SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

April 2023, Vol. 66 142102:1–142102:15
https://doi.org/10.1007/s11432-022-3539-8

State space representation and phase analysis of gradient descent optimizers

Biyuan YAO[1], Guiqing LI[1*] & Wei WU[2]

[1]*School of Computer Science and Engineering, South China University of Technology, Guangzhou* 510006*, China;*
[2]*School of Computer, Wuhan University, Wuhan* 430072*, China*

Received 24 January 2022/Revised 27 April 2022/Accepted 23 June 2022/Published online 27 March 2023

Abstract Deep learning has achieved good results in the field of image recognition due to the key role of the optimizer in a deep learning network. In this work, the optimizers of dynamical system models are established, and the influence of parameter adjustments on the dynamic performance of the system is proposed. This is a useful supplement to the theoretical control models of optimizers. First, the system control model is derived based on the iterative formula of the optimizer, the optimizer model is expressed by differential equations, and the control equation of the optimizer is established. Second, based on the system control model of the optimizer, the phase trajectory process of the optimizer model and the influence of different hyperparameters on the system performance of the learning model are analyzed. Finally, controllers with different optimizers and different hyperparameters are used to classify the MNIST and CIFAR-10 datasets to verify the effects of different optimizers on the model learning performance and compare them with related methods. Experimental results show that selecting appropriate optimizers can accelerate the convergence speed of the model and improve the accuracy of model recognition. Furthermore, the convergence speed and performance of the stochastic gradient descent (SGD) optimizer are better than those of the stochastic gradient descent-momentum (SGD-M) and Nesterov accelerated gradient (NAG) optimizers.

Keywords optimizer, control model, phase trajectory, parameter adjustment, classification, dynamic performance

Citation Yao B Y, Li G Q, Wu W. State space representation and phase analysis of gradient descent optimizers. Sci China Inf Sci, 2023, 66(4): 142102, https://doi.org/10.1007/s11432-022-3539-8

1 Introduction

Deep learning (DL) is a branch of machine learning that is widely used in image recognition [1], classification, and control. Optimizers are important for the optimization of network models and for obtaining parameterized expressions for specific problems in DL. Optimizers are the basic tool of the DL models, and optimizer models based on gradient descent algorithms have developed quickly [2]. However, the theoretical and application support of gradient optimizer algorithms is still mainly from a data-driven perspective. Therefore, a more systematic and comprehensive analysis of the principles of gradient optimizer algorithms requires further theoretical study. In the training stage of a DL model, the procedures for selecting parameters, such as the learning rate, batch size, and step size, vary for specific problems, and the model requires sufficient training to achieve a good convergence effect.

Stochastic gradient descent (SGD) is the basic method for the parameter optimization of the DL network models [3]. Many evolutionary forms, such as stochastic gradient descent-momentum (SGD-M) [4] and Nesterov accelerated gradient (NAG) [5], have been proposed and used in real applications. A theoretical proportional-integral-derivative (PID) controller was used to simulate the stochastic optimizer of a DL model by An et al. [6] by combining the error calculation in the feedback control system with a gradient calculation. The partial derivative of the network output was used as an error variable, which revealed the close connection between the error-based PID control and the gradient method. Wang et al. [7] introduced the relationship between PID controllers, stochastic optimizers (e.g., SGD, SGD-M, and

* Corresponding author (email: ligq@scut.edu.cn)

NAG), and their variables. According to the model structure of the stochastic optimizers [6, 7], Wu et al. [8] explained the learning problem of the gradient optimizer in DL models using stochastic differential equations. Control theory has been used to explain the convergence of deep learning models and system output stability. For example, An et al. [6] and Wang et al. [7] used PID controllers, and Wu et al. [8] used time-domain analysis. We propose a new iterative algorithm interpretation scheme in which the phase plane method is used to analyze the influences of different hyperparameters of optimizers on the dynamic performances of second-order systems. This approach solves the parameter setting problem of optimizers, which was not explored in An et al. [6] and Wang et al. [7]. In addition, we propose to use a dynamic equation to determine the values of all the independent variables of an optimizer system, which was not addressed in [8]. The establishment of control models of optimizers and the parameter setting problem are addressed in this paper. The results will be beneficial for developing theoretical control models of optimizers.

The optimal iterative algorithms (e.g., SGD, SGD-M, and NAG) were reinterpreted using control theory, which has significance in the field of cybernetics, and the convergence characteristics of the iterative algorithms were analyzed. The main contributions of this work are as follows.

(1) By deriving the differential equation of the DL optimizer, the control equation of the controller was established, and the mathematical expression method of the controller was solved, which provides a basic tool for dynamically analyzing the input-output relationship of the model.

(2) The phase trajectory method of a typical optimizer model was proposed for the first time, and the influences of different hyperparameter settings on the performance of the learning model were analyzed, providing theoretical support for the rational design of the optimizer and the hyperparameters.

(3) The dynamic learning process of the optimizer control model was analyzed by an example. The effects of different hyperparameters and different optimizers on the dynamic performance of the system were verified in the optimization process of the DL model.

The remainder of this paper is organized as follows. Section 2 introduces the work related to the optimizers. Section 3 introduces the relevant basic control theory, including the transfer function, dynamics equation, PID controller, and phase trajectories. In Section 4, the transfer functions of the SGD, SGD-M, and NAG optimizers are obtained using an iterative formula, and the dynamics equation of the system is established. Then, the dynamic performance of the system is analyzed based on the phase plane trajectories of the optimizers. Section 5 establishes the control models of the optimizers, and classification experiments on the MNIST and CIFAR-10 datasets are presented to verify the dynamic analysis process of different optimizers and different hyperparameters. Section 6 presents the conclusion.

2 Related work

SGD is used to realize the random approximation of monotonic functions [3, 9], which is the basis of the gradient descent method. SGD [10, 11] is an iterative, "online" objective function optimization method, in which an approximate gradient sequence is obtained by random sampling from a training dataset. The perceptron network model [12] uses SGD for error iteration, and the multi-layer perceptron and other network models arranged in layers use SGD to calculate the neuron weights [13]. SGD is introduced into the neural network model, and an artificial intelligence model is used to adjust the parameters [11]. The advantage of DL is that it can realize classification and recognition through its huge learning capacity with the help of the SGD's linear scalability and low reasoning complexity [14]. Optimizer models based on the SGD algorithm have developed quickly. However, there are still some problems. When the data satisfies a semi-positive-definite constraint, SGD must adjust the sampling scale to reduce the number of iterations [15]. Luo et al. [16] proposed the momentum-incorporated parallel SGD algorithm to address the low efficiency and scalability of the SGD algorithm in the processing of big data.

Many high-order optimization methods and the momentum improvement forms of SGD methods have been proposed to improve the training speed of SGD. Ruder [17] introduced common SGD algorithms: SGD-M, NAG, and other optimized asynchronous SGD algorithms. Ding et al. [18] proposed an SGD-M optimization method of dynamic pruning that divided the parameters into two parts in the training iteration and updated them using different rules to reduce the network complexity. Wang et al. [19] proposed a decelerating momentum framework to implement a periodic momentum updating strategy. Jarek [20] proposed the variable-momentum method to downsize the variance and reduce the training interference caused by improper pre-designed hyperparameters. The NAG method, which improves the

momentum information to improve the original gradient, can strengthen this process [5], and it uses an advancing step to move the update point of the gradient information forward to achieve the next step of the gradient information approximation. Aleksandar et al. [21] deduced the NAG method and proposed an iterative regular gradient descent framework structure. Luo et al. [22] combined an adaptive method with the learning rate of manual SGD to achieve a balance between the generalization ability and training speed. Considering the improper application of the NAG method in extreme cases, Zeyuan [23] proposed the Katyusha momentum method to improve the convergence speed of the algorithm for offline convex optimization problems and proposed the Natasha II method to cross part of the saddle-point region.

The two core aspects of the DL optimizer are the gradient and the learning rate (LR). The former determines the direction of the parameter updating, and the latter determines the degree of parameter updating. The selection of the LR is very important, especially for large-scale samples. The rate selection requires many training samples and the expansion of the SGD algorithm to improve its convergence speed and prediction accuracy [24]. Luo et al. [25] studied eight extended SGD algorithms to improve the convergence speed of a generative model and the prediction accuracy with missing data. SGD does not proceed in the direction of optimization in every iteration. For this reason, Lei and Tang [26] developed a new SGD LR by giving bounds on the computational and statistical errors with high probability. Meta-learning helps to set an appropriate LR for DL optimizers. Meta-learning uses previously learned knowledge in a data-driven way to enable artificial intelligence to learn new tasks autonomously and quickly. Amid et al. [27] proposed an algorithm to adjust the LR based on the idea of meta-learning. In order to select an appropriate optimizer for new task optimization via gradient descent, Andrychowicz et al. [28] made the long short-term memory network (LSTM) optimizer to use the previous task learning to predict the gradient, and replaced the traditional optimizer (e.g., SGD) with the LSTM network. In view of the fact that the appropriate LR sought by the traditional method is not suitable for a specific target problem, Xiong et al. [29] constructed a directed graph for the underlying neural network of the target problem, and used a graph neural network to set the appropriate LR. To solve the problem that LR is not suitable for practical non-convex optimization due to dynamic and diversified training, Shu et al. [30] used the MLR-SNet of meta-learning to learn a dynamic LR that was adapted to DNN training.

