperature is set to ambient temperature at all points between the initial and final locations (Eq 42)

$$u(0,x_i) = T_{amb} \quad \text{Eq 42}$$

5.2 State Space Representation

The states can be characterized as follows:

$$X = [u(t,x_2), u(t,x_3), \dots\dots\dots u(t,x_{10})] \quad \text{Eq 43}$$

An infinite-dimensional ordinary differential equation can be transformed into a system of ninth-order differential equations via the discretization process.

$$for\ i = 2 \rightarrow \dot{X}_1 = a\frac{u(t,x_3) - 2u(t,x_2) + u(t,x_1)}{\delta^2 x} + b\,Q(t,x_2)$$

$$for\ i = 3 \rightarrow \dot{X}_2 = a\frac{u(t,x_4) - 2u(t,x_4) + u(t,x_2)}{\delta^2 x} + b\,Q(t,x_3)$$

$$\vdots$$

$$for\ i = 10 \rightarrow \dot{X}_9 = a\frac{u(t,x_{11}) - 2u(t,x_{10}) + u(t,x_9)}{\delta^2 x} + b\,Q(t,x_{10})$$

The state space representation is formulated based on the relationships previously described and the substitution of state variables into the discrete and approximate form of the second derivative, as shown below:

$$\dot{X} = AX + BQ + T_{dist} \quad \text{Eq 44}$$

As a consequence, we will derive the matrices A, B, and T_{dist}, expressed as equations Eq 45 through 47.

$$A = \begin{bmatrix} -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 \end{bmatrix} \quad \text{Eq 45}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ b \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \text{Eq 46}$$

$$T_{dist} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{Eq 47}$$

6 Simulation

In this section, results of applying Q-learning algorithm on simulated systems are presented. The main objective of this section is to show the capability of using Q-learning in designing the LQR for one-dimensional heat diffusion equation in a rod as a system governed by PDE.

The theoretical concepts presented in Section 5 have been implemented within the software environment to elucidate the relationships discussed. By executing the program, one can observe and interpret the resulting data. This algorithm is designed to define a discrete system in accordance with the state space equations specified in Section 5.2. Following this, the relationships articulated in Section 5 were integrated into the software framework. Upon running the program, the optimal policy can be achieved by the fourth iteration. In this algorithm, the initial policy is characterized as a state feedback gain vector, with coefficients K_1 through K_9 assigned specific values:

Table 1: Coefficients K1 to K9 of initial policy

Ki	*Value*	*Ki*	*Value*	*Ki*	*Value*
K_1	1.2853	K_4	-0.0117	K_7	0.3611
K_2	0.2717	K_5	0.4471	K_8	-0.6917
K_3	-0.3343	K_6	-0.1506	K_9	-0.1962

The matrix K, representing the optimal policy established through four iterations, was derived in the following:

$$\begin{bmatrix} 1.2853 & 0.2717 & -0.3343 & -0.0117 & 0.4471 & -0.1506 & 0.3611 & -0.6917 & -0.1962 \\ 0.005729 & 0.000529 & -0.02596 & 0.001513 & 0.003971 & -0.00149 & 0.00507 & -0.002429 & 0.0015824 \\ 0.000711 & 0.0014108 & -0.02294 & 0.001813 & 0.001508 & 0.001196 & 0.000907 & 0.000595 & 0.000302 \\ 0.000703 & 0.0014123 & -0.02293 & 0.001812 & 0.001503 & 0.001201 & 0.000900 & 0.0006 & 0.00030 \\ 0.000703 & 0.0014123 & -0.02293 & 0.001812 & 0.001503 & 0.001201 & 0.000900 & 0.0006 & 0.00030 \end{bmatrix}$$

The results obtained from this program are compared with those derived from the classical method, as illustrated in Table 3. This comparison indicates that the Q-learning algorithm is both effective and satisfactory for designing the controller problem under consideration.

Table 2: Comparison of the optimal policy obtained from the classical method and the Q-learning

Method	K1	K2	K3	K4	K5	K6	K7	K8	K9
Classic	0.0006	0.001	-0.022	0.0017	0.0014	0.0011	0.0008	0.0005	0.0002
Q-learning	0.0006	0.0013	-0.022	0.0017	0.0014	0.0011	0.0008	0.0005	0.0002

The graphical representation of the interest coefficients' convergence is presented in Figures 4 through 12, whereas Figure 13 depicts the reward diagram associated with this algorithm.

