Actor-Critic Reinforcement Learning for Control With Stability Guarantee

Minghao Han , Lixian Zhang , Jun Wang, and Wei Pan

***Abstract*—Reinforcement Learning (RL) and its integration with deep learning have achieved impressive performance in various robotic control tasks, ranging from motion planning and navigation to end-to-end visual manipulation. However, stability is not guaran- teed in model-free RL by solely using data. From a control-theoretic perspective, stability is the most important property for any control system, since it is closely related to safety, robustness, and reliability of robotic systems. In this letter, we propose an actor-critic RL framework for control which can guarantee closed-loop stability by employing the classic Lyapunov’s method in control theory. First of all, a data-based stability theorem is proposed for stochastic nonlinear systems modeled by Markov decision process. Then we show that the stability condition could be exploited as the critic in the actor-critic RL to learn a controller/policy. At last, the effectiveness of our approach is evaluated on several well-known 3-dimensional robot control tasks and a synthetic biology gene network tracking task in three different popular physics simulation platforms. As an empirical evaluation on the advantage of stability, we show that the learned policies can enable the systems to recover to the equilibrium or way-points when interfered by uncertainties such as system parametric variations and external disturbances to a certain extent.**

***Index Terms*—Reinforcement learning, stability, lyapunov’s method.**

1. INTRODUCTION

**R**

EINFORCEMENT learning is promising to highly non- linear control robotic systems with large state and ac-

tion space [1]. Until recently, significant progress has been made by combining advances in deep learning with reinforce- ment learning. Impressive results are obtained in a series of high-dimensional robotic control tasks where sophisticated and hard-to-engineer behaviors can be achieved [2]–[5]. However, the performance of an RL agent is by large evaluated through

Manuscript received February 24, 2020; accepted July 2, 2020. Date of

publication July 23, 2020; date of current version August 10, 2020. This letter was recommended for publication by Associate Editor M. Saveriano and Editor

D. Lee upon evaluation of the Reviewers’ comments. This work was supported in part by the Harbin Institute of Technology Scholarship (MHH), in part by the Major Scientific Research Project Cultivation Plan No. ZDXMPY20180101 (LXZ), and in part by AnKobot Smart Technologies (WP). *(Corresponding author: Lixian Zhang.)*

Minghao Han and Lixian Zhang are with the Department of Control Science and Engineering, Harbin Institute of Technology, Harbin 150001, China (e-mail: [mhhan@hit.edu.cn;](mailto:mhhan@hit.edu.cn) [lixianzhang@hit.edu.cn).](mailto:lixianzhang@hit.edu.cn)

Jun Wang is with the Department of Computer Science, University College London, London WC1E 6BT, U.K. (e-mail: [jun.wang@cs.ucl.ac.uk).](mailto:jun.wang@cs.ucl.ac.uk)

Wei Pan is with the Department of Cognitive Robotics, Delft University of Technology, Delft 2628 CD Delft, Netherlands (e-mail: [wei.pan@tudelft.nl).](mailto:wei.pan@tudelft.nl)

[This letter has supplementary downloadable material available at http:// ieeexplore.ieee.org, provided by the authors.](http://ieeexplore.ieee.org/)

Digital Object Identifier 10.1109/LRA.2020.3011351

trial-and-error and RL could hardly provide any guarantee for the reliability of the learned control policy.

Given a control system, regardless of which controller design method is used, the first and most important property of a system needs to be guaranteed is stability, because an unstable control system is typically useless and potentially dangerous [6]. A stable system is guaranteed to converge to the equilibrium or reference signal and it could recover to these targets even in the presence of parametric uncertainties and disturbances [7]. Thus stability is closely related to the robustness, safety, and reliability of the robotic systems.

The most useful and general approach for studying the stabil- ity of robotic systems is Lyapunov’s method [8], which is dom- inant in control engineering [9], [10]. In Lyapunov’s method, a scalar “energy-like” function called Lyapunov function *L* is constructed to analyze the stability of the system. The controller is designed to ensure that the difference of Lyapunov function along the state trajectory is negative definite for all time instants so that the state goes in the direction of decreasing the value of Lyapunov function and eventually converges to the equilib- rium [11], [12]. In learning methods, as the dynamic model is unknown, the “energy decreasing” condition has to be verified by trying out *all* possible consecutive data pairs in the state space,

i.e., to verify infinite inequalities *Lt*+1 *Lt <* 0. Obviously,

*−*

the “infinity” requirement is impossible thus making the direct

exploitation of Lyapunov’s method impossible.

In this letter, we propose a data-based stability theorem and an actor-critic reinforcement learning algorithm to jointly learn the controller/policy and a Lyapunov critic function both of which are parameterized by deep neural networks, with a focus on stabilization and tracking tasks in robotic systems. The contri- bution of our letter can be summarized as follows: 1) a novel data-based stability theorem where only one inequality on the expected value over the state space needs to be evaluated; 2) a sample approximation of the stability condition proposed above is exploited to derive an actor-critic algorithm to search for a controller with asymptotic stability guarantee (in the number of data points); 3) we show through experiments that the learned controller could stabilize the systems when interfered by un- certainties such as unseen disturbances and system parametric variations of a certain extent. In our experiment, we show that the stability guaranteed controller is more capable of handling uncertainties compared to those without such guarantees in non- linear control problems including classic CartPole stabilization tasks, control of 3D legged robots and manipulator, and reference tracking tasks for synthetic biology gene regulatory networks.

2377-3766 © 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.

See [https://www.ieee.org/publications/rights/inde](http://www.ieee.org/publications/rights/index.html)x.html for more information.

* 1. *Related Works*

In model-free reinforcement learning, stability is rarely ad- dressed due to the formidable challenge of analyzing and de- signing the closed-loop system dynamics in a model-free man- ner [13], and the associated stability theory in model-free RL remains as an open problem [13], [14].

Recently, Lyapunov analysis is used in model-free RL to solve control problems with safety constraints [15], [16]. In [15], the Lyapunov-based approach for solving constrained Markov decision processes is proposed with a novel way of constructing the Lyapunov function through linear programming. In [16], the above results were further generalized to continuous control tasks. Even though Lyapunov-based methods were adopted in these results, neither of them addressed the stability of the system. In this letter, the sufficient conditions for a dynamic system being stable are derived. Furthermore, it is shown that these conditions can be verified through sampling and ensured through model-free learning.