Construction of a control equation of the gradient optimizer is beneficial for understanding the dynamic performance parameters [31]. The classical control model uses the characteristics of a transfer function in the time and frequency domains and introduces Lyapunov stability theory to study the state changes [32–35]. Wang et al. [36] described that the commonly used control system models are dynamic equations and transfer function models. Bhaya and Kaszkurewicz [37] introduced the analysis and design of iterative numerical methods from a control perspective. Chen et al. [38] analyzed GD, SGD-M, and NAG from the perspective of control systems and found that they can be expressed as feedback control problems for tracking extreme points to provide a design method of high-order transfer functions. Liu et al. [39] introduced two stochastic control techniques to ensure the convergence of the GD and SGD algorithms. Under the influence of uncertain factors, such as the data loss and the unknown structural parameters of the system, Xu [40] performed systematic analysis of the performance of the closed-loop system.

3 Preliminary theory

3.1 Transfer function and dynamics equation

Definition 1. Under a zero initial condition, the ratio of the pull-type transformation of the system output variable to the pull-type transformation of the input variable is called the transfer function of the system [40], denoted by $G(s) = \frac{Y(s)}{U(s)}$, where $Y(s)$ and $U(s)$ are the pull-type transformations of the output variables and input variables, respectively. An open-loop transfer function expresses the functional relationship between the ratio of the output to the input of an open-loop system and the frequency. For a single-variable linear time-invariant system, its equation of motion is an n-order linear differential equation with constant coefficients, expressed as follows:

$$a_n \cdot \frac{\mathrm{d}^n y(t)}{\mathrm{d}t^n} + a_{n-1} \cdot \frac{\mathrm{d}^{n-1} y(t)}{\mathrm{d}^{n-1}} + \cdots + a_1 \cdot \frac{\mathrm{d}(t)}{\mathrm{d}t} + a_0 \cdot y(t) = b_m \cdot \frac{\mathrm{d}^m u(t)}{\mathrm{d}t^m} + b_{m-1} \cdot \frac{\mathrm{d}^{m-1} u(t)}{\mathrm{d}t^{m-1}} + \cdots + b_1 \cdot \frac{\mathrm{d}u(t)}{\mathrm{d}t} + b_0 \cdot u(t), \tag{1}$$

where $u(t)$ and $y(t)$ are input and output variables, respectively. $n \geqslant m$ is a positive integer, and n is the order of the system. The derivative terms of $y(t)$ and $u(t)$ are arranged on the left and right sides of the

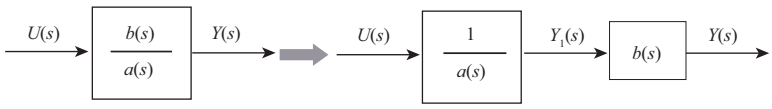

Figure 1 Decomposition of the transfer function solution.

equals sign in descending order. a_i $(i = 0, 1, 2, \ldots, n)$ represents the structural parameters of the system, b_j $(j = 0, 1, 2, \ldots, m)$ represents the structural parameters of the input function, and they are all constant coefficients. The initial values of $y(t)$ and its derivatives are $y(0) = y_0$, $\frac{\mathrm{d}y(0)}{\mathrm{d}t} = y'$, $\ldots$, $\frac{\mathrm{d}^{n-1}y(0)}{\mathrm{d}t^{n-1}} = y_0^{(n-1)}$. According to the definition of the transfer function and (1), let $U(s) = L[u(t)]$ and $Y(s) = L[y(t)]$. The incremental differential equation describes the essence of the dynamic characteristics of the system, and Eq. (1) is an incremental equation that omits the Δ symbol. Thus,

$$\begin{aligned} \left.\frac{\mathrm{d}^n y(t)}{\mathrm{d}t^n}\right|_{t=0} &= \left.\frac{\mathrm{d}^{n-1} y(t)}{\mathrm{d}t^{n-1}}\right|_{t=0} = \cdots = \left.\frac{\mathrm{d}y(t)}{\mathrm{d}t}\right|_{t=0} = y(0) = 0, \\ \left.\frac{\mathrm{d}^m u(t)}{\mathrm{d}t^m}\right|_{t=0} &= \left.\frac{\mathrm{d}^{m-1} u(t)}{\mathrm{d}t^{m-1}}\right|_{t=0} = \cdots = \left.\frac{\mathrm{d}u(t)}{\mathrm{d}t}\right|_{t=0} = u(0) = 0. \end{aligned} \tag{2}$$

Eq. (2) is used to find the Laplace transform of both sides of (1). According to the differential theorem, when the initial condition is 0, the Laplace transform is

$$\left(a_n s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0\right) Y(s) = \left(b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0\right) U(s). \tag{3}$$

Therefore, the corresponding transfer function is

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0}{a_n s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0}. \tag{4}$$

If the value of $H(s)$ is 1 in the transfer function for unit negative feedback, the closed-loop transfer function is

$$\phi(s) = \frac{G_1(s) \cdot G_2(s)}{(1 + G_1(s) \cdot G_2(s)) \cdot H(s)} = \frac{G(s)}{1 + G(s)} = \frac{b(s)}{a(s)}. \tag{5}$$

Eq. (5) can be written as

$$Y(s) = \frac{b(s)}{a(s)} \cdot U(s) = b(s) \cdot \left[\frac{1}{a(s)} \cdot U(s)\right]. \tag{6}$$

Figure 1 shows the decomposition of the transfer function of (6). According to Figure 1, we have the following two cases for discussion.

Case 1. For the inner layer of the transfer function,

$$Y_1(s) = \frac{1}{a(s)} \cdot U(s). \tag{7}$$

Case 2. For the outer layer of the transfer function,

$$Y(s) = b(s) \cdot Y_1(s) = b(s) \cdot X_1(s). \tag{8}$$

Based on the above theory, SGD, SGD-M, and NAG are all represented transfer functions whose numerator is not one, and we can derive their dynamic equations. Since the transfer function does not consider the dynamic motion of the internal state of the system, it only describes the relationship between the input and output of the system. Therefore, a dynamics equation is introduced to fully reflect the changes of all the independent variables of the system.

Definition 2. At a certain moment t, a set of independent state vectors $\boldsymbol{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^{\mathrm{T}}$ of a system constitutes a set of first-order differential equations. If the input u and output y of the system are established, the output equation of the system is determined. The state equation and output equations are combined to form a complete dynamic description; i.e., the dynamics equations are as follows:

$$\begin{cases} \boldsymbol{x}' = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}, \\ \boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{u}, \end{cases} \tag{9}$$

where $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^{\mathrm{T}}$ is the n-dimensional state vector, $\boldsymbol{y} = [y_1, y_2, \ldots, y_m]^{\mathrm{T}}$ is the m-dimensional output vector, $\boldsymbol{u} = [u_1, u_2, \ldots, u_r]^{\mathrm{T}}$ is the r-dimensional input vector, $\boldsymbol{A}$ is the $(n \times n)$-dimensional system state control matrix, determined by the internal structure of the system, $\boldsymbol{B}$ is the $(n \times r)$-dimensional input matrix, which reflects the application of the system input, $\boldsymbol{C}$ is the $(m \times n)$-dimensional output matrix, and $\boldsymbol{D}$ is $(m \times r)$-dimensional direct transfer matrix. In many systems, the output signal is not directly associated with the input signal, $\boldsymbol{D} = \boldsymbol{0}$, and the state space expression of the optimizer becomes

$$\begin{cases} \boldsymbol{x}' = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}, \\ \boldsymbol{y} = \boldsymbol{C}\boldsymbol{x}. \end{cases} \tag{10}$$

3.2 Proportional-integral-derivative controller

Definition 3. PID control uses a linear combination of proportional, integral, and derivative terms of the deviation to form a control quantity. This control quantity is used to control the controlled object, which is called a PID controller. The output signal of the PID controller is [40]

$$u(t) = K_{\mathrm{p}} \left[e(t) + \frac{1}{\tau_i} \cdot \int_0^t e(t)\mathrm{d}t + \tau_d \frac{\mathrm{d}e(t)}{\mathrm{d}t} \right], \tag{11}$$

where $u(t)$ is the output of the controller, $e(t)$ is the input of the controller, K_{p} is the proportional coefficient, τ_i is the integral time constant, τ_d is the differential time constant, and t is the time interval from the beginning of the adjustment to the output of the current control quantity.

Definition 4 ([41])**.** $f(t)$ is a real function if it satisfies the following: (1) when $t < 0$, $f(t) = 0$; (2) when $t \geqslant 0$, the integral of $f(t)$, $\int_0^\infty f(t)\mathrm{e}^{-st}\mathrm{d}t$, converges in a certain domain of s. Then, the Laplace transform of $f(t)$, $F(s) = \int_0^\infty f(t)\mathrm{e}^{-st}\mathrm{d}t$, denoted by $L[f(t)] = F(s)$, where s is a complex variable.

The differential theorem [42] states that if $L[f(t)] = F(s)$, then $L[\frac{\mathrm{d}f(t)}{\mathrm{d}t}] = \int_0^\infty [\frac{\mathrm{d}f(t)}{\mathrm{d}t}]\mathrm{e}^{-st}\mathrm{d}t = sF(s) - f(0^+)$, where $f(0^+) = f(t)|_{t=0^+}$.

