2.3. DMAdam algorithm

Algorithm 1 DMAdam

Require: $\eta_k > 0$, $\alpha_k > 0$, $\beta_k \in (0,1)$, $\epsilon > 0$,
Ensure: $\boldsymbol{x}^1 \in \mathbb{R}^d$, $\boldsymbol{m}^0 = 0$, $\boldsymbol{v}^0 = 0$
1: **for** $k = 1$ to K **do**
2: $\boldsymbol{g}^k = \nabla \ell\left(\boldsymbol{x}^k; \xi_k\right)$
3: $\boldsymbol{m}^k = \frac{\boldsymbol{m}^{k-1} + \lambda_k \boldsymbol{g}^k}{\sqrt{k+1}}$, $\lambda_k = \alpha_k \sqrt{k+1}$
4: $\boldsymbol{v}^k = \beta_k \boldsymbol{v}^{k-1} + \left(1-\beta_k\right)\left(\left(\boldsymbol{g}^k\right)^2 + \epsilon_0\right)$, $\epsilon_0 = \epsilon / \left(1-\beta_k\right)$
5: $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \eta_k \frac{\boldsymbol{m}^k}{\sqrt{\boldsymbol{v}^k + \epsilon}}$
6: **end for**

Dual averaging (DA) methods and adaptive gradient methods play different roles in optimization. The DA methods stabilize updates by accumulating gradient information, while adaptive gradient methods allow flexibility by equipping the learning rate for each component. On this basis, we propose the DMAdam algorithm (see Algorithm 1) and present a flowchart (Fig. 1) to highlight the major differences. Firstly, $\boldsymbol{m}^k$ is updated using scaled gradients $\boldsymbol{g}^k$ with a factor α_k, which enhances the precision of the updates. Secondly, to enhance numerical stability throughout the iteration process, the DMAdam algorithm adds ϵ_0 to $\boldsymbol{v}^k$. Finally, as shown in Eq. (5), the parameter η_k fine-tunes the linear interpolation between $\boldsymbol{x}^k$ and $\boldsymbol{z}^{k+1}$ to stabilize the update process and reduce oscillations.

3. Convergence trajectories of bench-mark functions

This section presents a comparison of the convergence trajectories of both non-momentum and momentum-based methods on six benchmark functions.[1] Notably, the term $\boldsymbol{m}^k$ in DMAdam is comparable to the momentum term in SGDM and Adam.

3.1. Accelerating convergence

The gradient accumulation term $\boldsymbol{m}^k$ improves the convergence rate by balancing historical gradients [29]. We validated this result using standard benchmark functions, including Sphere, Booth, and Matyas.

- Sphere function:

$$f\left(x_1, x_2\right) = x_1^2 + x_2^2, \tag{8}$$

where the minimum point is $(0,0)$. The Sphere function is convex and smooth, and we compare the optimization trajectories of momentum-based and non-momentum algorithms in Fig. 2. Compared to traditional methods such as SGD and AdaGrad, momentum-based algorithms like Adam typically achieve convergence more effectively. Among these algorithms, DMAdam exhibits faster convergence to the global minimum.

- Booth function:

$$f\left(x_1, x_2\right) = \left(x_1 + 2x_2 - 7\right)^2 + \left(2x_1 + x_2 - 5\right)^2, \tag{9}$$

where the minimum point is $(1,3)$. The Booth function is a continuous and differentiable function, and we compare the optimization trajectories of momentum-based and non-momentum algorithms in Fig. 3. Similar to the optimization trajectory of the Sphere function, momentum-based SGDM and Adam algorithms achieve faster convergence rates compared to traditional SGD and AdaGrad algorithms. Moreover, the DMAdam algorithm still achieves fast convergence to the minimum point.

[1] https://www.sfu.ca/~ssurjano/optimization.html

- Matyas function:

$$f(x_1, x_2) = 0.26(x_1^2 + x_2^2) - 0.48 x_1 x_2, \tag{10}$$

where the minimum point is $(0,0)$. The Matyas function is a symmetric and smooth function. As shown in Fig. 4, the Adam algorithm converges to the minimum faster than the non-momentum algorithms, while DMAdam further improves the convergence rate, exhibiting additional speedups compared to the other algorithms evaluated in this study.