Other interesting results on the stability of learning-based con- trol systems are reported in recent years. In [17], an initial result is proposed for the stability analysis of deterministic nonlinear systems with optimal controller for infinite-horizon discounted cost, based on the assumption that discount is sufficiently close to 1. However, in practice, it is rather difficult to guarantee the op- timality of the learned policy unless certain assumptions on the system dynamics are made [18]. Furthermore, the exploitation of multi-layer neural networks as function approximators [19] only adds to the impracticality of this requirement. In this letter, it is shown that it is sufficient to ensure stability by satisfying the Lyapunov criterion that is evaluated on samples and thus one is exempt from finding the optimal/suboptimal solutions. In [20], local stability of Lipschitz continuous dynamic systems is analyzed by validating the “energy decreasing” condition on discretized points in the subset of state space with the help of a learned model (Gaussian process). Nevertheless, the dis- cretization technique may become infeasible as the dimension and space of interest increases, limiting its application to rather simple and low-dimensional systems. In this letter, the proposed method is applicable to the general class of stochastic dynamic systems modeled by MDP and does not need to learn a model for stability analysis and controller design.

1. PROBLEM STATEMENT

In this letter, we focus on the stabilization and tracking tasks for systems modeled by Markov decision process (MDP). The state of a robot and its environment at time *t* is given by the state *st* R*n*, where denotes the state space. The robot then takes an action *at* R*m* according to a stochastic policy

*∈A⊆*

*∈S ⊆ S*

*π*(*at st*), resulting in the next state *st*+1. The transition of the state is modeled by the transition probability *P* (*st*+1 *st, at*). In both stabilization and tracking tasks, there always is a cost

*|*

*|*

function *c*(*st, at*) to measure how good or bad a state-action pair is.

In stabilization tasks, the goal is to find a policy *π* such that

*ǁ ǁ ǁ·ǁ*

notes the Euclidean norm. In this case, cost function *c*(*st, at*)= the norm of state *st* goes to zero eventually, where de-

E*P* (*·|st,at*) *st*+1 . In tracking tasks, we divide the state *s* into two vectors, *s*1 and *s*2, where *s*1 is composed of elements of *s* that

are aimed at tracking the reference signal *r*, while *s*2 contains the rest. The reference signal could be the desired velocity, path

*ǁ ǁ*

tracking tasks, *c*(*st, at*)= E*P* (*·|s ,a* ) *s*1 *r* . and even the picture of grasping an object in a certain pose. For

*t*

*t*

*t*+1

*ǁ − ǁ*

From a control perspective, both stabilization and tracking

tasks are related to the asymptotic stability of the closed-loop system (or error system) under *π*, i.e., starting from an initial point, the trajectories of state always converge to the origin or

reference trajectory. Let *cπ*(*st*) ¾ E*a∼πc*(*st, at*) denote the cost

function under the policy *π*, the definition of stability studied in

this letter is given as follows.

mean cost if lim*t→∞* E*st cπ*(*st*)=0 holds for any initial con- dition *s*0 *s*0 *cπ*(*s*0) *b* . If *b* is arbitrarily large then the *Definition 1:* The stochastic system is said to be stable in

*∈ { | ≤ }*

stochastic system is globally stable in mean cost.

The above definition is equivalent to the mean square sta- bility [21], [22] when the cost *c* is chosen to be the norm of

when *c*(*st, at*)= E*P* (*·|s ,a* ) *s*1 *r* . Thus the stabilization the state; it is also equivalent to the partial stability [23], [24] and tracking tasks can be collectively summarized as finding a

*t*

*t*

*t*+1

*ǁ − ǁ*

policy *π* such that the closed-loop system is stable in mean cost according to Definition 1.

*ρ*(*s*0) denotes the distribution of starting states. The closed-loop transition probability is denoted as *Pπ*(*sj|s*) ¾ Before proceeding, some notations are to be defined. *A π*(*a|s*)*P* (*sj|s, a*)d*a*. We also introduce the closed-loop state distribution at a certain instant *t* as *P* (*s|ρ, π, t*), which could be defined iteratively: *P* (*sj|ρ, π, t* + 1) = *S Pπ*(*sj|s*)*P* (*s|ρ, π, t*)d*s, ∀t ∈* Z+ and *P* (*s|ρ, π,* 0) = *ρ*(*s*).

∫

∫

1. DATA-BASED STABILITY ANALYSIS

In this section, we propose the main assumptions and a new theorem for stability analysis of stochastic systems. We assume that the Markov chain induced by policy *π* is ergodic with a unique stationary distribution *qπ*,

*qπ*(*s*) = lim *P* (*s ρ, π, t*)

*|*

*t→∞*

as commonly exploited by many RL literature [25]–[28].

In Definition 1, stability is defined in relation to the set of starting states, which is also called the region of attraction (ROA). If the MSS system starts within the ROA, its trajectory will be surely attracted to the equilibrium. To build a data-based stability guarantee, we need to ensure that the states in ROA are accessible for the stability analysis. Thus the following assumption is made to ensure that every state in ROA has a chance to be sampled as the starting state.

*Assumption 1:* There exists a positive constant *b* such that

*ρ*(*s*) *>* 0*, s s cπ*(*s*) *b* .

*∀ ∈ { | ≤ }*

Our approach is to construct/find a Lyapunov function *L* :

*S →*

R+ of which the difference along the state trajectory is

negative definite, so that the state goes in the direction of decreas- ing the value of Lyapunov function and eventually converges to the origin. The Lyapunov’s method has long been used for stability analysis and controller design in control theory [29],

but mostly exploited along with a *known* model so that the energy decreasing condition on the entire state space could be

*ρ*(*s*)

Thus taking the relations above into consideration, (3) infers

transformed into one inequality regarding model parameters [6],

[30]. In the following, we show that without a dynamic model,

lim

1 .E

*L*(*s*) *−* E

*L*(*s*)Σ

this “infinity” problem could be solved through sampling and sufficient conditions for a stochastic system to be stable in mean cost are given.