Theorem 1. The transfer function $G(s) = \frac{K_{\mathrm{p}}\tau_d\tau_i s^2 + K_{\mathrm{p}}\tau_i s + K_{\mathrm{p}}}{\tau_i s}$ is obtained from the PID differential equation (11).

Proof. By Definition 4 and the differential theorem, let $L[e(t)] = F(s) = \int_0^\infty f(t)\mathrm{e}^{-st}\mathrm{d}t$. We take the Laplace transform of both sides of the PID differential equation (11) and obtain $L[u(t)] = L\left[K_{\mathrm{p}}e(t)\right] + L[\frac{K_{\mathrm{p}}}{\tau_i} \cdot \int_0^t e(t)\mathrm{d}t] + L[K_{\mathrm{p}}\tau_d \frac{\mathrm{d}e(t)}{\mathrm{d}t}] = K_{\mathrm{p}}F(s)(1 + \frac{1}{\tau_i s} + \tau_d s)$, where L is a symbol that represents the Laplace integral of a function. Due to the differential property of the Laplace transform, $\frac{\mathrm{d}e(t)}{\mathrm{d}t} = s \cdot e(s) - e(0)$, and the transfer function for PID control is

$$G_{\mathrm{PID}}(s) = \frac{Y(s)}{U(s)} = \frac{K_{\mathrm{p}}F(s)(1 + \frac{1}{\tau_i s} + \tau_d s)}{F(s)} = K_{\mathrm{p}}\left(1 + \frac{1}{\tau_i s} + \tau_d s\right) = \frac{K_{\mathrm{p}}\tau_d\tau_i s^2 + K_{\mathrm{p}}\tau_i s + K_{\mathrm{p}}}{\tau_i s}. \tag{12}$$

3.3 Phase trajectory analysis method for control model

The closed-loop transfer function of the system is $G(s) = \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}$, where ξ is the damping coefficient, and ω_n is the angular frequency of the undamped oscillation. The characteristic equation of the system is $s^2 + 2\xi\omega_n s + \omega_n^2 = 0$. The characteristic roots are $s_{1,2} = -\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1}$.

The phase plane method converts the equations of the second-order system motion equations into phase trajectories on the position-velocity planes by a graphical method. This approach can intuitively and accurately reflect the stability and equilibrium state of the system as well as the influence of the initial conditions and parameters on the system motion. Let a second-order ordinary differential equation describe the system as follows:

$$x'' = f\left(x, x'\right), \tag{13}$$

where $f(x, x')$ is a linear or nonlinear function of $x(t)$ and $x'(t)$, x' represents the first derivative with respect to x, and t is a variable parameter. $x(t)$ is the time solution of the system, and it describes the variation process of x with time. By the definition of the derivative, if $x = x_1$, $x' = x_2$, $\frac{\mathrm{d}x_1}{\mathrm{d}t} = x_2$, and $\frac{\mathrm{d}x_2}{\mathrm{d}t} = f\left(x_1, x_2\right)$, then

$$\frac{\mathrm{d}x'}{\mathrm{d}x} = \frac{\mathrm{d}x_2}{\mathrm{d}x_1} = \frac{f\left(x_1, x_2\right)}{x_2}. \tag{14}$$

Definition 5. The plane with rectangular coordinates (x, x') is called the phase plane. Eq. (13) is called the phase trajectory equation. Eq. (14) is called the phase trajectory expression, where $\frac{\mathrm{d}x'}{\mathrm{d}x} = \frac{\mathrm{d}x_2}{\mathrm{d}x_1}$ is the slope of the phase trajectory. The phase plane and the cluster of phase trajectory curves form a phase plane diagram. A certain state of the system corresponds to a point on the phase plane, and the state transition with time corresponds to the movement of a point on the plane. The curve drawn by points in the phase plane over time is called phase trajectories.

4 Dynamics analysis of optimizer

In this section, the content is divided into two parts: first, because the differential equation and transfer function express the relationship between the input and output of the system, they do not indicate the internal structure of the system. Therefore, the transfer function is calculated based on the iterative formula of the optimizer, and the dynamics model of the system is established. This not only yields the complete expression of the internal structure of the system but also maintains the input-output dynamic relationship determined by the original transfer function. Second, the unstable system cannot complete the expected control task, and the primary problem of the system analysis is to improve the stability of the system. Therefore, based on the transfer function control models of the SGD, SGD-M, and NAG, the influence of different hyperparameters on the dynamic performance of the second-order system is analyzed by using the phase plane method. This solves the problem that the stability and dynamics of the system cannot be analyzed because the differential equation of the system is difficult to solve.

4.1 Stochastic gradient descent

The time-domain expression of the SGD is given as follows [7]:

$$\theta_{n+1} = \theta_n - r\frac{\partial L_n}{\partial \theta_n}, \tag{15}$$

where n is the total number of training data points, θ_n represents the network parameter when the parameter is in step n, L is the loss, the gradient $\frac{\partial L_n}{\partial \theta_n}$ is regarded as $e(n)$, and r is the learning rate.

Since $\theta_{n+1} - \theta_n = -r\frac{\partial L_n}{\partial \theta_n}$ and $\frac{\partial L_n}{\partial \theta_n} \approx \int_0^n e(n)\mathrm{d}t$, this indicates that the SGD transfer function is equivalent to the integration loop transfer function of the PID. According to (11), the output of the integration loop is proportional to the integration of the input, and its output signal is $K_{\mathrm{p}} \cdot \frac{1}{\tau_i} \cdot \int_0^t e(t)\mathrm{d}t$ [8]. In the integration loop, the integration loop is a property of the Laplace transform: when $L[e(n)] = F(s)$, $e(n)$ is the unit-impulse loop, and $F(s) = 1$, then $L[\int_0^n e(n)\mathrm{d}n] = \frac{F(s)}{s}$. Therefore, the open-loop transfer function is given by

$$G_{\mathrm{SGD}}(s) = K_0 \cdot \frac{1}{\tau_1} \cdot r \cdot \int_0^n e(n)\mathrm{d}n = K_0 \cdot \frac{1}{\tau_1} \cdot r \cdot \frac{1}{s} = \frac{K_0 r}{\tau_1 s}, \tag{16}$$

where K_0 is the proportional coefficient, s is equivalent to the cumulative error of the model, and τ_1 is the integral constant. The closed-loop transfer function of the SGD system is obtained as follows:

$$\phi(s) = \frac{G_{\mathrm{SGD}}(s)}{1 + G_{\mathrm{SGD}}(s)} = \frac{K_0 r}{\tau_1 s + K_0 r} = \frac{\frac{K_0 r}{\tau_1}}{s + \frac{K_0 r}{\tau_1}}. \tag{17}$$

Theorem 2. The transfer function is a true rational expression of a complex variable s, and the order of its denominator is 1, which defines the SGD as a first order control system, with the state equation given by

$$\begin{cases} x_1' = -\frac{K_0 r}{\tau_1} \cdot x_1 + \frac{u_1}{\tau_1}, \\ y = K_0 \cdot r \cdot x_1, \end{cases}$$

where u_1 is a scalar.

Proof. The numerator of the closed-loop transfer function (17) is not 1. Let $\phi(s) = \frac{K_0 r}{\tau_1 s + K_0 r} = \frac{Y(s)}{U(s)} = \frac{b(s)}{a(s)}$, and its inner layer is $Y_1(s) = \frac{1}{\tau_1 s + K_0 r} \cdot U(s)$. The Laplace transform of the differential equation is $u_1 = \tau_1 y' + K_0 r y$, where u_1 is the input and y is the output. The denominator of the SGD transfer

function is first order, and a new state variable $x_1 = y$ is defined. Therefore, the differential equation is $x_1' = -\frac{K_0 r}{\tau_1} \cdot x_1 + \frac{u_1}{\tau_1}$, and the dynamic equation of the inner layer is

$$\begin{cases} x_1' = -\frac{K_0 r}{\tau_1} \cdot x_1 + \frac{u_1}{\tau_1}, \\ y = x_1. \end{cases}$$

From (8) and (17), the outer layer is $Y(s) = K_0 \cdot r \cdot X_1(s)$, and the dynamic equation of the outer layer is $y = K_0 \cdot r \cdot x_1$. Therefore, the dynamic equations of SGD are

$$\begin{cases} x_1' = -\frac{K_0 r}{\tau_1} \cdot x_1 + \frac{u_1}{\tau_1}, \\ y = K_0 \cdot r \cdot x_1. \end{cases}$$

SGD is a first order system, and the differential equation of the system is $y' + \frac{K_0 r}{\tau_1} \cdot y = 0$. The phase trajectory equation is $y' = -\frac{K_0 r}{\tau_1} \cdot y$. The characteristic root is $s = -\frac{K_0 r}{\tau_1}$. SGD only calculates the gradient based on one random sample at a time, which is fast and can update the model in real time based on new samples, and the parameters are updated frequently. To solve the problem of a slow learning process of the SGD method, the SGD-M method is introduced to accelerate learning. When determining the search direction, SGD only uses the first derivative (gradient) of the objective function, so its descent direction may not be globally optimal. SGD-M uses the second (partial) derivative to obtain a directional method with faster convergence.