3.2. Skipping local minima

The momentum term $\boldsymbol{m}^k$ uses historical gradient information, prevents updates from stopping at local minima, and improves the ability of the algorithm to search for the minimum [30]. We validated this result using standard benchmark functions, including Beale, Goldstein-Price, and Bukin.

- **Beale Function:**

$$\begin{aligned} f\left(x_1, x_2\right) = &\left(1.5 - x_1 + x_1 x_2\right)^2 + \left(2.25 - x_1 + x_1 x_2^2\right)^2 \\ &+ \left(2.625 - x_1 + x_1 x_2^3\right)^2, \end{aligned} \tag{11}$$

where the minimum point is $(3, 0.5)$. The Beale function is a nonconvex function with several local minima. Fig. 5 shows that momentum-based optimization methods, such as Adam, effectively avoid local minima. Among these methods, DMAdam exhibits superior performance, converging to the global minimum in just a few iterations.

- **Goldstein-Price Function:**

$$\begin{aligned} f\left(x_1, x_2\right) = &\left(1 + \left(x_1 + x_2 + 1\right)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2\right)\right) \\ &\cdot \left(30 + \left(2x_1 - 3x_2\right)^2 \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2\right)\right), \end{aligned} \tag{12}$$

where the minimum point is $(0,-1)$. The Goldstein-Price function is also a non-convex test function with multiple local minima. As shown in Fig. 6, only DMAdam reaches the global minimum point relying on the unique $\boldsymbol{m}^k$.

- **Bukin Function:**

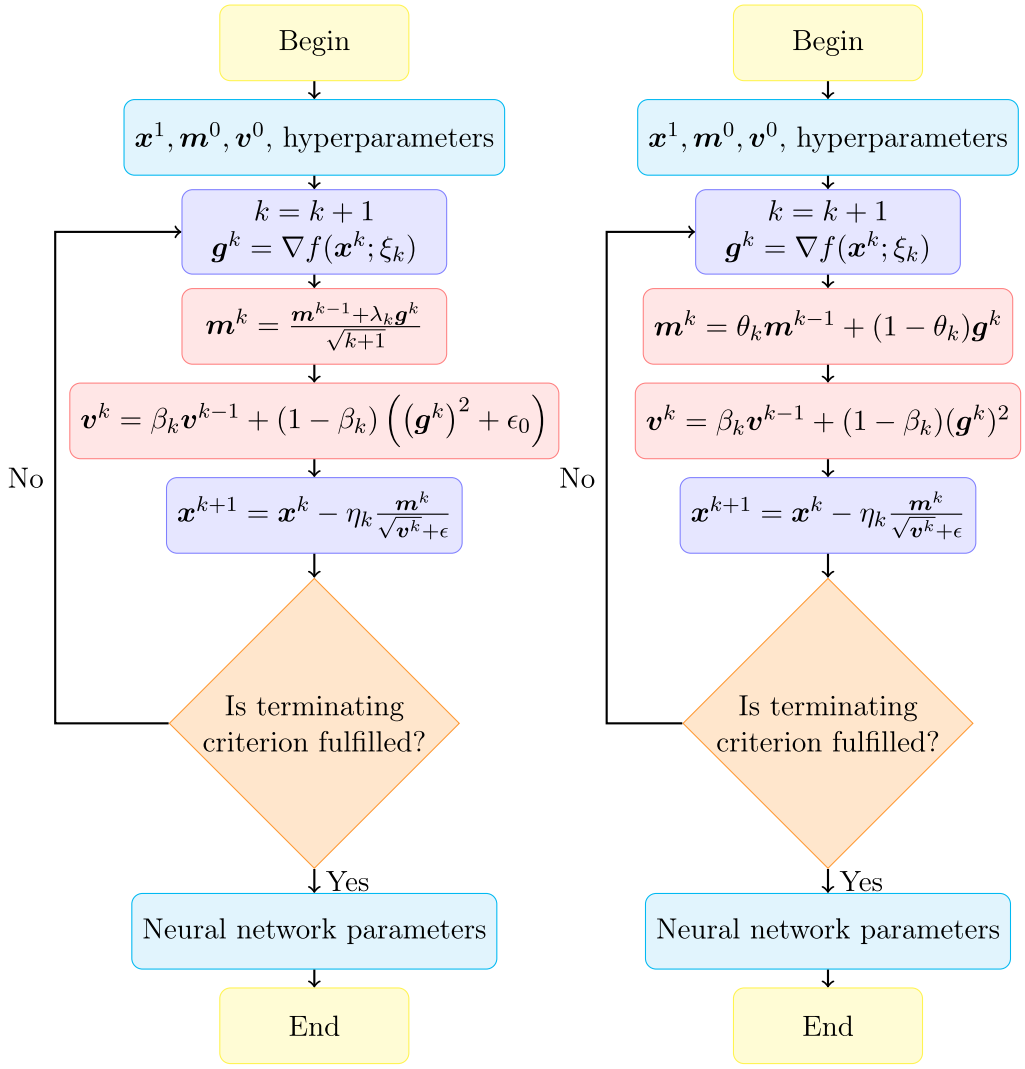
$$f\left(x_1, x_2\right) = 100\sqrt{\left|x_2 - 0.01x_1^2\right|} + 0.01\left|x_1 + 10\right|, \tag{13}$$