*Theorem 1:* The stochastic system is stable in mean cost if

*α*3 lim E*P* (*s ρ,π,t*)*cπ*(*s*) (4)

*t→∞*

*≤ − |*

*N→∞ N*

*P* (*s|ρ,π,N* +1)

Since E*ρ*(*s*)*L*(*s*) is a finite value and *L* is positive definite, it follows that

there exists a function *L* : *S →* R+ and positive constants *α*1,

*α*2 and *α*3, such that

*t→∞*

*P* (*s|ρ,π,t*) *π*

*N→∞ N*

*α*3

*ρ*(*s*)

lim E

*c* (*s*) *≤* lim

1 . 1 E

*L*(*s*)Σ =0 (5)

*α*1*cπ*(*s*) *≤ L*(*s*) *≤ α*2*cπ*(*s*) (1)

Suppose that there exists a state *s*0 *∈ {s*0*|cπ*(*s*0) *≤ b}* and a

E*s∼μ*

(E*sj∼P*

*L*(*sj*) *− L*(*s*)) *≤ −α*3E*s∼μ*

*cπ*(*s*) (2)

positive constant *d* such that lim*t→∞* E*P* (*s|s*0 *,π,t*)*cπ*(*s*)= *d*, or

*π π*

where

*π* lim

*t→∞*

E*P* (*s|s*0*,π,t*)*cπ*

(*s*) = *∞*. Since *ρ*(*s*0) *>* 0 for all start-

*μπ*(*s*) ¾ lim

1 Σ

*P* (*st* = *s|ρ, π, t*)

ing states in *{s*0*|cπ*(*s*0) *≤ b}* (Assumption 1), it follows that

lim*t→∞* E*st∼P* (*·|π,ρ*)*cπ*(*st*) *>* 0, which is contradictory with

(5). Thus *∀s*0 *∈ {s*0*|cπ*(*s*0) *≤ b}*, lim*t→∞* E*P* (*s|s*0 *,π,t*)*cπ*(*s*)=

*N→∞ N t*=0

*N*

is the (infinite) sampling distribution.

*Proof:* The existence of the sampling distribution *μπ*(*s*) is guaranteed by the existence of *qπ*(*s*). Since the se- quence *{P* (*s|ρ, π, t*)*,t ∈* Z+*}* converges to *qπ*(*s*) as *t* ap-

proaches *∞*, then by the Abelian theorem, the sequence

1. Thus the system is stable in mean cost by Definition 1. □
   1. directs the choice and construction of Lyapunov function,

of which the details are deferred to Section IV. (2) is called the energy decreasing condition and is the major criterion for determining stability.

*Remark 1:* This remark is on the connection to previous

*{* 1 Σ*N*

*N*

*t*=0

*P* (*s|ρ, π, t*)*,N ∈* Z+*}* also converges and *μπ*(*s*)=

results concerning the stability of stochastic systems. It should

be noted that the stability conditions of Markov chains have

*qπ*(*s*). Combined with the form of *μπ*, (2) infers that

been reported in [21], [32], however, of which the validation

∫ lim

1 Σ

*P* (*s|ρ, π, t*)(E*Pπ* (*sj|s*)*L*(*sj*) *− L*(*s*))d*s*

requires verifying infinite inequalities on the state space if

is continuous. On the contrary, our approach solely validates

*S*

*S N→∞ N t*=0

*N*

one inequality (2) related to the sampling distribution *μ*, which

*≤ −α*3E*s∼qπ cπ*(*s*) (3)

First, on the left-hand-side, *L*(*s*) *α*2*cπ*(*s*) for all *s* ac- cording to (1). Since the probability density function *P* (*s ρ, π, t*) is (assumed to be) a bounded function on for all *t*, thus there

*S*

*|*

*≤ ∈S*

exists a constant *M* such that

*P* (*s|ρ, π, t*)*L*(*s*) *≤ Mα*2*cπ*(*s*)*, ∀s ∈ S, ∀t ∈* Z+

Second, the sequence *{* 1 Σ*N P* (*s|ρ, π, t*)*L*(*s*)*,N ∈* Z+*}*

*N*

*t*=0

further enables data-based stability analysis and policy learning.

1. ALGORITHM

In this section, we propose an actor-critic RL algorithm to learn stability guaranteed policies for the stochastic system. First, we introduce the Lyapunov critic function *Lc* and show how it is constructed. Then based on the maximum entropy actor-critic framework, we use the Lyapunov critic function in

the policy gradient formulation.

converges point-wise to the function *qπ*(*s*)*L*(*s*). According

sequence *fn*(*s*) converges point-wise to a function *f* and is to the Lebesgue’s Dominated convergence theorem [31], if a dominated by some integrable function *g* in the sense that,

*|fn*(*s*)*|≤ g*(*s*)*, ∀s ∈ S, ∀n*

Then we get

∫

∫

1. *Lyapunov Critic Function*

In our framework, the Lyapunov critic *Lc* plays a role in both stability analysis and the learning of the actor. To enable the actor-critic learning, the Lyapunov critic is designed to be

dependent on *s* and *a* and satisfies *L*(*s*)= E*a∼πLc*(*s, a*) with

lim

*n→∞*

*fn*(*s*)d*s* =

*S S*

lim *fn*

*n→∞*

(*s*)d*s*

the Lyapunov function *L*(*s*), such that it can be exploited in judging the value of (2). In view of the requirement above,

*Lc* should be a non-negative function of the state and action,

Thus the left-hand-side of (3)

*Lc* : *S× A →* R+. In this letter, we construct Lyapunov critic

∫ lim

1 Σ

*P* (*s|ρ, π, t*) .∫

*Pπ*(*sj|s*)*L*(*sj*)d*sj − L*(*s*)Σ d*s*

with the following parameterization technique,

*T*

*S N→∞ N t*=0 *S*

*N*

*N*

*Lc*(*s, a*)= *fφ*(*s, a*)

*fφ*(*s, a*) (6)

= lim

1 .*N*Σ+1

E*P* (*s|ρ,π,t*)*L*(*s*) *−* Σ

E*P* (*s|ρ,π,t*)*L*(*s*)Σ

where *fφ* is the output vector of a fully connected neural network with parameter *φ*. This parameterization ensures the positive

*N→∞ N*

*t*=1

*t*=0

definiteness of *Lc*(*s, a*), which is necessary since *L*(*s*) is positive

definite according to (1) and *L*(*s*) is the expectation of *L* (*s, a*)

*N→∞ N*

*P* (*s|ρ,π,N* +1)

*ρ*(*s*)

= lim

1 .E

*L*(*s*) *−* E

*L*(*s*)Σ

*c*

over the distribution of actions.