4.2 Stochastic gradient descent with momentum = SGD-M

SGD-M originates from the momentum improvement of SGD. The main idea is to introduce a momentum of the accumulated historical gradient information to accelerate the SGD process. To suppress the oscillations of the SGD, SGD-M adds inertia in the gradient descent process. The time-domain expression of the SGD-M is obtained according to the SGD method:

$$\theta_{n+1} = \theta_n - r \cdot \frac{\partial L_n}{\partial \theta_n} - r \sum_{i=0}^{n-1} \frac{\partial L_i}{\partial \theta_i} \cdot \alpha^{n-i}. \tag{18}$$

Eq. (18) is equivalent to the coaction of the proportional and integral elements. In the last term, the control hyperparameter α is used to control the weights of the historical and current gradients.

The inertial link is a first-order system with a delay to the input signal. The inertia of a system is the ability of the system to maintain its own state in the current state, and the state at this time is $\frac{\mathrm{d}y(t)}{\mathrm{d}t}$. For a directed curve, the direction of the tangent is the direction of the point. Therefore, based on the law of inertia, we obtain $x(t) - y(t) = T \cdot \frac{\mathrm{d}y(t)}{\mathrm{d}t}$. The relationship between the output and input of the inertial link is $T \cdot \frac{\mathrm{d}y(t)}{\mathrm{d}t} + y(t) = x(t)$, $t \geqslant 0$. Then, the transfer function of the inertial link is $G(s) = \frac{Y(s)}{X(s)} = \frac{1}{Ts+1}$, and T is the time constant [43].

Deep network optimization has a high similarity with controllers. They both update based on the loss of the actual output and expected output. In PID control, feedback corresponds to back propagation in the network optimization. The difference is that the controller uses $e(n)$ to calculate updates, while the deep network optimizer determines the update based on the gradient $\frac{\partial L}{\partial \theta}$.

Remark 1. The purpose of DL is to learn a mapping function f that maps input x to output y with parameters θ, that is, $y = f(x, \theta)$. To measure the difference between the deep neural network output and the expected output, a loss function L is introduced. Given the training data, DL can calculate some parameters (weight ω) to analyze the complex relationship between x and y and calculate L. In general, L is defined in terms of the expected output y and the expected output $f(x, \theta)$ to measure whether the target is reached. L distributes the error to each node by computing the gradient of the weight. To minimize L, the network updates its weight θ according to the gradient. Therefore, it is reasonable to associate the error in the control model with the gradient in DL. The process is iterative until L converges or is sufficiently small.

Remark 2. The derivations of the error $e(n)$ and the partial derivative were reported previously [6–8]. The goal of the control system is to continuously measure the output system state through the control unit and update it to the required state. In the feedback control, the output affects the input, and the

controller updates the system state according to $e(n)$. To achieve this goal, the PID controller calculates the error $e(n)$ of each step n and then applies the modified control variable $u(n)$ to the system as a function of the proportional (P), integral (I) and differential (D) terms of $e(n)$. The formula is as follows:

$$u(n) = K_{\mathrm{p}} \cdot e(n) + K_{\mathrm{i}} \cdot \int_0^n e(n)\mathrm{d}n + K_{\mathrm{d}} \cdot \frac{\mathrm{d}}{\mathrm{d}n} e(n), \tag{19}$$

where K_{p} is the proportional coefficient, K_{i} is the integral time constant, and K_{d} is the differential time constant.

In summary, the relationship between SGD-M equation (18) and PID equation (19) is as follows: the current gradient is $\frac{\partial L_n}{\partial \theta_n} \approx e(n)$, and the historical gradient is $\sum_{i=0}^{n-1} \alpha^{n-i} \cdot \frac{\partial L_i}{\partial \theta_i} \approx \int_0^n e(n)\mathrm{d}n$. Therefore, it has two characteristics: (1) the historical gradient and current gradients are selected for weight calculation, that is, there is a certain delay/lag for the current input error signal; (2) a certain attenuation is performed on the integral link, and the iteration is more stable. Based on the SGD method, the SGD-M is equivalent to adding an inertial link, namely, a momentum factor $G_m = \frac{1}{Ts+1}$. With $T = g(\mu)$, the SGD-M controller is a typical second-order system, with the open-loop transfer function of the SGD-M expressed as

$$G_{\text{SGD-M}}(s) = G_{\text{SGD}}(s) \cdot G_m = K_0 \cdot \frac{r}{s} \cdot \frac{1}{g(\mu)s + 1} = \frac{K_0 r}{g(\mu)s^2 + s}, \tag{20}$$

where $g(\mu)$ is the momentum regulation function. Based on the momentum, the following closed-loop transfer function for the SGD-M is derived:

$$\phi(s) = \frac{G_{\text{SGD-M}}(s)}{1 + G_{\text{SGD-M}}(s)} = \frac{K_0 r}{g(\mu)s^2 + s + K_0 r} = \frac{\frac{K_0 r}{g(\mu)}}{s^2 + \frac{1}{g(\mu)} \cdot s + \frac{K_0 r}{g(\mu)}}. \tag{21}$$

Theorem 3. The dynamic process of the SGD-M's controller is controlled by $\omega_n = \sqrt{\frac{K_0 r}{g(\mu)}}$ and $\xi = \frac{1}{\sqrt{4K_0 r g(\mu)}}$, with the dynamics equations

$$\begin{cases} \begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{K_0 r}{g(\mu)} & -\frac{1}{g(\mu)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u, \\ y = \begin{pmatrix} \frac{K_0 r}{g(\mu)} & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \end{cases}$$

where u is the two-dimensional input vector.

Proof. According to the definition of the closed-loop transfer function of the SGD-M, the characteristic equation is $s^2 + \frac{1}{g(\mu)} \cdot s + \frac{K_{\mathrm{d}} r}{g(\mu)} = 0$. The second-order response characteristics are controlled by K_0, r, and $g(\mu)$. According to the characteristic equation of a second-order control system and the method of undetermined coefficients, we obtain $2\xi\omega_n = \frac{1}{g(\mu)}$ and $\omega_n^2 = \frac{K_0 r}{g(\mu)}$. Therefore, the parameters in the aforementioned theorem are determined.

The numerator of the closed-loop transfer function (21) is not 1. Then, let $\phi(s) = \frac{\frac{K_0 r}{g(\mu)}}{s^2 + \frac{1}{g(\mu)} \cdot s + \frac{K_0 r}{g(\mu)}} = \frac{b(s)}{a(s)} = \frac{Y(s)}{U(s)}$, and its inner layer is $Y_1(s) = \frac{1}{s^2 + \frac{1}{g(\mu)} \cdot s + \frac{K_0 r}{g(\mu)}} \cdot U(s)$. The Laplace transform of the differential equation is $y'' + \frac{1}{g(\mu)} \cdot y' + \frac{K_0 r}{g(\mu)} \cdot y = u$. The denominator of the SGD-M transfer function is second-order. We define the state variables

$$\begin{cases} x_1 = y, \\ x_2 = y' = x_1', \end{cases}$$

and the differential equation is transformed into $x_2' = -\frac{1}{g(\mu)} \cdot x_2 - \frac{K_0 r}{g(\mu)} \cdot x_1 + u$. Therefore, the dynamics equations of the inner layer are

$$\begin{cases} \begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{K_0 r}{g(\mu)} & -\frac{1}{g(\mu)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u, \\ y_1 = x_1 = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}. \end{cases}$$

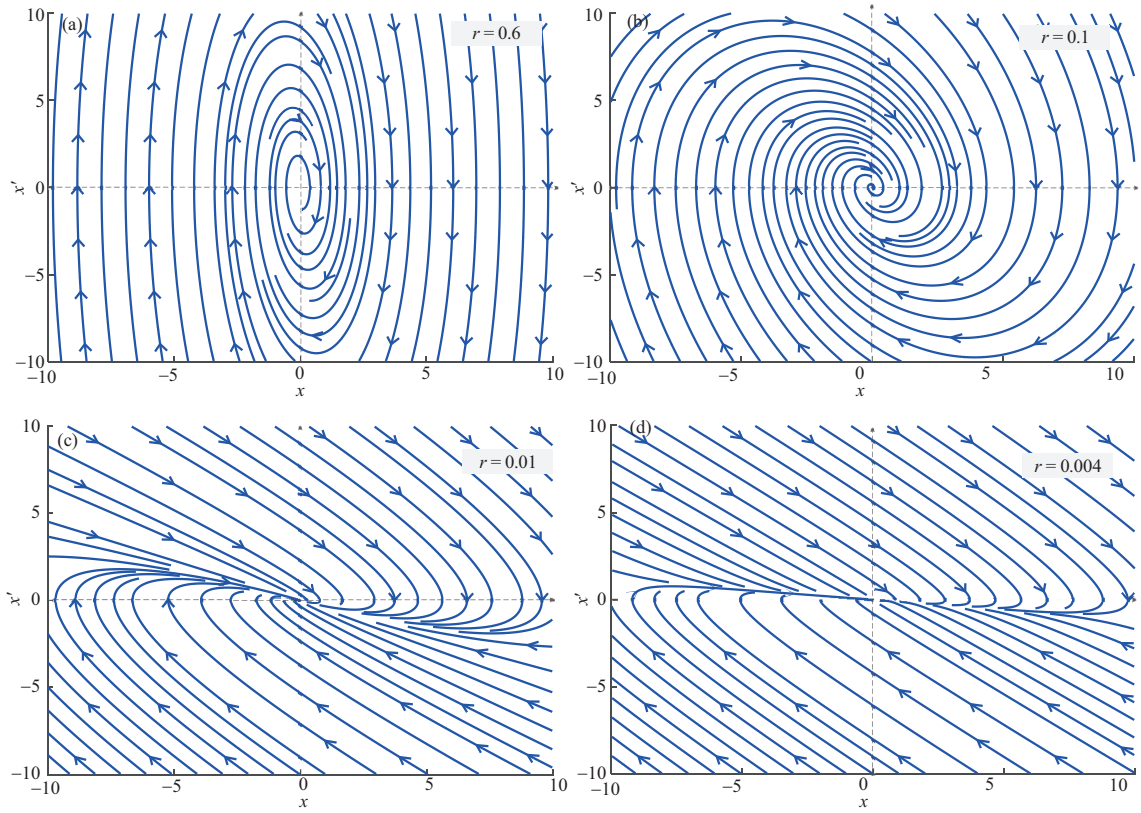

Figure 2 Phase plane of stochastic gradient descent-momentum. (a) $r = 0.6$; (b) $r = 0.1$; (c) $r = 0.01$; (d) $r = 0.004$.

