

## 820 A Methodology

821 **Conceptual illustration** The optimization landscape of neural networks is highly dynamic due to  
 822 data distribution and sampling, and sensitive to various factors, such as network architecture, opti-  
 823 mization and hyper-parameters. Figure 3(a) illustrates the inherent uncertainties during the training  
 824 process. The black-and-white surfaces reflect the loss landscape processing one batch, while the  
 825 colored surface represents the landscape following the next batch. The substantial variations in the  
 826 landscape between consecutive batches significantly affect the training trajectory, making it challeng-  
 827 ing to design a one-size-fits-all learning rate strategy that can effectively adapt to such fluctuations.  
 828 Therefore, we aim to develop a strategy a robust approach to be scheduled leaning rate across di-  
 829 verse optimization landscapes. We construct a budget-iteration-aware framework for learning rate  
 830 optimization model that guarantees minimal loss within constrained training iterations under the  
 831 worst-case conditions.

832 We analyze loss landscape by a quadratic approximation around nearby strict local optima. This  
 833 approach enables us to capture the key features of the loss surface near these points. Therefore,  
 834 the trajectory of optimization is impacted by the characteristics of the nearby strict local optima,  
 835 since the optimization process is inherently shaped by the loss landscape and the key features of the  
 836 surface are captured by these optima, as illustrated in Figure 3(b). Consequently, we can derive the  
 837 learning rate within the optimization model by the information of these nearby strict local optima.

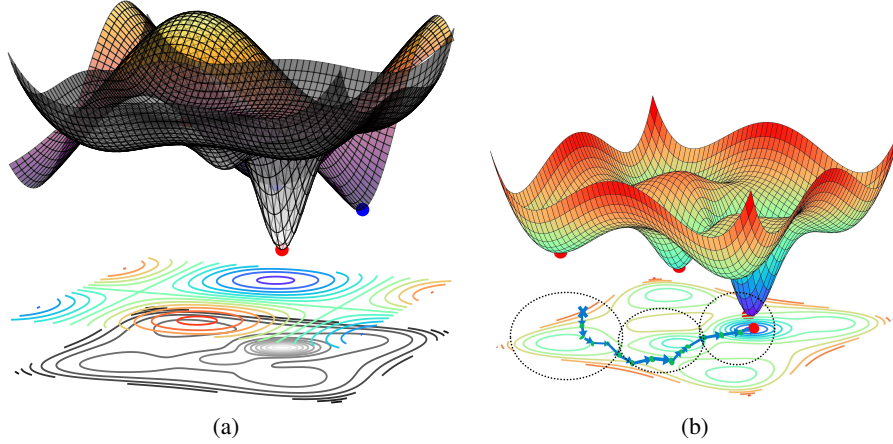


Figure 3: Conceptual illustration: visualization of the motivations behind the proposed modeling approaches. (a) shows the rationale for the modeling approach in (1), while (b) explains the modeling approach for (2). The solid line represents the optimization trajectory, while the dashed circles indicate the local approximation around corresponding minima.

838 **Assumptions** The optimization behavior of a neural network is fundamentally governed by the ge-  
 839 ometry of its loss landscape, which emerges from the interaction of parameter configuration, datasets  
 840 characteristics, sampling, network architecture, optimization preconditioning effect. In the vicinity  
 841 of any strict local minimum, the loss surface can be approximated by a quadratic method whose cur-  
 842 vature is determined by the local Hessian matrix. Thus the optimization trajectory is consequently  
 843 dominated by the geometric properties (e.g., Hessian features, curvature profiles) of the nearest strict  
 844 local minima.

845 Many tasks train models using epochs, where one epoch represents a full pass through the datasets.  
 846 Under this circumstance, an iteration occurs when the model processes a single batch. While the  
 847 number of iterations per epoch varies with batch size, training with a fixed epoch budget becomes  
 848 equivalent to using a fixed iteration count when batch size remains constant. For consistency, this  
 849 paper uses iterations as our primary unit of optimization model construction and networks training.