Theoretically, some functions, such as the norm of state and value function, naturally satisfy the basic requirement of being a Lyapunov function (1). These functions are referred to as Lyapunov candidates. However, Lyapunov candidates are conceptual functions without any parameterization, thus their gradient with respect to the controller is intractable and are not directly applicable in an actor-critic learning process. In the proposed framework, the Lyapunov candidate acts as a supervision signal during the training of *Lc*. During training,

*Lc* is updated to minimize the following objective function,

1. *Lyapunov Actor-Critic Algorithm*

In this subsection, we will focus on how to learn the controller in a novel actor-critic framework called Lyapunov Actor-Critic (LAC), such that the inequality (2) is satisfied. The policy learning problem is summarized as the following constrained optimization problem,

find *πθ* (10)

s.t. E (*L* (*s,j f* (*s, sj*)) *− L* (*s, a*)+ *α c*) (11)

*D*

*c*

*θ*

*c*

3

Σ Σ *−* E*D* log(*π*(*a|s*)) *≥ Ht* (12)

(*s, a*))2

(7)

where the stochastic policy

is parameterized by a deep neural

*c D* 2 *c*

*J*(*L* )= E

1

(*L* (*s, a*) *− L*

target

*πθ*

network *fθ* that is dependent on *s* and a Gaussian noise *s*. The

Lyapunov candidate, *L*(*s*)= E*a∼πL*target(*s, a*) and is the set where *L*target is the approximation target related to the chosen of collected transition pairs. In [15] and [20], the value function

*D*

has been proved to be a valid Lyapunov candidate where the approximation target is

*L*target(*s, a*)= *c* + max *γLjc*(*s,j aj*) (8)

*aj*

where *Lj* is the target network parameterized by *φj* as typically

constraint (11) is the parameterized inequality (2), which can be evaluated through sampling. One may be curious why in the second term of (13), only one Lyapunov critic is explicitly dependent on the stochastic policy, while the other dependent on the samples of the action. First, note that this estimator is also unbiased estimation of (2), although the variance may be

increased compared to replacing *a* with *fθ*(*s*). From a more prac-

tical perspective, having the second Lyapunov critic explicitly

dependent on *θ* will introduce a term in the policy gradient that

*c j* updates *θ* to increase the value of *L*(*s*), which is contradictory to

used in the actor-critic methods [19], [33]. *Lc* has the same struc- ture as *Lc*, but the parameter *φj* is updated through exponentially moving average of weights of *Lc* controlled by a hyperparameter

*τ* R(0*,*1), *φjk*+1 *τφk* + (1 *τ* )*φjk*.

*∈ ← −*

In addition to value function, the sum of cost over a finite time

horizon could also be employed as Lyapunov candidate, which is exploited in model predictive control literature [10], [34] for stability analysis. In this case,

our goal of stabilization. (12) is the minimum entropy constraint borrowed from the maximum entropy RL framework to improve the exploration in the action space [33], and *t* is the desired bound. Solving the above constrained optimization problem is equivalent to minimizing the following objective function,

*J*(*θ*)= E(*s,a,s,jc*)*∼D* [*β*(log(*πθ*(*fθ*(*s, s*)*|s*)) + *Ht*)

*H*

+ *λ*(*L* (*s,j f* (*s, sj*)) *− L* (*s, a*)+ *α c*)] (13)

*c θ c* 3

*t*+*N*

Σ

*L*target(*s, a*)= E*ct*

*t*

(9)

where *β* and *λ* are Lagrange multipliers that control the relative importance of constraint (11) and (12).

In the actor-critic framework, the parameters of the policy

Here, the time horizon *N* is a hyperparameter to be tuned, of which the influence will be demonstrated in the experiment in Section V.

The choice of the Lyapunov candidate plays an important role in learning a policy. Value function evaluates the infinite time horizon and thus offers a better performance in general but is rather difficult to approximate because of significant variance and bias [35]. On the other hand, the finite horizon sum of cost provides an explicit target for learning a Lyapunov function, thus inherently reduces the bias and enhances the learning process. However, as the model is unknown, predicting the future costs based on the current state and action inevitably introduces vari- ance, which grows as the prediction horizon extends. In prin- ciple, for tasks with simple dynamics, the sum-of-cost choice enhances the convergence of learning and robustness of the trained policies, while for complicated systems the choice of value function generally produces better performance. In this letter, we use both value function and sum-of-cost as Lyapunov candidates. Later in Section V, we will show the influence of these different choices upon the performance and robustness of trained policies.

network are updated through stochastic gradient descent of (13), which is approximated by

*∇θJ*(*θ*)= *β∇θ* log(*πθ*(*a|s*))

+ *β∇a* log(*πθ*(*a|s*))*∇θfθ*(*s, s*)

+ *λ∇aj Lc*(*s,j aj*)*∇θfθ*(*s, sj*) (14)

The value of Lagrange multipliers *β* and *λ* are adjusted by gra-

dient ascent to maximize the following objectives respectively while being clipped to be positive,

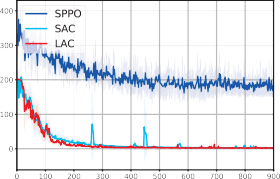
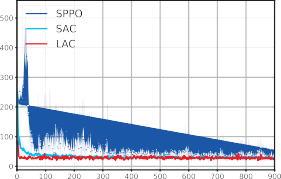
*J*(*β*)= *β*E*D*[log *πθ*(*a|s*)+ *Ht*]

*J*(*λ*)= *λ*E*D*[*Lc*(*sj, fθ*(*sj, s*)) *− Lc*(*s, a*)+ *α*3*c*]

During training, the Lagrange multipliers are updated by

*λ ←* max(0*, λ* + *δ∇λJ*(*λ*)) *β ←* max(0*,β* + *δ∇βJ*(*β*))

where *δ* is the learning rate. The pseudo-code of the proposed algorithm is shown in Algorithm 1.