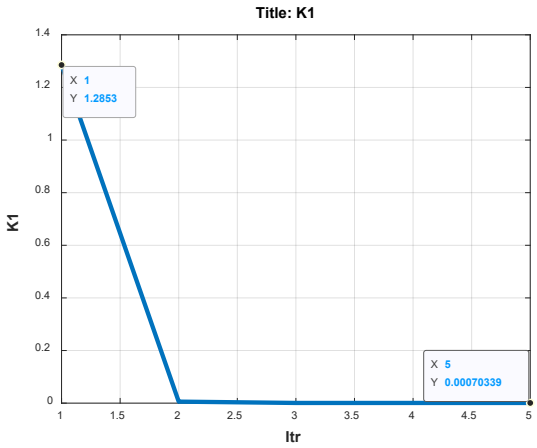


Figure 4: Convergence diagram of *K*1 showing both the initial and optimal values.

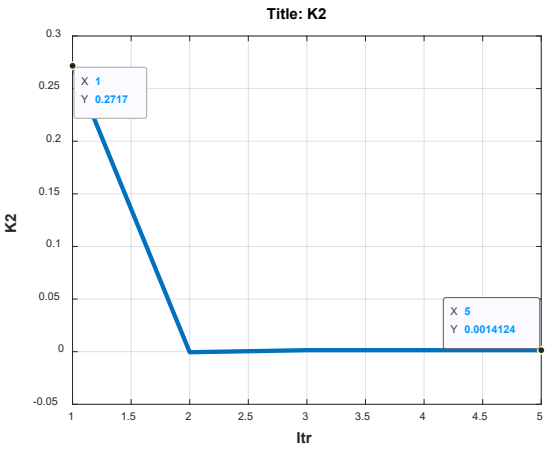


Figure 5: Convergence diagram of *K2* showing both the initial and optimal values.

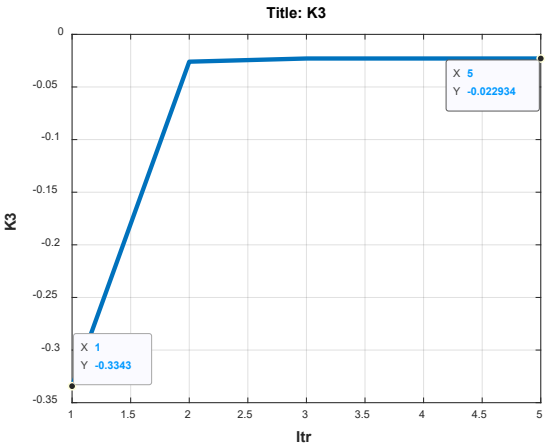


Figure 6: Convergence diagram of *K3* showing both the initial and optimal values.

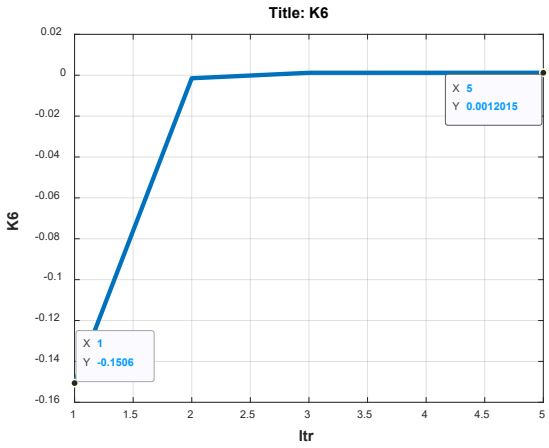


Figure 9: Convergence diagram of *K6* showing both the initial and optimal values.

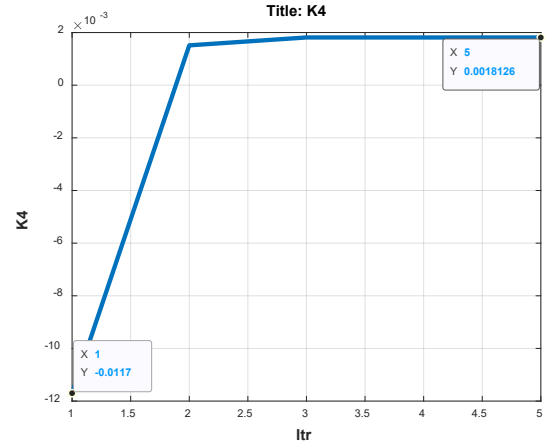


Figure 7: Convergence diagram of *K4* showing both the initial and optimal values.