where the minimum point is $(-10, 1)$. The Bukin function is a nonconvex test function that exhibits significant sensitivity to changes in variables. As shown in Fig. 7, DMAdam demonstrates faster convergence compared to other methods, particularly performing well in avoiding local minima.

The experimental results on the Sphere, Booth, and Matyas test functions indicate that introducing momentum can effectively improve the convergence rate of optimization algorithms. On the other hand, results from experiments on the Beale, Goldstein-Price, and Bukin test functions highlight the significant role of momentum in avoiding local minima. DMAdam consistently demonstrates superior performance, achieving faster and more stable convergence.

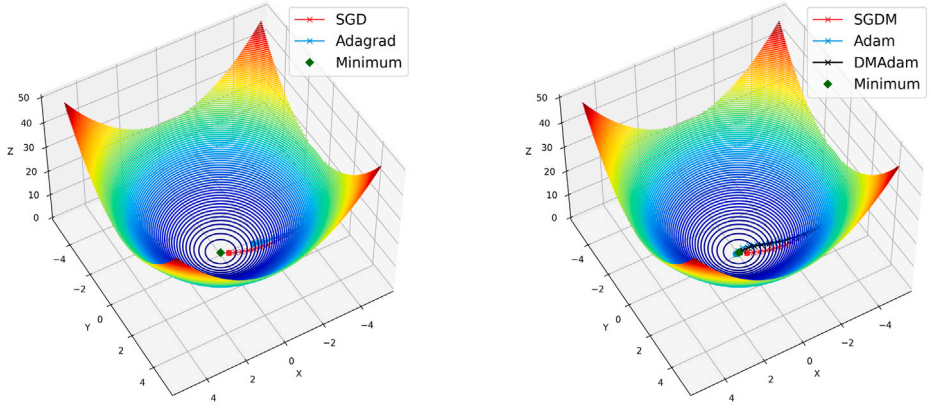
4. Convergence analysis of DMAdam

In Section 4, we first formalize the assumptions required for our convergence analysis. Next, Theorem 1 establishes the minimum convergence of Algorithm 1, followed by Theorem 2, which demonstrates its uniform convergence. Finally, Theorem 3 presents the non-ergodic convergence of the gradient sequences generated by Algorithm 1.