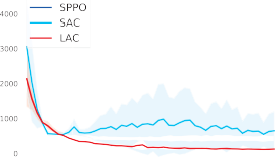
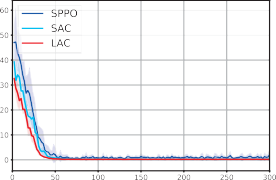
**Algorithm 1:** Lyapunov-Based Actor-Critic (LAC).

**Require:** Maximum episode length *N* and maximum update steps *M*

# repeat

Sample *s*0 according to *ρ*

**for** *t* =1 to *N* **do**

Sample *a* from *πθ*(*a s*) and step forward Observe *sj*, *c* and store (*s, a, c, sj*) in

*D*

*|*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# end for

**for** *i* =1 to *M* **do**

*D*

Sample mini-batches of transitions from and

update *Lc*, *π*, Lagrange multipliers with (7), (14)

# end for

**until** (11) is satisfied

1. EXPERIMENT

In this section, we illustrate five simulated robotic control problems to demonstrate the general applicability of the pro- posed method. First of all, the classic control problem of Cart- Pole balancing from control and RL literature [36] is illustrated. Then, we consider more complicated high-dimensional con- tinuous control problems of 3D robots, e.g., HalfCheetah and FetchReach, using MuJoCo physics engine [37], a multi-joint Swimmer robot [38], and a full quadruped (Minitaur) simulated by PyBullet platform [39]. Last, we extend our approach to control autonomous systems in the cell, i.e., molecular synthetic biological gene regulatory networks (GRN). Specifically, we consider the problem of reference tracking for two GRNs [40]. The proposed method is evaluated for the following aspects:

. Convergence: does the proposed training algorithm con-

verge with random parameter initialization and does the stability condition (2) hold for the learned policies;

. Performance: can the goal of the task be achieved or the

cumulative cost be minimized;

. Stability: if (2) hold, are the closed-loop systems stable

indeed and generating stable state trajectories;

. Robustness: how do the trained policies perform when

faced with uncertainties unseen during training, such as parametric variation and external disturbances;

. Generalization: can the trained policies generalize to fol-

low reference signals that are different from the one seen during training.

We compare our approach with soft actor-critic (SAC) [33], one of the state-of-the-art actor-critic algorithms that outperform a series of RL methods such as DDPG [19], PPO [42] on the continuous control benchmarks. The variant of safe proximal policy optimization (SPPO) [16], a Lyapunov-based method, is also included in the comparison. The original SPPO is developed to deal with constrained MDP, where safety constraints exist. In our experiments, we modify it to apply the Lyapunov constraints on the MDP tasks and see whether it can achieve the same stability guarantee as LAC. In CartPole, we also compare with the linear quadratic regulator (LQR), a classical model-based optimal control method for stabilization. For both algorithms, the hyperparameters are tuned to reach their best performance.

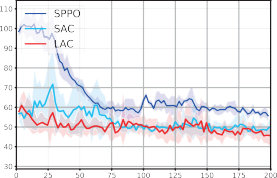
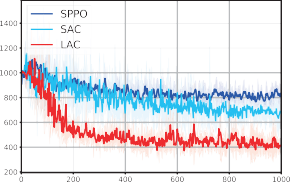
 



Fig. 1. Cumulative control performance comparison. The Y-axis indicates the total cost during one episode and the X-axis indicates the total time steps in thousand. The shadowed region shows the 1-SD confidence interval over 10 random seeds. Across all trials of training, LAC converges to stabilizing solution with comparable or superior performance compared with SAC and SPPO.

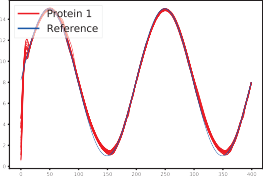
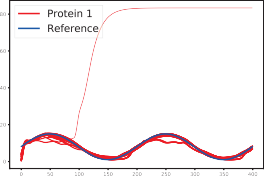
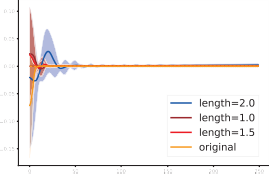
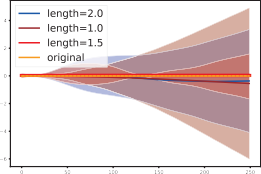
The outline of this section is as follows. In Section V-A, the performance, and convergence of LAC are demonstrated and compared with SAC. In Section V-B, a straight forward demon- stration of stability is made by comparing with the baseline method. In Section V-C, the ability of generalization and robust- ness of the trained policies are evaluated and analyzed. Finally, in Section V-D, we show the influence of choosing different Lyapunov candidates upon the performance and robustness of trained policies.

The hyperparameters of LAC and the detailed experiment setup are deferred to Appendix [43]. The code for reproduction can be found in our GitHub repository.1

1. *Performance*

In each task, both LAC, SAC, and SPPO are trained 10 times with random initialization, average total cost, and its variance during training are demonstrated in Fig. 1. In the examples (a)-(c) and (e), SAC and LAC perform comparably in terms of the total cost at convergence and the speed of convergence, while SPPO could converge in Cartpole, FetcheReach, and Swimmer. In GRN and CompGRN (see Fig. 1(d) and Fig. S9(b) in the supplementary material), SAC is not always able to find a policy that is capable of completing the control objective, resulting in the bad average performance. In the Minitaur example (see Fig. 1(f)), SAC and SPPO can only converge to suboptimal

1<https://github.com/hithmh/Actor-critic-with-stability-guarantee>

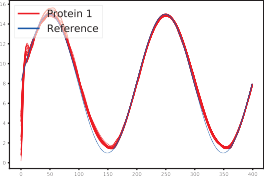
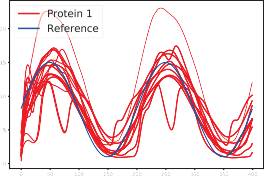
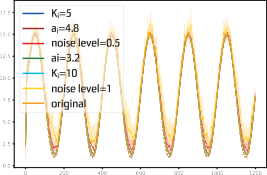
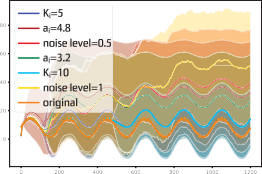
   

Fig. 2. State trajectories over time under policies trained by LAC and SAC in the GRN and CompGRN. In each experiment, the policies are tested over 20 random initial states and all the resulting trajectories are displayed above. The X-axis indicates the time and Y-axis shows the concentration of the target protein— Protein 1.

solutions. On the contrary, LAC performs stably regardless of the random initialization. As shown in Fig. 1, LAC converges stably in all experiments.