850 The sampling Hessian satisfies:  $\mathbb{E}_\xi [n_t n_t^\top] \preceq \sigma^2 H_f^{(k)}$  for some constant  $\sigma$ . This assumption is  
 851 followed by the work [15, 35]. Here,  $\sigma^2$  measures the degree of Hessian inconsistency across data  
 852 samples, i.e., the variance in the per-sample Hessian matrices. Besides, a smaller  $\sigma^2$  indicates a  
 853 more stable curvature landscape between each data sampling, leading to lower noise amplification  
 854 during optimization.

855 **Optimization problem derivation for Section 3.1** Considering the first constraint  $W_{t+1} = W_t -$   
 856  $\eta_t H_f^{(k)}(\xi)(W_t - \bar{W}^{(k)})$  in min-max optimization (2), we can reformulate it as

$$W_{t+1} - \bar{W}^{(k)} = (I - \eta_t H_f^{(k)}(\xi))(W_t - \bar{W}^{(k)}), \quad (9)$$

857 where  $I$  denotes the identity matrix. By iteratively applying this equation (9), we obtain

$$\begin{aligned} (W_{T_{k+1}} - \bar{W}^{(k)}) &= \\ (I - \eta_{T_{k+1}-1} H_f^{(k)}(\xi)) \cdots (I - \eta_{T_k+1} H_f^{(k)}(\xi)) (W_{T_k+1} - \bar{W}^{(k)}). \end{aligned} \quad (10)$$

858 Since the matrix  $H_f^{(k)}(\xi)$  is positive semi-definite, it possesses  $N$  eigenvalues  
 859  $\lambda_1^{(k)}(f(\xi)), \lambda_2^{(k)}(f(\xi)), \dots, \lambda_N^{(k)}(f(\xi))$  abbreviated as  $\lambda_i^{(k)}$ , which satisfy  $0 < \lambda_l^{(k)} \leq$   
 860  $\lambda_i^{(k)}(f(\xi)) \leq \lambda_u^{(k)}$  for all  $i$ . Here,  $\lambda_l^{(k)}$  and  $\lambda_u^{(k)}$  denote the bounds on the non-zero eigen-  
 861 values of the Hessian around  $k$ -th optimum. It also satisfies  $\lambda_u^{(k)} \leq \frac{2}{\eta}$  where  $\eta$  is the learning  
 862 rate to guarantee the convergence [55]. Moreover, there exist  $N$  linearly independent eigenvectors  
 863  $u_1^{(k)}(f(\xi)), u_2^{(k)}(f(\xi)), \dots, u_N^{(k)}(f(\xi))$  abbreviated as  $u_i^{(k)}$ , which can form the basis for the  $N$   
 864 dimension vector space. Then we have  $H_f^{(k)}(\xi)u_i^{(k)} = \lambda_i^{(k)}u_i^{(k)}$  ( $i = 1, 2, \dots, N$ ) and the initial  
 865 deviation from the optimal solution can be expressed as  $W_{T_k} - \bar{W}^{(k)} = \sum_{i=1}^N s_i^{(k)} u_i^{(k)}$  where  $s_i^{(k)}$  are  
 866 the coefficients corresponding to the eigenvector components. Thus, we obtain

$$\begin{aligned} \|W_{T_{k+1}} - \bar{W}^{(k)}\|_2^2 &= \sum_{i=1}^N (s_i^{(k)})^2 \|u_i^{(k)}\|_2^2 \left[ \prod_{t=T_k+1}^{T_{k+1}} (1 - \eta_t \lambda_i^{(k)}) \right]^2 \\ &\leq \sum_{i=1}^N (s_i^{(k)})^2 \|u_i^{(k)}\|_2^2 \max_{\lambda_l^{(k)} \leq \lambda_i^{(k)} \leq \lambda_u^{(k)}} \left[ \prod_{t=T_k+1}^{T_{k+1}} (1 - \eta_t \lambda_i^{(k)}) \right]^2 \\ &= \max_{\lambda_l^{(k)} \leq \lambda_i^{(k)} \leq \lambda_u^{(k)}} \left[ \prod_{t=T_k+1}^{T_{k+1}} (1 - \eta_t \lambda_i^{(k)}) \right]^2 \|W_{T_k} - \bar{W}^{(k)}\|_2^2 \end{aligned} \quad (11)$$