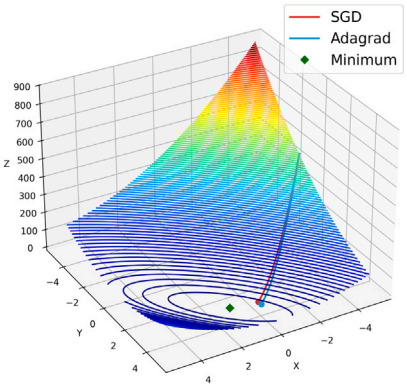
(a) DMAdam **(b)** Adam

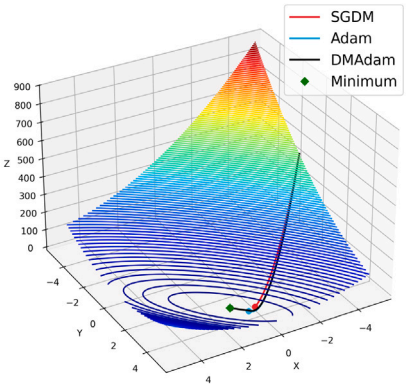
Fig. 1. Flowcharts of DMAdam and Adam.



(a) Non-Momentum Algorithms (b) Momentum-Based Algorithms

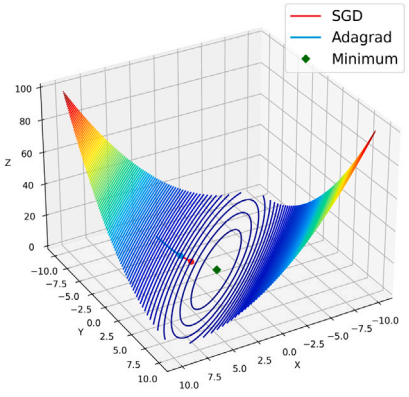
Fig. 2. Convergence trajectories for the Sphere function.

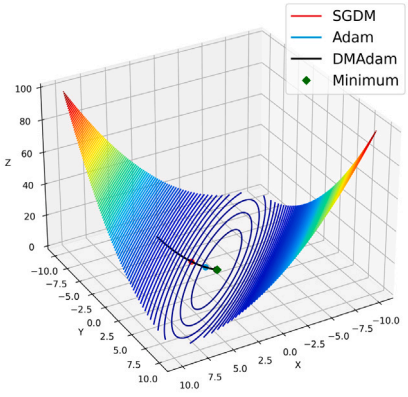




(a) Non-Momentum Algorithms (b) Momentum-Based Algorithms

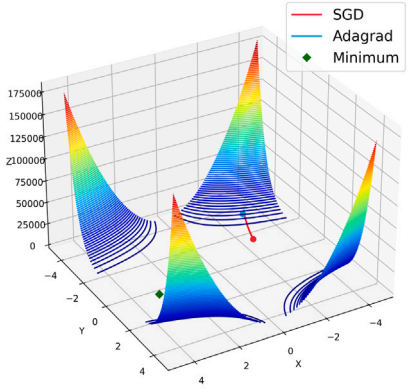
Fig. 3. Convergence trajectories for the Booth function.

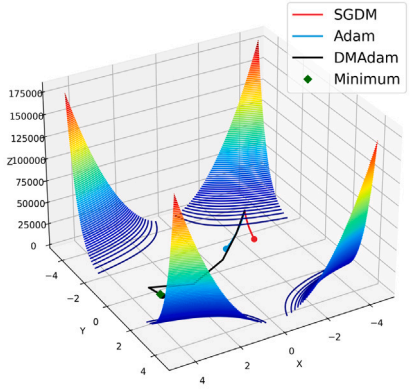




(a) Non-Momentum Algorithms (b) Momentum-Based Algorithms

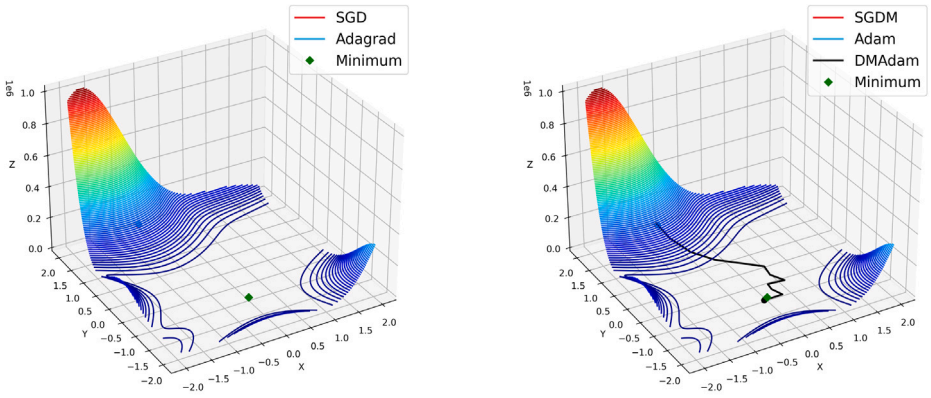
Fig. 4. Convergence trajectories for the Matyas function.