1. *Evaluation of Stability*

In this part, further comparison between the stability-assured method (LAC) and that without such guarantee (SAC) is made, by demonstrating the closed-loop system dynamic with the trained policies. A distinguishing feature of the stability assured policies is that it can force and sustain the state or tracking error to zero. This could be intuitively demonstrated by the state trajectories of the closed-loop system.

We evaluated the trained policies in the GRN and CompGRN and the results are shown in Fig. 2. In our experiments, we found that the LAC agents stabilize the systems well. All the state trajectories converge to the reference signal eventually (see Fig. 2(a) and (c)). On the contrary, without stability guarantee, the state trajectories either diverge (see Fig. 2(b)), or continu- ously oscillate around the reference trajectory (see Fig. 2(d)).

1. *Empirical Evaluation on Robustness and Generalization*

It is well-known that over-parameterized policies are prone to become overfitted to a specific training environment. The ability of generalization is the key to the successful implementation of RL agents in an uncertain real-world environment. In this part, we first evaluate the robustness of policies in the presence of parametric uncertainties and process noise. Then, we test the ro- bustness of controllers against external disturbances. Finally, we evaluate whether the policy is generalizable by setting different reference signals. To make a fair comparison, we removed the policies that did not converge in SAC and only evaluate the ones that perform well during training. During testing, we found that SPPO appears to be prone to variations in the environment, thus the evaluation results are contained in Appendix [43].

Fig. 3. LAC and SAC agents in the presence of dynamic uncertainties. The solid line indicates the average trajectory and shadowed region for the 1-SD confidence interval. In (a) and (b), the pole length is varied during the inference. In (c) and (d), three parameters are selected to reflect the uncertainties in gene expression. The X-axis indicates the time and Y-axis shows the angle of the pole in (a), (b) and concentration of target protein in (c), (d), respectively. The dashed line indicates the reference signal. The line in orange indicates the dynamic in the original environment. For each curve, only the noted parameter is different from the original setting.

* 1. *Robustness to Dynamic Uncertainties:* In this part, during the inference, we vary the system parameters and introduce process noises in the model/simulator to evaluate the algorithm’s robustness. In CartPole, we vary the length of pole *l*. In GRN, we vary the promoter strength *ai* and dissociation rate *Ki*. Due to stochastic nature in gene expression, we also introduce

uniformly distributed noise ranging from [ *δ, δ*] (we indicate the

*−*

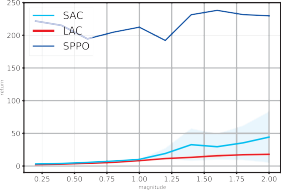
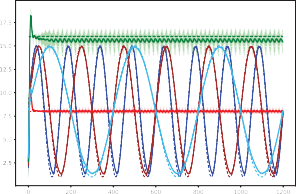
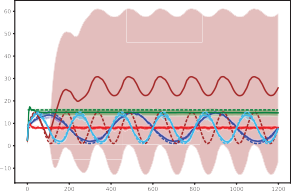
*noise level* by *δ*) to the dynamic of GRN. The state trajectories

of the closed-loop system under LAC and SAC agents in the varied environment are demonstrated in Fig. 3.

As shown in Fig. 3(a), (c), the policies trained by LAC are very robust to the dynamic uncertainties and achieve high tracking precision in each case. On the other hand, though SAC performs well in the original environment (Fig. 3(b), (d)), it fails in all of the varied environments.

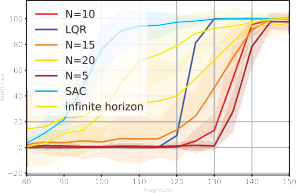
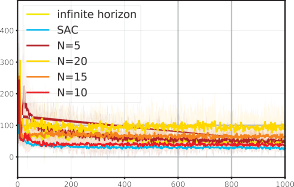
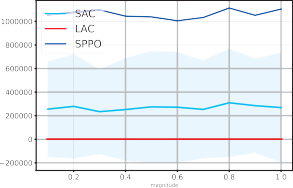
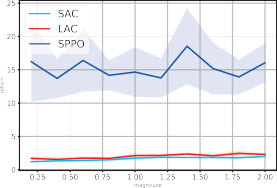
* 1. *Robustness to Disturbances:* An inherent property of a stable system is to recover from perturbations such as external forces and wind. To show this, we introduce periodic external disturbances with different magnitudes in each environment and observe the performance difference between policies trained by LAC and SAC. We also include LQR as the model-based base- line. In CartPole, the agent may fall over when interfered by an external force, ending the episode in advance. Thus in this task, we measure the robustness of controllers through the death-rate, i.e., the probability of falling over after being disturbed. For other tasks where the episodes are always of the same length and we measure the robustness of controller by the variation in the cumulative cost. Under each disturbance magnitude, the policies are tested for 100 trials and the performance is shown in Fig. 4.

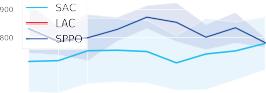
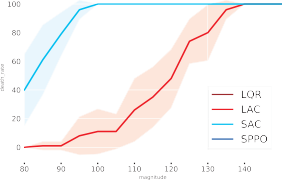
As shown in Fig. 4, the controllers trained by LAC outperform SAC and LQR to a great extent in CartPole and GRN (lower death rate and cumulative cost). In HalfCheetah, SAC and LAC

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

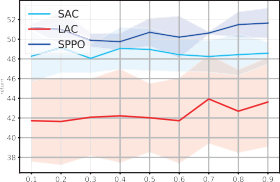


Fig. 5. State trajectories under policies trained by LAC and SAC when tracking different reference signals. The solid line indicates the average trajectory and shadowed region for the 1-SD confidence interval. The X-axis indicates the time and Y-axis shows the concentration of protein to be controlled. Dashed lines in different colors are the different reference signals: sinusoid with a period of 150 (brown); sinusoid with a period of 200 (sky-blue); sinusoid with a period of 400 (blue); constant reference of 8 (red); constant reference of 16 (green).