## 867 B Numerical solution and curve fitting

868 To solve the constrained min-max problem (4) as follow,

$$\begin{aligned} \min_{\eta_{1+T_k}, \eta_{2+T_k}, \dots, \eta_{T_{k+1}}} \max_{\lambda_l^{(k)} \leq \lambda_i^{(k)} \leq \lambda_u^{(k)}} \prod_{t=1+T_k}^{T_{k+1}} \left[ (1 - \eta_t \lambda_i^{(k)}) \right]^2 \\ \text{s.t. } \eta_{1+T_k}, \eta_{2+T_k}, \dots, \eta_{T_{k+1}} \in [\eta_{\min}, \eta_{\max}] \\ k = 1, 2, \dots, K \end{aligned}$$

869 we adopt an iterative projected gradient method that alternates between minimizing over the vari-  
870 ables  $\eta_t$  and maximizing over the parameters  $\lambda_i^{(k)}$ . Specifically, at each iteration, we update  $\eta_t$  with  
871 fixed  $\lambda_i^{(k)}$  as follows

$$\eta_t \leftarrow \mathcal{P}_{[\eta_{\min}, \eta_{\max}]} \left( \eta_t - \alpha \nabla_{\eta_j} \mathcal{L}(\eta_t, \lambda_i^{(k)}) \right).$$

872 For fixed  $\eta_t$ , we update  $\lambda_i^{(k)}$  via

$$\lambda_i^{(k)} \leftarrow \mathcal{P}_{[\lambda_l, \lambda_u]} \left( \lambda_i^{(k)} + \beta \nabla_{\lambda_i^{(k)}} \mathcal{L}(\eta_t, \lambda_i^{(k)}) \right),$$

873 where  $\mathcal{P}$  is the projection over box constraints,  $\alpha$  and  $\beta$  are step sizes.

874 This approach requires per-network numerical optimization to determine optimal learning rates case-  
875 by-case. To circumvent repeated numerical optimization for learning rate scheduling whenever a  
876 new network is trained, we propose to fit these solutions with a parametric function universally  
877 across networks.

878 We note that the solution of optimization model is an unordered set  $\{\eta_{1+T_k}, \eta_{2+T_k}, \dots, \eta_{T_{k+1}}\}$ , mean-  
879 ing that any permutation of these values yields the same objective value. To facilitate function  
880 approximation and improve the training stability of the network, we sort the solution set in both  
881 descending and ascending orders before fitting them with a smooth function. This pre-processing  
882 step preserves the optimality of the solution while enhancing the stability and interpretability of the  
883 fitted function.

884 Due to the symmetry between the descending and ascending orders, and without loss of generality,  
885 we first fit a function to the descending order solution. The corresponding function for the ascending  
886 order can then be easily obtained via a simple transformation.

887 Our key observation is that the numerical solutions under varying  $\lambda_l, \lambda_u$  configurations exhibit two  
888 characteristic decay modes. As shown in Figure 4, these transformations manifest through two  
889 distinct mechanisms: (i) gradual-to-accelerated decay: gentle initial decrease transitioning to rapid  
890 drop (as shown in Figures. 4(d), 4(e) and 4(f)); (ii) rapid-to-gradual decay: Initial steep decline  
891 followed by slow decay (as shown in Figures. 4(g), 4(h) and 4(i)). These pattern can be formulated  
892 as transformations of a base function, which resembles the curvature variation of the transformed  $\ell_1$   
893 norm modulated by a parameter [41].