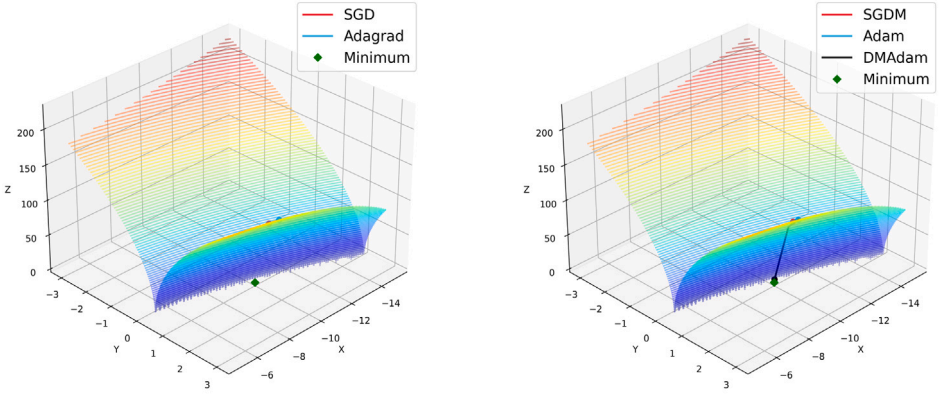
(a) Non-Momentum Algorithms (b) Momentum-Based Algorithms

Fig. 5. Convergence trajectories for the Beale function.



(a) Non-Momentum Algorithms (b) Momentum-Based Algorithms

Fig. 6. Convergence trajectories for the Goldstein-Price function.



(a) Non-Momentum Algorithms (b) Momentum-Based Algorithms

Fig. 7. Convergence trajectories for the Bukin function.

Assumption 1. The stochastic gradient is an unbiased estimate of the full gradient, i.e., $\mathbb{E}\left[\boldsymbol{g}^k \mid \mathcal{F}^k\right] = \nabla f\left(\boldsymbol{x}^k\right)$, and is uniformly almost surely bounded by some constant $G > 0$ such that $\left\|\boldsymbol{g}^k\right\| \leq G$ *a.s.*

Assumption 2. The function f is L-smooth, meaning that for all $\boldsymbol{x},\ \boldsymbol{y} \in \mathbb{R}^d$, there exists a constant $L > 0$ such that $\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|$. Additionally, f is lower-bounded with $f^* > -\infty$.

It is worth noting that Assumptions 1 and 2 are frequently applied in the analysis of stochastic optimization algorithms [19,31].

4.1. Minimum convergence and uniform convergence

Theorem 1. *Suppose that Assumptions 1 and 2 hold, and let* $\left\{\boldsymbol{x}^k\right\}_{k\geq 1}$ *be the sequence generated by Algorithm 1 with* $\eta_k = \frac{\eta_0}{\sqrt{k}}$ *and* $\beta_k = 1 - \frac{1}{\sqrt{k}}$. *For any* $K \geq 1$, *we have*

$$\min_{1\leq k\leq K} \mathbb{E}\left[\left\|\nabla f\left(\boldsymbol{x}^k\right)\right\|^2\right] = \mathcal{O}\left(\ln K/\sqrt{K}\right), \tag{14}$$

where η_0 *is a positive constant.*

Proof. See Appendix B.1. □

Theorem 2. *Under the same settings and assumptions as in Theorem 1, for any* $K \geq 1$, *with* τ *uniformly selected from* $\{1, 2, \ldots, K\}$, *we have*

$$\mathbb{E}\left[\left\|\nabla f\left(\boldsymbol{x}^\tau\right)\right\|^2\right] = \mathcal{O}\left(\ln K/\sqrt{K}\right). \tag{15}$$

Proof. See Appendix B.2. □

In the non-convex setting, Chen et al. [31] established a convergence rate of $\mathcal{O}(\ln K/\sqrt{K})$ for AMSGrad and AdaFom in expectation. Similarly, Zou et al. [32] reported the same convergence rate for Adam and Weighted AdaEMA. In line with these results, we demonstrate that Algorithm 1 achieves a convergence rate of $\mathcal{O}(\ln K/\sqrt{K})$ for both minimum convergence and uniform convergence, as shown in Theorems 1 and 2.