From (8) and (21), we have $Y(s) = \frac{K_0 r}{g(\mu)} \cdot X_1(s)$. Thus, the dynamics equation of the outer layer is

$$y = \frac{K_0 r}{g(\mu)} \cdot x_1 = \frac{K_0 r}{g(\mu)} \cdot \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}.$$

In summary, the dynamics equation of the SGD-M method was obtained. The SGD-M is a second order system, and the differential equation of the system is $y'' + \frac{1}{g(\mu)} \cdot y' + \frac{K_0 r}{g(\mu)} \cdot y = 0$. When $K_0 = 25$ and $g(\mu) = 0.9$, the stability and dynamic performance of the system are analyzed according to the $x - x'$ phase trajectory of the SGD-M, as shown in Figure 2.

In Figure 2, when $0 < \xi = \frac{1}{\sqrt{4K_0 r g(\mu)}} < 1$ and $K_0 \cdot r \cdot g(\mu) > 0.25$, the second-order control system has a pair of conjugate complex roots, and $s_{1.2} = -\xi\omega_n \pm \mathrm{j}\omega_n\sqrt{1-\xi^2} = -\frac{1}{2g(\mu)} \pm \mathrm{j} \cdot \frac{\sqrt{4K_0 r g(\mu) - 1}}{2g(\mu)}$. The system exhibits a damped oscillation process, and the free motion of the system is a periodic attenuation oscillation converging to the equilibrium point. The greater the learning rate is, the longer the oscillation process becomes, as shown in Figures 2(a) and (b). When $\xi = \frac{1}{\sqrt{4K_0 r g(\mu)}} > 1$ and $K_0 \cdot r \cdot g(\mu) \leqslant 0.25$, the second-order control system has two different real roots, and $s_{1.2} = -\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1} = -\frac{1}{2g(\mu)} \pm \frac{\sqrt{1 - 4K_0 r g(\mu)}}{2g(\mu)}$. When $\xi = \frac{1}{\sqrt{4K_0 r g(\mu)}} = 1$ and $K_0 \cdot r \cdot g(\mu) = 0.25$, the second-order control system has a pair of equal real roots, and $s_{1.2} = -\omega_n = -\sqrt{\frac{K_0 r}{g(\mu)}}$. Therefore, when $\xi \geqslant 1$, as the learning rate decreases, the system decays monotonically, the free movement of the system tends to the equilibrium point non-periodically, and the oscillations decrease, as shown in Figures 2(c) and (d). In the learning process, the position of the initial value will also affect the convergence time. The asymptotes in Figures 2(c) and (d) gradually separate, indicating that due to the influence of the learning rate, the speed and time gap required for the system to approach the equilibrium position increase under different initial value selection schemes. In short, the parameters K_0 and $g(\mu)$ are fixed. When r is larger, the oscillations of the system are stronger. When r is smaller, the oscillations of the system are weaker.

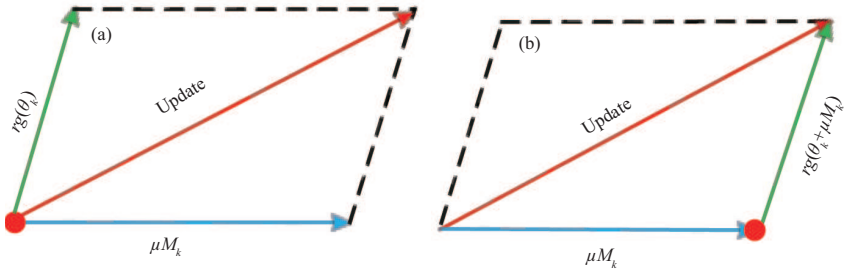

Figure 3 Updated track of SGD-M (a) and NAG (b) [8].

4.3 Stochastic gradient descent with Nesterov acceleration = NAG

The further improvement of the NAG method based on SGD and SGD-M aims to suppress the oscillations of the SGD staying at the local optimal point and accelerating the convergence speed of the SGD-M method. The time-domain expression of the NAG is obtained as follows:

$$\theta_{n+1} = \theta_n - r \cdot (1+\alpha) \cdot \frac{\partial L_n}{\partial \theta_n} - \alpha r \left(\alpha^{n-i} \cdot \sum_{i=0}^{n-1} \frac{\partial L_i}{\partial \theta_i} \right). \tag{22}$$

SGD-M calculates the gradient and then adds momentum. NAG adds the momentum term before calculating the gradient. Figure 3 shows the difference between NAG and SGD-M: NAG updates the momentum by using the future gradient. Therefore, the G_n of the NAG method is set to the advanced correction. Then, $G_n = \alpha \cdot \frac{Ts+1}{\alpha T s+1}$ and $G_{\mathrm{NAG}}(s) = G_{\mathrm{SGD}}(s) \cdot G_n = K_0 \cdot \frac{r}{s} \cdot \alpha \cdot \frac{Ts+1}{\alpha T s+1}$. With $g(\mu) = \alpha T$ and $K_1 = K_0 \alpha$, the open-loop transfer function of the NAG controller is as follows:

$$G_{\mathrm{NAG}}(s) = K_0 \cdot \frac{r}{s} \cdot \frac{1}{g(\mu)s+1} \cdot \left(\frac{g(\mu)}{\alpha} s + 1 \right) = \frac{K_0 r g(\mu) s + K_0 \alpha r}{g(\mu) s^2 + s}. \tag{23}$$

The NAG controller is a typical second-order control system with a zero point, and its closed-loop system is an accelerating gradient method expressed by

$$\phi(s) = \frac{G_{\mathrm{NAG}}(s)}{1+G_{\mathrm{NAG}}(s)} = \frac{K_0 r g(\mu) \cdot s + K_0 \alpha r}{g(\mu) \cdot s^2 + s + K_0 r g(\mu) \cdot s + K_0 \alpha r} = \frac{K_0 r \cdot s + \frac{K_0 \alpha r}{g(\mu)}}{s^2 + (K_0 r + \frac{1}{g(\mu)}) \cdot s + \frac{K_0 \alpha r}{g(\mu)}}. \tag{24}$$

Theorem 4. The dynamic process of the NAG controller is controlled by $\omega_n = \sqrt{\frac{K_0 \alpha r}{g(\mu)}}$ and $\xi = \frac{K_0 \cdot r \cdot g(\mu)+1}{2\sqrt{K_0 \cdot \alpha \cdot r \cdot g(\mu)}}$, with the dynamics equations

$$\begin{cases} \begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{K_0 \alpha r}{g(\mu)} & -\left(K_0 r + \frac{1}{g(\mu)}\right) \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u, \\ y = \begin{pmatrix} \frac{K_0 \alpha r}{g(\mu)} & K_0 r \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \end{cases}$$

where u is the two-dimensional input vector.

Proof. The characteristic equation of the NAG controller is $s^2 + (K_0 r + \frac{1}{g(\mu)}) \cdot s + \frac{K_0 \alpha r}{g(\mu)} = 0$. The second-order response characteristics are controlled by K_0, r, and $g(\mu)$, and the control coefficient α. Based on the characteristic equation of the second-order control system and the method of undetermined coefficients, we obtain $2\xi\omega_n = K_0 r + \frac{1}{g(\mu)}$ and $\omega_n^2 = \frac{K_0 \alpha r}{g(\mu)}$. We then obtain the parametric solutions $\omega_n = \sqrt{\frac{K_0 \alpha r}{g(\mu)}}$ and $\xi = \frac{K_0 \cdot r \cdot g(\mu)+1}{2\sqrt{K_0 \cdot \alpha \cdot r \cdot g(\mu)}}$.