Fig. 4. Performance of LAC, SAC, SPPO and LQR in the presence of persis- tent disturbances with different magnitudes. The X-axis indicates the magnitude of the applied disturbance. The Y-axis indicates the death rate in CartPole

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

(a) and the cumulative cost in other examples (b) (d). All of the trained policies are evaluated for 100 trials in each setting.

are both robust to small external disturbances while LAC is more reliable to larger ones. In FetchReach, SAC and LAC perform reliably across all of the external disturbances. The difference between SAC and LAC becomes obvious in GRN, Swimmer, and Minitaur, where the dynamics are more vulnerable to the external disturbances. In all of the experiments, SPPO agents could hardly sustain any external disturbances.

* 1. *Generalization Over Different Tracking References:* In this part, we introduce four different reference signals that are unseen during training in the GRN: sinusoids with periods of 150 (brown) and 400 (blue), and the constant reference of 8 (red) and 16 (green). We also show the original reference signal used for training (skyblue) as a benchmark. Reference signals are indicated in Fig. 5 by the dashed line in respective colors. All of the trained policies are tested for 10 times with each reference signal. The average dynamics of the target protein are shown in Fig. 5 with the solid line, while the variance of dynamic is indicated by the shadowed area.

As shown in Fig. 5, the policies trained by LAC could gener- alize well to follow previously unseen reference signals with low variance (dynamics are very close to the dashed lines), regardless of whether they share the same mathematical form with the one used for training. On the other hand, though SAC tracks the original reference signal well after the training trials without convergence being removed (see the sky-blue lines), it is still unable to follow some of the reference signals (see the brown line) and possesses larger variance than LAC.

Fig. 6. Influence of different Lyapunov candidates. In (a), the Y-axis indicates cumulative cost during training and the X-axis indicates the total time steps in thousand. (b) shows the death-rate of policies in the presence of instant impulsive force *F* ranging from 80 to 150 Newton.

1. *Influence of Different Lyapunov Candidates*

As an independent interest, we evaluate the influence of choosing different Lyapunov candidates in this part. First, we adopt candidates of different time horizon *N*

*∈*

5*,* 10*,* 15*,* 20*,* to train policies in the CartPole example

*{ ∞}*

and robustness. Here, *N* = implies using value function and compare their performance in terms of cumulative cost as the Lyapunov candidate. Both of the Lyapunov critics are

*∞*

impulsive force *F* at 100th instant and observe the death-rate of parameterized as (6). For evaluation of robustness, we apply an trained policies. The results are demonstrated in Fig. 6.

As shown in Fig. 6, both choices of Lyapunov candidates converge fast and achieve comparable cumulative cost at con- vergence. However, in terms of robustness, the choice of *N* plays an important role. As observed in Fig. 6(b), the robustness of the controller decreases as the time horizon *N* increases. Besides, it is interesting that LQR is more robust than SAC when faced with instant impulsive disturbance.

1. CONCLUSIONS AND DISCUSSIONS

In this letter, we proposed a data-based approach for analyz- ing the stability of discrete-time nonlinear stochastic systems modeled by Markov decision process, by using the classic Lya- punov’s method in control theory. By employing the stability condition as a critic, an actor-critic algorithm is proposed to learn a controller/policy to ensure the closed-loop stability in stabilization and tracking tasks. We evaluated the proposed method in various examples and show that our method achieves

not only comparable or superior performance compared with the state-of-the-art RL algorithm but also outperforms impressively in terms of robustness to uncertainties such as model parameter variations and external disturbances. For future work, it might be interesting to extend this method to constrained Markov decision process in which state and action constraints are considered. Also, to quantify the robustness induced by the stability will be investigated.

REFERENCES

1. J. Kober, J. A. Bagnell, and J. Peters, “Reinforcement learning in robotics: A survey,” *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1238–1274, 2013.
2. J. Peters and S. Schaal, “Reinforcement learning of motor skills with policy gradients,” *Neural Netw.*, vol. 21, no. 4, pp. 682–697, 2008.
3. S. Löckel, J. Peters, and P. Van Vliet, “A probabilistic framework for imitating human race driver behavior,” *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2086–2093, Apr. 2020.
4. Y. Zhu *et al.*, “Target-driven visual navigation in indoor scenes using deep reinforcement learning,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 3357–3364.
5. S. Gu, E. Holly, T. Lillicrap, and S. Levine, “Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 3389–3396.
6. J.-J. E. Slotine, *et al.*, *Applied Nonlinear Control*. vol. 199. Englewood Cliffs, NJ, USA: Prentice-Hall, 1991.
7. M. Vidyasagar, *Nonlinear Systems Analysis*. vol. 42. Philadelphia, PA, USA: SIAM, 2002.
8. A. M. Lyapunov, “The general problem of the stability of motion (in Russian),” Ph.D. Dissertation, Univ. Kharkov, Kharkiv, Ukraine, 1892.
9. K. J. A˚ ström and B. Wittenmark, *Adaptive Control*. Chelmsford, MA, US: Courier Corporation, 1989.
10. D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, “Constrained model predictive control: Stability and optimality,” *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.
11. M. Corless and G. Leitmann, “Continuous state feedback guaranteeing uniform ultimate boundedness for uncertain dynamic systems,” *IEEE Trans. Autom. Control*, vol. AC-26, no. 5, pp. 1139–1144, Oct. 1981.
12. A. Thowsen, “Uniform ultimate boundedness of the solutions of uncertain dynamic delay systems with state-dependent and memoryless feedback control,” *Int. J. Control*, vol. 37, no. 5, pp. 1135–1143, 1983.
13. L. Bus¸oniu, T. de Bruin, D. Tolic´, J. Kober, and I. Palunko, “Reinforcement learning for control: Performance, stability, and deep approximators,” *Annu. Rev. Control*, vol. 46, pp. 8–28, 2018.
14. D. Gorges, “Relations between model predictive control and reinforcement learning,” *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4920–4928, 2017.
15. Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, “A lyapunov-based approach to safe reinforcement learning,” *Adv. Neural Inf. Process. Syst.*, pp. 8092–8101, 2018.
16. Y. Chow, O. Nachum, A. Faust, M. Ghavamzadeh, and E. Duenez- Guzman, “Lyapunov-based safe policy optimization for continuous con- trol,” 2019, *arXiv:1901.10031*.
17. R. Postoyan, L. Bus¸oniu, D. Nešic´, and J. Daafouz, “Stability analysis of discrete-time infinite-horizon optimal control with discounted cost,” *IEEE Trans. Autom. Control*, vol. 62, no. 6, pp. 2736–2749, Jun. 2017.
18. J. J. Murray, C. J. Cox, and R. E. Saeks, “The adaptive dynamic program- ming theorem,” in *Stability and Control of Dyn. Syst. With Appl.*, Berlin, Germany: Springer, 2003, pp. 379–394.
19. T. P. Lillicrap *et al.*, “Continuous control with deep reinforcement learn- ing,” 2015, *arXiv:1509.02971*.
20. F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, “Safe model- based reinforcement learning with stability guarantees,” in *Proc. Advances Neural Inf. Process. Syst.*, 2017, pp. 908–918.
21. L. Shaikhet, “Necessary and sufficient conditions of asymptotic mean square stability for stochastic linear difference equations,” *Appl. Math. Lett.*, vol. 10, no. 3, pp. 111–115, 1997.
22. P. Bolzern, P. Colaneri, and G. De Nicolao, “Markov jump linear systems with switching transition rates: Mean square stability with dwell-time,” *Automatica*, vol. 46, no. 6, pp. 1081–1088, 2010.
23. V. I. Vorotnikov, “Partial stability and control: The state-of-the-art and de- velopment prospects,” *Autom. Remote Control*, vol. 66, no. 4, pp. 511–561, 2005.
24. W. M. Haddad *et al.*, “Finite-time partial stability and stabilization, and optimal feedback control,” *J. Franklin Inst.*, vol. 352, no. 6, pp. 2329–2357, 2015.
25. R. S. Sutton, H. R. Maei, and C. Szepesvári, “A convergent *o*(*n*) temporal-