4.2. Non-ergodic convergence

Theorem 3 (*Non-ergodic Convergence*). *Suppose Assumptions 1 and 2 hold, and* $\left\{\boldsymbol{x}^k\right\}_{k\geq 1}$ *is the sequence generated by Algorithm 1. If the step sizes satisfy* $\sum_{k=1}^{\infty} \eta_k\left(1 - \beta_k\right) < \infty$, $\sum_{k=1}^{\infty} \eta_k^2 < \infty$, *and* η_k *is non-increasing, then we have*

$$\lim_{k\to\infty} \left\|\nabla f\left(\boldsymbol{x}^k\right)\right\| = 0 \ a.s., \ \text{and} \ \lim_{k\to\infty} \mathbb{E}\left[\left\|\nabla f\left(\boldsymbol{x}^k\right)\right\|\right] = 0. \tag{16}$$

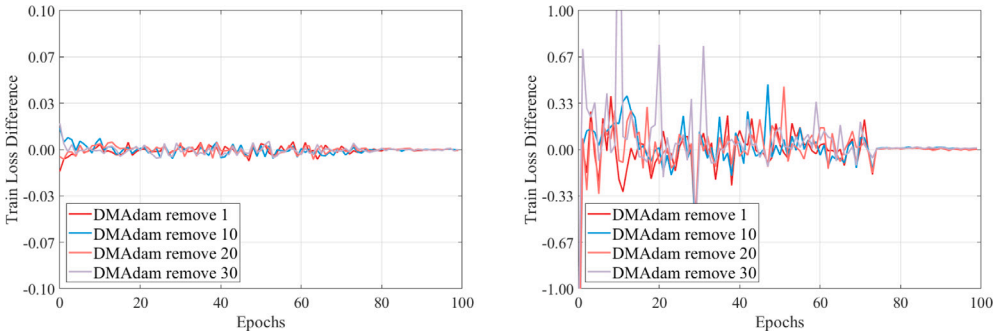


Fig. 8. Uniform stability error for DMAdam.

Proof. See Appendix B.3. □

In fact, the conditions in Theorem 3 are the same as those in Theorem 1. Therefore, to satisfy these conditions, it is sufficient to choose $\eta_k = \frac{\eta_0}{\sqrt{k}}$ and $\beta_k = 1 - \frac{1}{\sqrt{k}}$, where η_0 is a positive constant.

4.3. Uniform stability

Uniform stability provides a quantitative measure of the sensitivity of an algorithm to small changes in the dataset. Consider two samples S and S' of size n differing by one example, and let $\boldsymbol{x} = P(S)$ denote the output of a (potentially randomized) algorithm P (e.g., DMAdam) on data S [33].

Definition 1 ([34]). Algorithm P is ϵ-uniformly stable if

$$\epsilon_{stab} = \sup_{S,S'} \sup_{\xi \in \mathbb{P}} \mathbb{E}_P \left[\ell\left(P(S); \xi\right) - \ell\left(P\left(S'\right); \xi\right) \right] \leq \epsilon, \tag{17}$$

where ϵ_{stab} denotes the upper bound on the expected difference in the loss ℓ between outputs of P on samples S and S' that differ by only one example.

Fig. 8 illustrates the performance of VGG-13 trained on the CIFAR-10 dataset, with the curves highlighting differences between datasets with deleted samples and the complete dataset. Deleting a small number of samples (e.g., 1, 10, 20, or 30) from the training set has minimal impact on the training loss, indicating that DMAdam is insensitive to small data variations and can maintain stable model outputs. On the test set, the loss difference initially oscillates but gradually diminishes as training progresses, eventually approaching zero, indicating that the algorithm exhibits uniform stability.