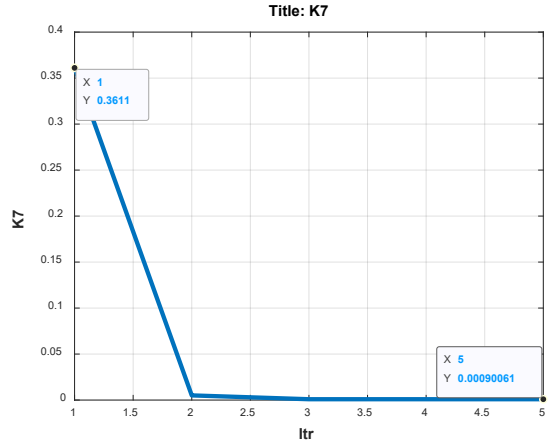


Figure 10: Convergence diagram of *K7* showing both the initial and optimal values.

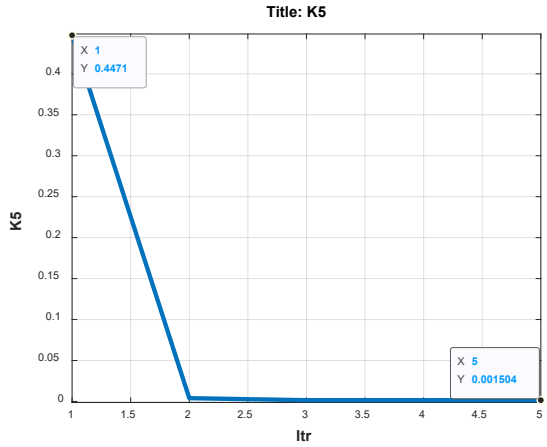


Figure 8: Convergence diagram of *K5* showing both the initial and optimal values.

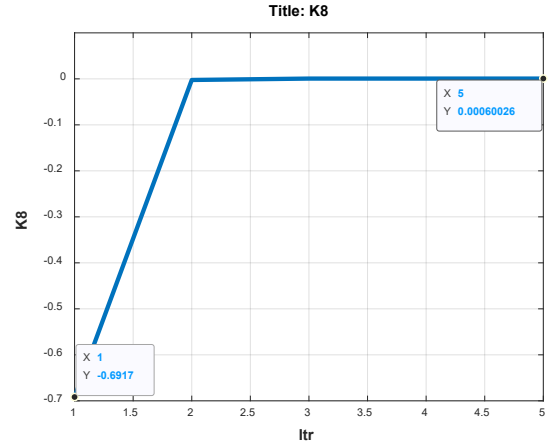


Figure 11: Convergence diagram of *K8* showing both the initial and optimal values.

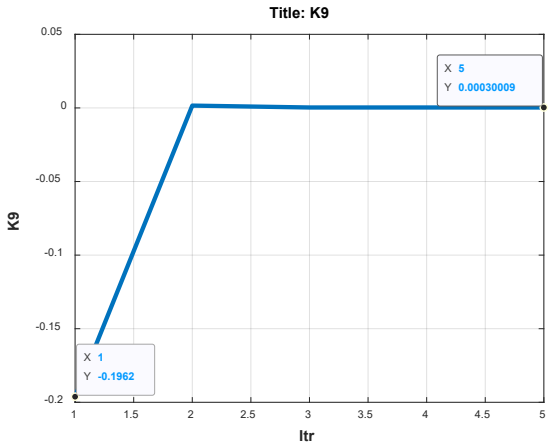


Figure 12: Convergence diagram of *K9* showing both the initial and optimal values.

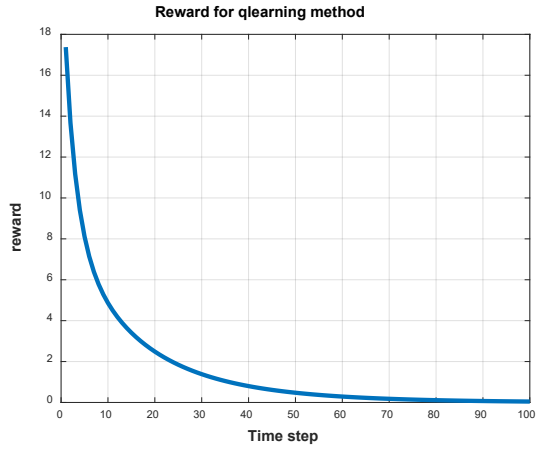


Figure 13: Reward for the Q-learning algorithm.