The numerator of the closed-loop transfer function (24) is not 1. Then, let $\phi(s) = \frac{K_0 r \cdot s + \frac{K_0 \alpha r}{g(\mu)}}{s^2 + (K_0 r + \frac{1}{g(\mu)}) \cdot s + \frac{K_0 \alpha r}{g(\mu)}} = \frac{b(s)}{a(s)} = \frac{Y(s)}{U(s)}$, and its inner layer is $Y_1(s) = \frac{1}{s^2 + (K_0 r + \frac{1}{g(\mu)}) \cdot s + \frac{K_0 \alpha r}{g(\mu)}}$. The corresponding differential equation

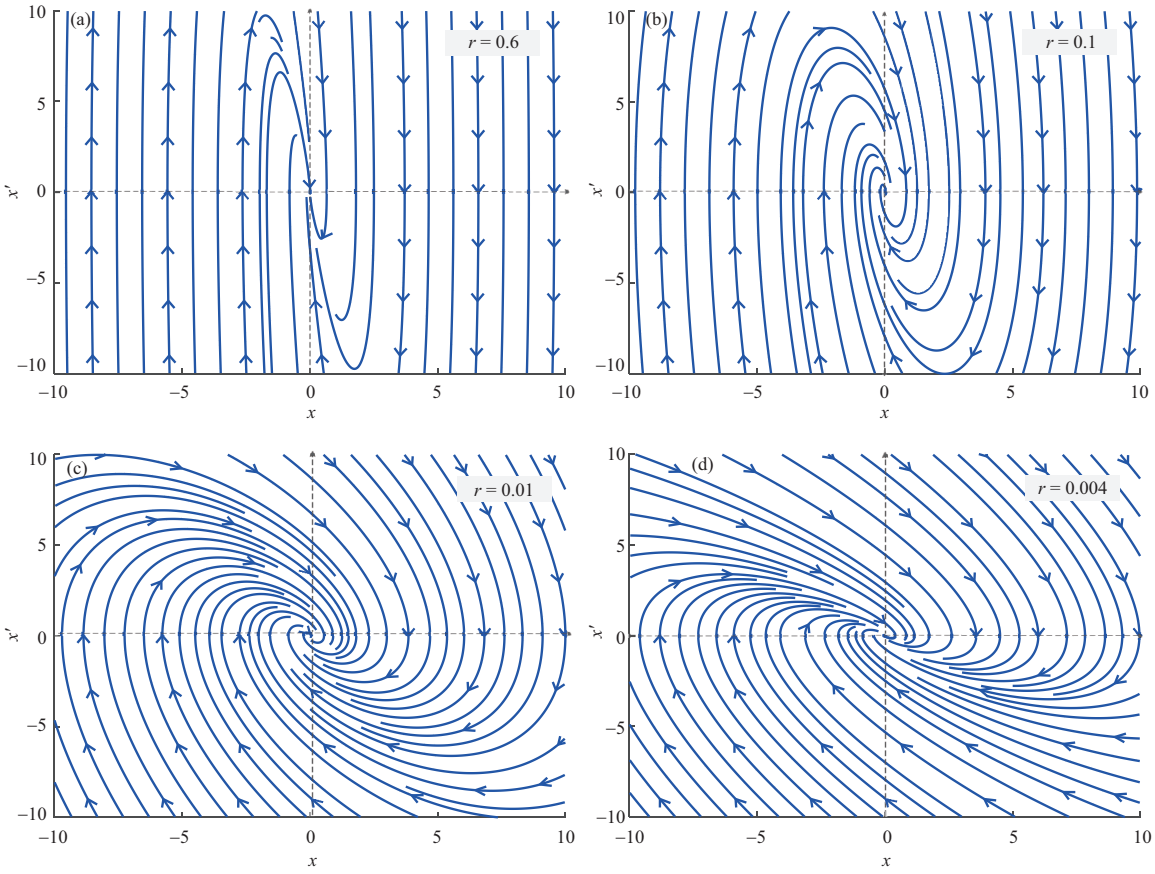

Figure 4 Phase plane of Nesterov accelerated gradient. (a) $r = 0.6$; (b) $r = 0.1$; (c) $r = 0.01$; (d) $r = 0.004$.

obtained by the inverse Laplace transform is $y'' + (K_0 r + \frac{1}{g(\mu)}) \cdot y' + \frac{K_0 \alpha r}{g(\mu)} \cdot y = u$. The denominator of the NAG transfer function is the second order. We define the state variables

$$\begin{cases} x_1 = y, \\ x_2 = y' = x_1', \end{cases}$$

and the differential equation is transformed into $x_2' = -(K_0 r + \frac{1}{g(\mu)}) \cdot x_2 - \frac{K_0 \alpha r}{g(\mu)} \cdot x_1 + u$. Thus, the dynamics equations of the inner layer are as follows:

$$\begin{cases} \begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{K_0 \alpha r}{g(\mu)} & -\left(K_0 r + \frac{1}{g(\mu)}\right) \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u, \\ y_1 = x_1. \end{cases} \tag{25}$$

For the outer layer, $Y(s) = b(s) \cdot X_1(s)$ and $\phi(s) = \frac{K_0 r}{g(\mu) s^2 + s + K_0 r}$. Then, we have $Y(s) = K_0 r \cdot s \cdot X_1(s) + \frac{K_0 \alpha r}{g(\mu)} \cdot X_1(s)$. Therefore, the differential equation is $y = K_0 r \cdot x_1' + \frac{K_0 \alpha r}{g(\mu)} \cdot x_1$, so it is reduced to $y = K_0 r \cdot x_2 + \frac{K_0 \alpha r}{g(\mu)} \cdot x_1$.

In summary, the dynamics equation of the NAG method was obtained. NAG is a second-order system, and the differential equation of the system is $y'' + (K_0 r + \frac{1}{g(\mu)}) \cdot y' + \frac{K_0 \alpha r}{g(\mu)} \cdot y = 0$. When $K_0 = 25$, $\alpha = 6$, and $g(\mu) = 0.9$, the stability and dynamic performance of the system are analyzed based on the $x - x'$ phase trajectories of the NAG controller, as shown in Figure 4.

In Figure 4, when $\xi = \frac{K_0 \cdot \alpha \cdot r \cdot g(\mu) + 1}{2\sqrt{K_0 \cdot \alpha \cdot r \cdot g(\mu)}} > 1$, the second-order control system has two different real roots, and $s_{1,2} = -\xi \omega_n \pm \omega_n \cdot \sqrt{\xi^2 - 1} = -\frac{K_0 \cdot \alpha \cdot r \cdot g(\mu) + 1}{2g(\mu)} \pm \frac{K_0 \cdot \alpha \cdot r \cdot g(\mu) - 1}{2g(\mu)}$. We have $s_1 = -\frac{1}{g(\mu)}$ and $s_2 = -K_0 \cdot \alpha \cdot r$, and the system decays monotonically. Figures 4(a)–(d) show cyclical tracking from the initial state to the equilibrium state. When r is large, the oscillations of the system increase, as shown in Figure 4(a).

When r is small, the oscillations are expected to be weakened and finally tend to the equilibrium state, as shown in Figure 4(d). After obtaining a suitable r, the system is balanced in terms of stability and rapidity, and the effects shown in Figures 4(b) and (c) can be achieved. Moreover, the oscillations in Figure 4(b) are stronger than those in Figure 4(c), and the stability of the system is stronger.

In summary, the steps in the derivation process of the control equation of optimizers are as follows: the principle of optimizer → time-domain expression → transfer function → control equation. Based on the SGD method, SGD-M introduces first order momentum, which is equivalent to adding an inertial link, namely, a momentum factor $G_m = \frac{1}{Ts+1}$. NAG adds a momentum term on the basis of SGD-M, namely, the advanced correction factor $G_n = \alpha \cdot \frac{Ts+1}{\alpha Ts+1}$. SGD is a first order system. The overshoot is weak when the noise is low. The model is approximately a first order monotonically ascending process, and the convergence speed of the model is fast. The overshoot in the SGD-M and NAG second-order systems slows the convergence speed of training. The phase plane analysis of SGD-M and NAG shows that when the parameters K_0, α, and $g(\mu)$ are fixed, the larger r is, the stronger the system oscillations become; the smaller r is, the weaker the system oscillations become.

5 Experiments

The experiment aims to explore the effects of different optimizer methods (SGD, SGD-M, and NAG) and different hyperparameters on the classification and recognition of the MNIST and CIFAR-10 datasets, so as to analyze their effects on the performance of the DL model and verify the proposed theory of the control model of the optimizers.

5.1 Experimental data and experimental settings

The MNIST dataset [44] consists of 60000 training samples (55000 training samples and 5000 validation samples) and 10000 test samples. Each sample is a 28×28 pixel gray-scale handwritten digital image with numbers ranging from 0 to 9. The hyperparameters set in the experiment are as follows: the learning rates are 0.05 and 0.1, the momentum factor is 0.9, and the weight decay is 0.01.

The CIFAR-10 dataset [45] consists of the training samples of 50000 color images (45000 training samples and 5000 validation samples) and 10000 samples. Each sample is a 32×32 pixel color image with 10 categories (airplane, car, bird, cat, deer, dog, frog, horse, boat, and truck). The hyperparameters set in the experiment are as follows: the learning rates are 0.005 and 0.001, the momentum factor is 0.5, and the weight decay is 0.01.