5. Experiment

This section evaluates the performance of DMAdam through a series of numerical experiments. A comparative analysis is conducted with several algorithms, including SGD [5], Adam [11], NAdam [12], AMSGrad [13], MSVAG [14], AdamW [15], RAdam [16], AdaBelief [17], AdamP [18], Adan [19], and Lion [35]. Table 2 provides the hyperparameter settings employed in the DMAdam algorithm.

We obtain the hyperparameters in Table 2 from grid search results and optimize hyperparameters based on performance evaluations across various models and datasets. The hyperparameter η_k typically does not have a determining effect on experimental results, and using the default value is sufficient (see Fig. 17). However, fine-tuning η_k may improve experimental performance. In contrast, α_k is the most important hyperparameter and is usually preferred for tuning. Refer to Table 2 for α_k adjustment suggestions. To simplify the hyperparameter tuning process for researchers, η_k can be viewed as similar to β_1 in Adam, while α_k corresponds to the learning rate. The value of β_k is fixed at 0.999 and ϵ is set at 10^{-8}.

Table 2
Hyperparameter settings for the DMAdam.

Model	Dataset	η_k	α_k	β_k	ϵ	Weight decay
VGG-13	CIFAR-10	0.9	0.001	0.999	10^{-8}	0.0005
	CIFAR-100	2.1	0.001	0.999	10^{-8}	0.0005
ResNet-34	CIFAR-10	1.4	0.001	0.999	10^{-8}	0.0005
	CIFAR-100	1.2	0.001	0.999	10^{-8}	0.0005
LSTM-1	PTB	0.8	0.1	0.999	10^{-8}	0.0000012
LSTM-2		1.9	0.1	0.999	10^{-8}	0.0000012
LSTM-3		2.5	0.1	0.999	10^{-8}	0.0000012
YOLOv5 (s)	VOC (2007)	0.037	0.001	0.999	10^{-8}	0.0005
MobileNet-V3-L	Tiny ImageNet	1.5	0.005	0.999	10^{-8}	0.0001

To ensure the fairness and accuracy of the comparison, we use the parameters adopted for the comparison algorithms in their original papers. When the comparison algorithms fail to perform well on our task, we adjust their learning rates to optimize performance. Further details can be consulted in Appendix C.

5.1. Image classification experiment results

• Performance on CIFAR-10 Dataset

CIFAR-10, a widely used dataset for image classification, consists of 60,000 color images (32 × 32 pixels) distributed across 10 classes. In this study, we evaluate the performance of DMAdam against other optimization algorithms using VGG [36] and ResNet [37] models. As shown in Figs. 9 and 10, DMAdam achieves fast convergence and high accuracy during the training and testing of VGG-13 and ResNet-34. DMAdam outperforms traditional algorithms such as SGD and Adam, as well as recent improvements like RAdam, AdamP, and AdaBelief. After learning rate decay, DMAdam effectively mitigates oscillations and demonstrates high accuracy and stability in deep neural networks.

• Performance on CIFAR-100 Dataset

CIFAR-100 is an extension of CIFAR-10, consisting of 100 classes and providing a more challenging benchmark for optimization algorithms. As shown in Figs. 11 and 12, DMAdam performs similarly on the CIFAR-100 dataset as it does on CIFAR-10 when applied to VGG-13 and ResNet-34 models. DMAdam achieves higher accuracy compared to traditional algorithms like SGD and Adam, as well as improved methods such as AdamW, RAdam, and AdamP. Additionally, after 150 epochs of learning rate decay, DMAdam effectively maintains stability.

• Performance on Tiny ImageNet

Tiny ImageNet, a derivative of the ImageNet dataset, is widely used for training and evaluating deep learning models. It contains 200 categories, each comprising 500 training images, 50 validation images, and 50 unannotated test images at a resolution of 64 × 64 pixels. Since the test set lacks annotations, the validation set is commonly employed for evaluation. Despite its smaller scale, Tiny ImageNet