difference algorithm for off-policy learning with linear function approx- imation,” in *Proc. Advances Neural Inf. Process. Syst.*, 2009, pp. 1609– 1616.

1. N. Korda and P. La, “On td (0) with function approximation: Concentration bounds and a centered variant with exponential convergence,” in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 626–634.
2. J. Bhandari, D. Russo, and R. Singal, “A finite-time analysis of temporal difference learning with linear function approximation,” *Conf. Learn. Theory*, 2018, pp. 1691–1692.
3. S. Zou, T. Xu, and Y. Liang, “Finite-sample analysis for sarsa with linear function approximation,” in *Proc. Advances Neural Inf. Process. Syst.*, 2019, pp. 8665–8675.
4. E. Boukas and Z. Liu, “Robust stability and h/sub/spl infin//control of discrete-time jump linear systems with time-delay: An LMI approach,” in *Proc. 39th IEEE Conf. Decis. Control*, 2000, vol. 2, pp. 1527–1532.
5. S. Sastry, *Nonlinear Systems: Analysis, Stability, and Control*. vol. 10. Berlin, Germany: Springer, 2013.
6. H. L. Royden, *Real Analysis*. Meerut, Uttar Pradesh, India: Krishna Prakashan Media, 1968.
7. S. P. Meyn and R. L. Tweedie, *Markov Chains and Stochastic Stability*. Berlin, Germany: Springer, 2012.
8. T. Haarnoja *et al.*, “Soft actor-critic algorithms and applications,” 2018,

*arXiv:1812.05905*.

1. D. Q. Mayne and H. Michalska, “Receding horizon control of nonlin- ear systems,” *IEEE Trans. Autom. Control*, vol. 35, no. 7, pp. 814–824, Jul. 1990.
2. J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, “High- dimensional continuous control using generalized advantage estimation,” 2015, *arXiv:1506.02438*.
3. A. G. Barto, R. S. Sutton, and C. W. Anderson, “Neuronlike adaptive elements that can solve difficult learning control problems,” *IEEE Trans. Syst., man, Cybern.*, vol. SMC-13, no. 5, pp. 834–846, Sep./Oct. 1983.
4. E. Todorov, T. Erez, and Y. Tassa, “MuJoCo: A physics engine for model- based control,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 5026–5033.
5. Y. Tassa *et al.*, “Deepmind control suite,” 2018, *arXiv:1801.00690*.
6. E. Coumans and Y. Bai, “Pybullet, a python module for physics simulation for games, robotics and machine learning,” *GitHub Repository*, 2016.
7. M. B. Elowitz and S. Leibler, “A synthetic oscillatory network of transcrip- tional regulators,” *Nature*, Nature Publishing Group, vol. 403, no. 6767, pp. 335–338, 2000.
8. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” 2017, *arXiv:1707.06347*.
9. M. Han, L. Zhang, J. Wang, and W. Pan, “Actor-critic reinforcement learning for control with stability guarantee,” 2020, *arXiv:2004.14288*.
10. G. Brockman *et al.*, “Openai gym,” 2016, *arXiv:1606.01540*.
11. N. Strelkowa and M. Barahona, “Switchable genetic oscillator operating in quasi-stable mode,” *J. Roy. Soc. Interface*, vol. 7, no. 48, pp. 1071–1082, 2010.
12. A. Sootla, N. Strelkowa, D. Ernst, M. Barahona, and G.-B. Stan, “On periodic reference tracking using batch-mode reinforcement learning with application to gene regulatory network control,” in *Proc. 52nd IEEE Conf. Decis. Control*, 2013, pp. 4086–4091.
13. Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu, “Advances in optimizing recurrent networks,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2013, pp. 8624–8628.
14. I. Bello, B. Zoph, V. Vasudevan, and Q. V. Le, “Neural optimizer search with reinforcement learning,” in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, vol. 70, pp. 459–468.
15. Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, “Dueling network architectures for deep reinforcement learning,” *Int. Conf. Mach. Learn.*, 2016, pp. 1995–2003.