Following the acquisition of the gain coefficient vectors for the controller utilizing the reinforcement learning approach, simulations were conducted on the closed-loop system incorporating these coefficients. Subsequently, Figure 14 illustrates the timing diagram depicting the system's response to a step input when employing the feedback controller derived from this optimal state method.

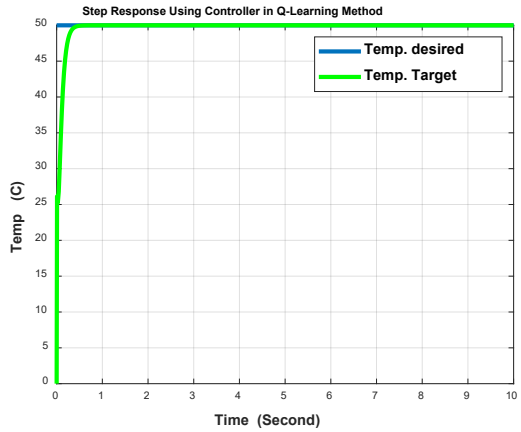


Figure 14: Response to a step input for the system utilizing the optimal state feedback controller derived from the Q-learning method.

This figure demonstrates the effectiveness of the reinforcement learning technique in the controller design process.

7 Conclusion

The comparison of the results indicates that, the Q-learning method is effective for design purposes and can be utilized in controller design for partial differential equation (PDE) systems. In a system described with PDE, it will be transferred to the ODE by discretization. Since the result of discretization is so closed to Markov decision process (MDP), it is possible to define the problem to the RL. A one-dimensional heat equation has been chosen as a case study to effectively illustrate and communicate the principles of controller design utilizing the reinforcement learning approach, and the controller design has been implemented for this system.

Disclosure of Potential Conflicts of Interest

The Authors declare that there is no conflict of interest

Reference

[1] R. Singh and B. Bhushan, "Reinforcement Learning-Based Model-Free Controller for Feedback Stabilization of Robotic Systems," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 10, pp. 7059-7073, Oct. 2023, doi: 10.1109/TNNLS.2021.3137548.

[2] K. Iqbal and M. Haras, "Reinforcement learning of LQR control policy by a double inverted-pendulum biomechanical model," *2023 IEEE International Conference on Industrial Technology (ICIT)*, Orlando, FL, USA, 2023, pp. 1-6, doi: 10.1109/ICIT58465.2023.10143150.

[3] S.A. Asad rizvi and Z. Lin, " Reinforcement Learning-Based Linear Quadratic Regulation of Continuous-Time Systems Using Dynamic Output Feedback," *IEEE Transactions on Cybernetics PP(99):1-10,* doi: 10.1109/TCYB.2018.2886735.

[4] S. Mukherjee and T. L. Vu, "Reinforcement Learning of Structured Stabilizing Control for Linear Systems With Unknown State Matrix," in *IEEE Transactions on Automatic Control*, vol. 68, no. 3, pp. 1746-1752, March 2023, doi: 10.1109/TAC.2022.3155384.

[5] Hwang, R., Lee, J. Y., Shin, J. Y., & Hwang, H. J. (2022). Solving pde-constrained control problems using operator learning. *In Proceedings of the AAAI Conference on Artificial Intelligence*, *Vol. 36*, pp. 4504-4512.

[6] Benosman, M., Chakrabarty, A., & Borggaard, J. (2020). Reinforcement learning-based model reduction for partial differential equations. *IFAC-PapersOnLine*, 7704-7709.

[7] Farahmand, A. M., Nabi, S., & Nikovski, D. N. (2017). Deep reinforcement learning for partial differential equation control. *In 2017 American Control Conference (ACC)*, (pp. 3120-3127).

[8] Voropai, R., Geletu, A., & Li, P. (2023). Model Predictive Control of Parabolic PDE Systems under Chance Constraints. *Mathematics*.

[9] Dodhia, A. W. (2021). Machine learning-based model predictive control of diffusion-reaction processes. *Chemical Engineering Research and Design*, 173, 129-139

[10] Yang, Y., Dubljevic, S., & Li, S. (2021). Economic model predictive control for transport-reaction systems with target profiles. *Control Engineering Practice*, 107, 104684.

[11] Lamare, P. O., & Bekiaris-Liberis, N. (2015). Control of 2× 2 linear hyperbolic systems: Backstepping-based trajectory generation and PI-based tracking. *Systems & Control Letters*, 86, 24-33.