In the basic verification process of the target recognition and classification, the theory of the DL generally uses two kinds of basic datasets: a gray image dataset (represented by MNIST) and a color image dataset (represented by CIFAR-10). The reason for using MNIST dataset is that it is a concave-convex dataset, the classification results are relatively stable, and the classification effect is good. Therefore, it is more conducive to focusing on the influence of the optimizer. Compared with the MNIST dataset, the concavity and convexity of the CIFAR-10 dataset are relatively complex. The network model selected for the former is a three-layer network configuration (linear transformation of input nodes, implicit computation with bias, and output classification mapping), and the latter is a 10-layer deep network configuration (six convolution layers, a maximum pooling layer, a mean two-dimensional pooling layer, a mean (8×8)-dimensional pooling layer, and an output layer). All the algorithms are implemented in Python 3.6 on a personal computer with 2.5-GHz IntelCore i7 with 8 GB of RAM.

5.2 Experimental results and analysis

The evaluation indices of the experimental results are analyzed in terms of the performance, convergence speed, accuracy, loss, and time consumption of the optimizer. The experimental results are consistent with the proposed theory. The classification results on the MNIST and CIFAR-10 datasets with different optimizers and different learning rates are shown in Table 1.

As shown in Table 1, for the classification and recognition of the MNIST and CIFAR-10 datasets, the relationship between the accuracies of the training, verification, and test sets of the optimizers is SGD-M $<$ NAG $<$ SGD. The relationship between the losses of training, verification and test sets of the optimizers is SGD $<$ NAG $<$ SGD-M. Therefore, compared with SGD-M and NAG, SGD has the characteristics of a low loss, low time consumption, and high precision, and its recognition performance is the best. This

Table 1 Classification and recognition results of MNIST and CIFAR-10 datasets

	MNIST						CIFAR-10					
	Epoch (50), $r = 0.05$			Epoch (50), $r = 0.1$			Epoch (40), $r = 0.001$			Epoch (40), $r = 0.005$		
	SGD	SGD-M	NAG	SGD	SGD-M	NAG	SGD	SGD-M	NAG	SGD	SGD-M	NAG
Training accuracy (%)	99.96	93.19	93.46	99.95	92.19	92.79	99.28	89.02	90.21	97.80	87.98	88.95
Training loss	0.007	0.275	0.27	0.007	0.3	0.285	0.041	0.325	0.296	0.081	0.352	0.331
Validation accuracy (%)	98.50	95.62	95.76	98.40	95.16	95.46	84.16	82.50	83.72	83.34	82.94	83.08
Validation loss	0.058	0.21	0.211	0.06	0.302	0.214	0.512	0.516	0.501	0.532	0.509	0.519
Test accuracy (%)	98.30	94.15	94.24	98.28	93.74	94.04	82.72	81.33	82.51	82.56	81.36	81.85
Test loss	0.056	0.243	0.245	0.057	0.336	0.25	0.505	0.544	0.514	0.551	0.539	0.532
Time	406.87 s	494.51 s	491.93 s	426.04 s	505.71 s	495.41 s	2.93 h	3.48 h	3.21 h	3.07 h	3.65 h	3.34 h

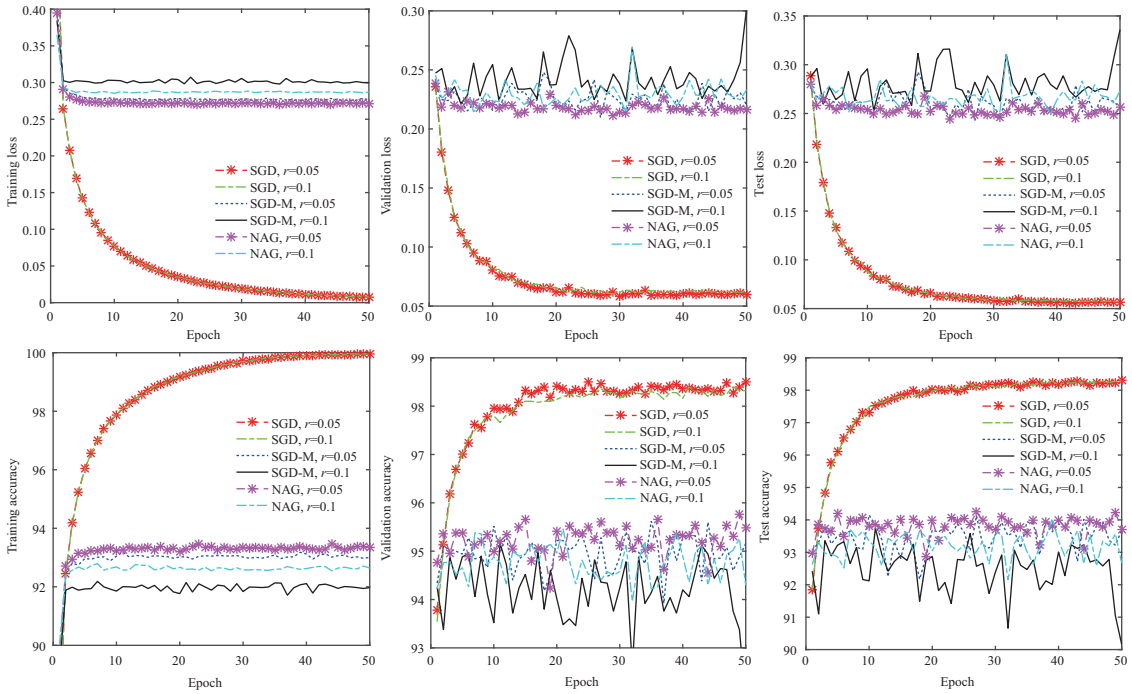

Figure 5 Effects of different optimizers on MNIST classification.

is consistent with the theoretical analysis in Section 4; namely, SGD is a first-order system with weak overshoot, and the convergence rate of the model is fast. SGD-M and NAG are second-order systems with overshoot, and the existing overshoot slows the convergence rate of training. The visualization of the training processes for the classification and recognition of the MNIST and CIFAR-10 datasets by different methods and the loss and accuracy changes of the training, validation, and test sets are shown in Figures 5 and 6.

Figure 5 shows that the accuracies for the training, verification, and test sets of the different optimizers at $r = 0.1$ are less than $r = 0.05$. The losses and training times of the optimizers at $r = 0.1$ are higher than those at $r = 0.05$. Figure 6 shows that the training, verification, and test set accuracies of the different optimizers at $r = 0.005$ are less than those at $r = 0.001$. The losses and training times of the different optimizers at $r = 0.005$ are higher than those $r = 0.001$. The experimental results show that the size of r has an impact on the training of the optimizer. The smaller r is, the weaker the system oscillations are, which is conducive to accelerating the convergence. This is consistent with the theoretical analysis in the previous section; namely, the larger r is, the stronger the system oscillations are; the weaker the system oscillations are.

According to Table 1, Figures 5 and 6, compared with SGD-M and NAG, SGD has the lowest loss, shortest training time, highest recognition accuracy, and fastest convergence. This shows that SGD is superior to SGD-M and NAG in terms of recognition performance. In addition, the appropriate optimizer determines the performance indices (stability, rapidity, and accuracy) of the control system.

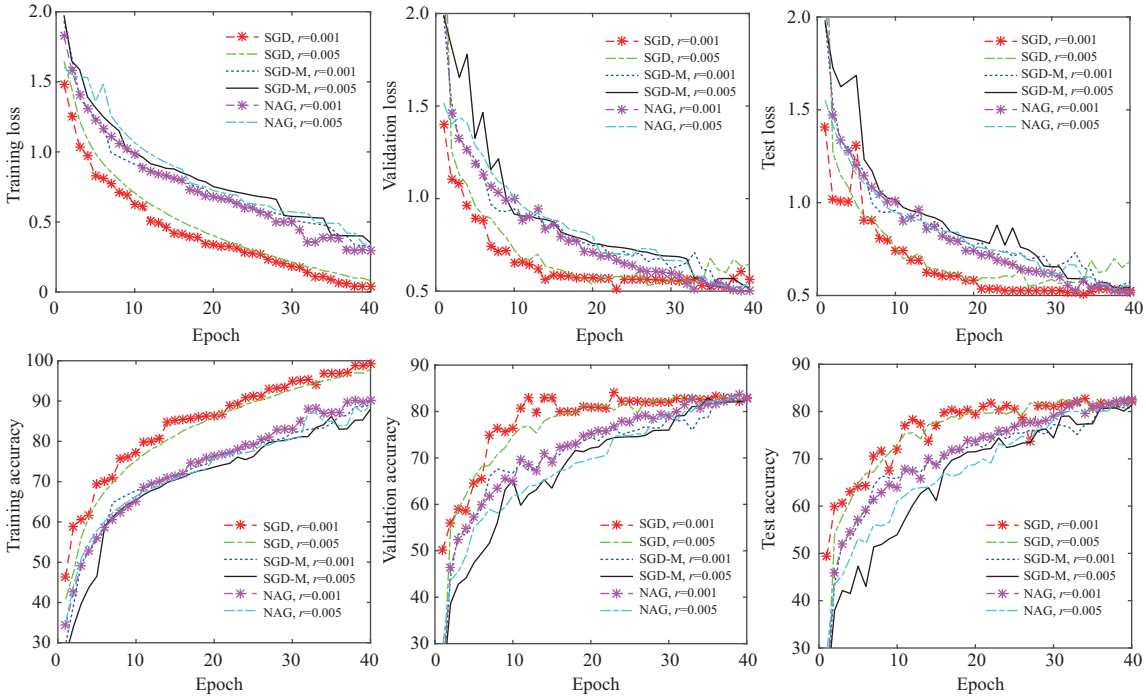

Figure 6 Effects of different optimizers on CIFAR-10 classification.

6 Conclusion

This study systematically provided a control perspective for gradient optimizers, with control models built from theoretical derivations and verified by numerical experiments. The differential equations of the SGD, SGD-M, and NAG optimizers were derived according to the iterative principles and mapped into transfer functions. SGD, SGD-M, and NAG were equivalent to a first-order integral model, a second-order damping model, and a second-order damping model with zero poles, respectively. To reveal the underlying characteristic of the aforementioned optimizers, the time domain method and the phase plane method were proposed to analyze how the dynamic learning processes of these optimizers were affected by various hyperparameters. SGD was a first-order system without overshoot. SGD-M and NAG were second-order systems, and in their learning processes, the overshoot increased with increasing learning rate when the parameters of K_0, α, and $g(\mu)$ were fixed. The experimental results on the MNIST and CIFAR-10 datasets were consistent with the theoretical analysis.

The control model of optimizers provides a new perspective for quantitatively describing the parameter function of optimizers with fixed coefficients, however, to describe optimizers with variable coefficients is not easy. For the optimizers (SGD, SGD-M, and NAG) with fixed coefficients, the process of constructing constant coefficient differential equation and establishing transfer function control model is clear. For the optimizer with time-varying coefficients, the learning rate is decaying, such as RMSProp and Adam, to build a control model to analyze their properties requires extra considerations from time-varying system. Therefore, we will continue our research to build time-varying control models for the adaptive optimizers. This study is meaningful for the control theories for the optimizers aforementioned.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant No. 61972160) and Natural Science Foundation of Guangdong Province (Grant No. 2021A1515012301).

References

1 Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks. Commun ACM, 2017, 60: 84–90
2 Sun Y L, Yu B G. Python Machine Learning Algorithm and Practice (in Chinese). Beijing: Publishing House of Electronics Industry, 2021
3 Robbins H, Monro S. A stochastic approximation method. Ann Math Statist, 1951, 22: 400–407
4 Polyak B T. Some methods of speeding up the convergence of iteration methods. USSR Comput Math Math Phys, 1964, 4: 1–17
5 Nesterov Y E. A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$. Doklady ANSSSR, 1983, 27: 543–547

6 An W P, Wang H Q, Sun Q Y, et al. A PID controller approach for stochastic optimization of deep networks. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2018. 8522–8531
7 Wang H Q, Luo Y, An W P, et al. PID controller-based stochastic optimization acceleration for deep neural networks. IEEE Trans Neural Netw Learn Syst, 2020, 31: 5079–5091
8 Wu W, Jing X Y, Du W C, et al. Learning dynamics of gradient descent optimization in deep neural networks. Sci China Inf Sci, 2021, 64: 150102
9 Blum J R. Multidimensional stochastic approximation methods. Ann Math Statist, 1954, 25: 737–744
10 Wang Z Y, Fu Y, Huang S T. Deep Learning Through Sparse and Low-Rank Modeling (in Chinese). Beijing: China Machine Press, 2021
11 Lei Y W, Hu T, Li G Y, et al. Stochastic gradient descent for nonconvex learning without bounded gradient assumptions. IEEE Trans Neural Netw Learn Syst, 2020, 31: 4394–4400
12 Engel I, Bershad N J. A transient learning comparison of Rosenblatt, backpropagation, and LMS algorithms for a single-layer perceptron for system identification. IEEE Trans Signal Process, 1994, 42: 1247–1251
13 Yang H H, Amari S. Complexity issues in natural gradient descent method for training multilayer perceptrons. Neural Computation, 1998, 10: 2137–2157
14 Bengio Y. Learning deep architectures for AI. FNT Machine Learn, 2009, 2: 1–127
15 Li Y Z, Liang Y Y. Learning overparameterized neural networks via stochastic gradient descent on structured data. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, Paris, 2018. 8168–8177
16 Luo X, Qin W, Dong A, et al. Efficient and high-quality recommendations via momentum-incorporated parallel stochastic gradient descent-based learning. IEEE CAA J Autom Sin, 2021, 8: 402–411
17 Ruder S. An overview of gradient descent optimization algorithms. 2017. ArXiv:1609.04747
18 Ding X H, Ding G G, Zhou X X, et al. Global sparse momentum SGD for pruning very deep neural networks. In: Proceedings of the 33rd Conference on Neural Information Processing Systems, Vancouver, 2019. 1–13
19 Wang J Y, Tantia V, Ballas N, et al. SlowMo: improving communication-efficient distributed SGD with slow momentum. In: Proceedings of International Conference on Learning Representation, 2020. 1–27
20 Jarek D. SGD momentum optimizer with step estimation by online parabola model. 2019. ArXiv:1907.07063
21 Aleksandar B, Guy L, David B. Nesterov's accelerated gradient and momentum as approximations to regularised update descent. In: Proceedings of International Joint Conference on Neural Networks, Anchorage, 2017. 1899–1903
22 Luo L C, Xiong Y H, Liu Y, et al. Adaptive gradient methods with dynamic bound of learning rate. In: Proceedings of International Conference on Learning Representation, New Orleans, 2019. 1–19
23 Zeyuan A Z. Katyusha: the first direct acceleration of stochastic gradient methods. J Mach Learning Res, 2017, 18: 8194–8244
24 Luo X, Zhou M C. Effects of extended stochastic gradient descent algorithms on improving latent factor-based recommender systems. IEEE Robot Autom Lett, 2019, 4: 618–624
25 Luo X, Wang D X, Zhou M C, et al. Latent factor-based recommenders relying on extended stochastic gradient descent algorithms. IEEE Trans Syst Man Cyber Syst, 2021, 51: 916–926
26 Lei Y W, Tang K. Learning rates for stochastic gradient descent with nonconvex objectives. IEEE Trans Pattern Anal Mach Intell, 2021, 43: 4505–4511
27 Amid E, Anil R, Fifty C, et al. Step-size adaptation using exponentiated gradient updates. 2022. ArXiv:2202.00145
28 Andrychowicz M, Denil M, Colmenarejo G S, et al. Learning to learn by gradient descent by gradient descent. In: Proceedings of Advances in Neural Information Processing Systems, 2016. 1–9
29 Xiong Y H, Lan L C, Chen X N, et al. Learning to schedule learning rate with graph neural networks. In: Proceedings of International Conference on Learning Representations, 2022. 1–21
30 Shu J, Zhu Y W, Zhao Q, et al. MLR-SNet: transferable LR schedules for heterogeneous tasks. In: Proceedings of International Conference on Learning Representations, 2021. 1–25
31 Alamo T, Ferramosca A, González A H, et al. A gradient-based strategy for integrating real time optimizer (RTO) with model predictive control (MPC). IFAC Proc Volumes, 2012, 45: 33–38
32 Chen J N, Hua C C. Adaptive full-state-constrained control of nonlinear systems with deferred constraints based on nonbarrier Lyapunov function method. IEEE Trans Cybern, 2022, 52: 7634–7642
33 Lee T H, Trinh H M, Park J H. Stability analysis of neural networks with time-varying delay by constructing novel Lyapunov functionals. IEEE Trans Neural Netw Learn Syst, 2018, 29: 4238–4247
34 Faydasicok O. A new Lyapunov functional for stability analysis of neutral-type Hopfield neural networks with multiple delays. Neural Networks, 2020, 129: 288–297
35 Yuan F Y, Liu Y J, Liu L, et al. Adaptive neural consensus tracking control for nonlinear multiagent systems using integral barrier Lyapunov functionals. IEEE Trans Neural Netw Learn Syst, 2021. doi: 10.1109/TNNLS.2021.3112763
36 Wang Z L, Wang S K, Chen G S, et al. MATLAB/Simulink and control system simulation (in Chinese). Beijing: Publishing House of Electronics Industry, 2019
37 Bhaya A, Kaszkurewicz E. Control Perspectives on Numerical Algorithms and Matrix Problems. Philadelphia: Society for Industrial and Applied Mathematics, 2006
38 Chen Y Q, Wei Y H, Wang Y, et al. On the unified design of accelerated gradient descent. In: Proceedings of 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 2019
39 Liu L, Liu J, Hsieh C J, et al. Stochastically controlled compositional gradient for composition problems. IEEE Trans Neural Netw Learn Syst, 2021, doi: 10.1109/TNNLS.2021.3098222
40 Xu B G. Principle of Automatic Control (in Chinese). Beijing: Publishing House of Electronics Industry, 2013
41 Oppenheim A V, Willsky A S, Nawab S H. Signals and Systems (in Chinese). 2nd ed. Beijing: Publishing House of Electronics Industry, 2020
42 Department of Mathematics, Tongji University. Higher Mathematics (in Chinese). Beijing: Posts & Telecom Press, 2016
43 Gao G S, Yu W X. Principle of Automatic Control (in Chinese). Guangzhou: South China University of Technology Press, 2013
44 Lecun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition. Proc IEEE, 1998, 86: 2278–2324
45 Krizhevsky A, Hinton G. Learning Multiple Layers of Features From Tiny Images. Technical Report, University of Toronto, 2009