[12] Luo, B., Wu, H. N., & Li, H. X. (2014). Adaptive optimal control of highly dissipative nonlinear spatially distributed processes with neuro-dynamic programming. *IEEE transactions on neural networks and learning systems*, 26(4), 684-696.

[13] Krstic, M. (2013). Adaptive control of anti-stable wave PDE systems: theory and applications in oil drilling. *IFAC Proceedings Volumes*, 46(11), 432-439.

[14] Wang, J. W., Tsai, S. H., Li, H. X., & Lam, H. K. (2018). Spatially piecewise fuzzy control design for sampled-data exponential stabilization of semilinear parabolic PDE systems. *IEEE Transactions on Fuzzy Systems*, 26(5), 2967-2980.

[15] Luo, B., Wu, H. N., & Li, H. X. (2014). Adaptive optimal control of highly dissipative nonlinear spatially distributed processes with neuro-dynamic programming. *IEEE transactions on neural networks and learning systems*, 26(4), 684-696.

[16] Yu, H., Park, S., Bayen, A., & Krstic, M. (2021). Reinforcement Learning versus PDE Backstepping and PI Control for Congested Freeway Traffic. *IEEE Transactions on Control Systems Technology*, 30(4). Retrieved from https://arxiv.org/abs/1904.12957

[17] Peitz, S., Stenner, J., Chidananda, V., Wallscheid, O., Brunton, S., & Taira, K. (2024). Distributed control of partial differential equations using convolutional reinforcement learning. *Physica D: Nonlinear Phenomena*, 461.

[18] Pirmorad, E., Khoshbakhtian, F., Mansouri, F., & Farahmand, A. (2021). Deep reinforcement learning for online control of stochastic partial differential equations. *arXiv preprint*, arXiv:2110.11265.

[19] Kim, J. W., Park, B. J., & Yoo, H. (2020). A model-based deep reinforcement learning method applied to finite-horizon optimal control of nonlinear control-affine system. *Journal of Process Control*, 166-178.

[20] He, W., H. Gao, C. Zhou, C. Yang, & Z. Li. (2021, Dec). Reinforcement Learning Control of a Flexible Two-Link Manipulator: An Experimental Investigation. *vol. 51*.

[21] Phuong Nam Dao; Yen-Chen Liu;. (2022). Adaptive reinforcement learning in control design for cooperating manipulator systems. *Asian Journal of Control*.

[22] I. Aksikas, A. Fuxman, J. F. Forbes, and J. J. Winkin, "LQ control design of a class of hyperbolic PDE systems: Application to fixed-bed reactor," Automatica, vol. 45, pp. 1542-1548, 2009.

[23] I. Aksikas, J. J. Winkin, and D. Dochain, "Optimal LQ-feedback regulation of a nonisothermal plug flow reactor model by spectral factorization," *IEEE Transactions on Automatic Control,* vol. 52, pp. 1179-1193, 2007

[24] J. Choi and K. S. Lee, "Model predictive control of concurrent first-order hyperbolic PDE systems," *Industrial & engineering chemistry research,* vol. 44, pp. 1812-1822, 2005

[25] S. Dubljevic, P. Mhaskar, N. H. El-Farra, and P. D. Christofides, "Predictive control of transport reaction processes," *Computers & chemical engineering,* vol. 29, pp. 2335-2345, 2005.

[26] P. D. Christofides and P. Daoutidis, "Feedback control of hyperbolic PDE systems," *AIChE Journal,* vol. 42, pp. 3063-3086, 1996.

[27] J.-W. Wang, H.-N. Wu, and H.-X. Li, "Stochastically exponential stability and stabilization of uncertain linear hyperbolic PDE systems with Markov jumping parameters," *Automatica,* vol. 48, pp. 569-576, 2012.

[28] E. M. Hanczyc and A. Palazoglu, "Sliding mode control of nonlinear distributed parameter chemical processes," *Industrial & engineering chemistry research,* vol. 34, pp. 557-566, 1995

[29] H. Sira-Ramirez, "Distributed sliding mode control in systems described by quasilinear partial differential equations," *Systems & Control Letters,* vol. 13, pp. 177-181, 1989.

[30] Lewis, Frank L. (2012). Reinforcement learning and feedback control using natural decision methods to design optimal adaptive controllers. *IEEE CONTROL SYSTEM MAGAZINE*.

[31] Sutton, R. S., and A. G. Barto, *Reinforcement Learning—An Introduction*, Cambridge, MA: MIT Press, 2018