894 We first isolate the base function by examining the limiting case where  $\lambda_l \approx \lambda_u$ . In this regime,  
895 the numerical solution within each interval  $[1 + T_k, T_{k+1}]$  manifests a cosine-like profile (Figures.  
896 4(a), 4(b) and 4(c)), suggesting the base form:

$$1 + \cos\left(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}\right)$$

897 Then we construct the fitting function as:

$$(\eta_{\max} - \eta_{\min}) \frac{a(1 + \cos(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}))}{b + c(1 + \cos(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}))} + \eta_{\min}. \quad (12)$$

898 where  $\{a, b, c\}$  control learning rate decay modes.

899 In addition, through systematic fitting across nine configurations (Table 4), we discover a universal  
900 scaling relationship among the parameters:

$$\frac{b + 2c}{a} \approx 2 \quad (13)$$

Table 4: The parameter details of curve fitting.

hyper-parameters				fitted parameters				objective value(log)(4)	
$\eta_{\min}$	$\eta_{\max}$	$\lambda_l^{(k)}$	$\lambda_u^{(k)}$	a	b	c	$\varphi$	by numerical solution	by fitting function
0	10	1.89	2.60	0.26	0.51	0.00	1.99	792.67	750.83
0	10	2.39	2.61	4.16	7.35	0.53	1.80	798.38	770.38
0	10	5.29	5.61	4.84	7.32	1.28	1.56	1118.79	1105.80
0	10	18.89	260.11	1124.67	212.28	1011.94	0.19	2933.34	2933.62
0	5	28.89	200.11	755.91	85.01	719.40	0.12	2607.11	2591.51
0	2	50.89	150.11	0.27	0.05	0.24	0.17	2031.66	2075.50
1	10	10.01	150.11	77.45	684.15	-268.20	8.40	2373.58	2431.42
2	10	18.89	50.11	0.11	2.52	-1.15	21.02	2007.68	2044.84
3	10	20.01	180.11	0.07	3.07	-1.47	38.34	2604.98	2632.21

Table 4 lists the fitted parameters  $\{a, b, c\}$ , where the mean absolute error is 0.04. Figure 5 visualizes the variability of the  $\frac{b+2c}{a}$  by shading the region within  $\pm 1$  standard deviation from the theoretical value in Figure 5. The empirical results falls within  $\pm 1\sigma$  of the predicted value ( $2.0 \pm 0.04$ ) in most of cases, demonstrating strong agreement with the assumption relation (13).

This empirical relationship (13) allows us to reduce the original 3-parameter system to 2 degrees of freedom as follows,

$$(\eta_{\max} - \eta_{\min}) \frac{2a(1 + \cos(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}))}{2b + (2a - b)(1 + \cos(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}))} + \eta_{\min}. \quad (14)$$

Actually, it is a 1-parameter system

$$(\eta_{\max} - \eta_{\min}) \frac{2(1 + \cos(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}))}{2d + (2 - d)(1 + \cos(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}))} + \eta_{\min}, \quad (15)$$

where  $\varphi := b/a$ . We list the value of fitted parameter  $\varphi$  in Table 4. Since the equation (14) and the equation (15) are mathematically equivalent, we adopt the equation (15) as the canonical representation throughout this work.

Then the corresponding function for the ascending order can then be easily obtained via a phase shift of  $\pi$  as follow,

$$(\eta_{\max} - \eta_{\min}) \frac{2(1 + \cos(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)} + \pi))}{2\varphi + (2 - \varphi)(1 + \cos(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)} + \pi))} + \eta_{\min}, \quad (16)$$

In summary, we obtain the  $\eta_t$  within the interval  $t \in [1 + T_k, T_{k+1}]$  as follows

$$\eta_t = (\eta_{\max} - \eta_{\min}) \frac{2(1 + \cos(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)} + (k-1)\pi))}{2\varphi + (2 - \varphi)(1 + \cos(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)} + (k-1)\pi))} + \eta_{\min}, \quad (17)$$

where  $d$  is the hyper-parameter controlling the variation speed of  $\eta_t$ .



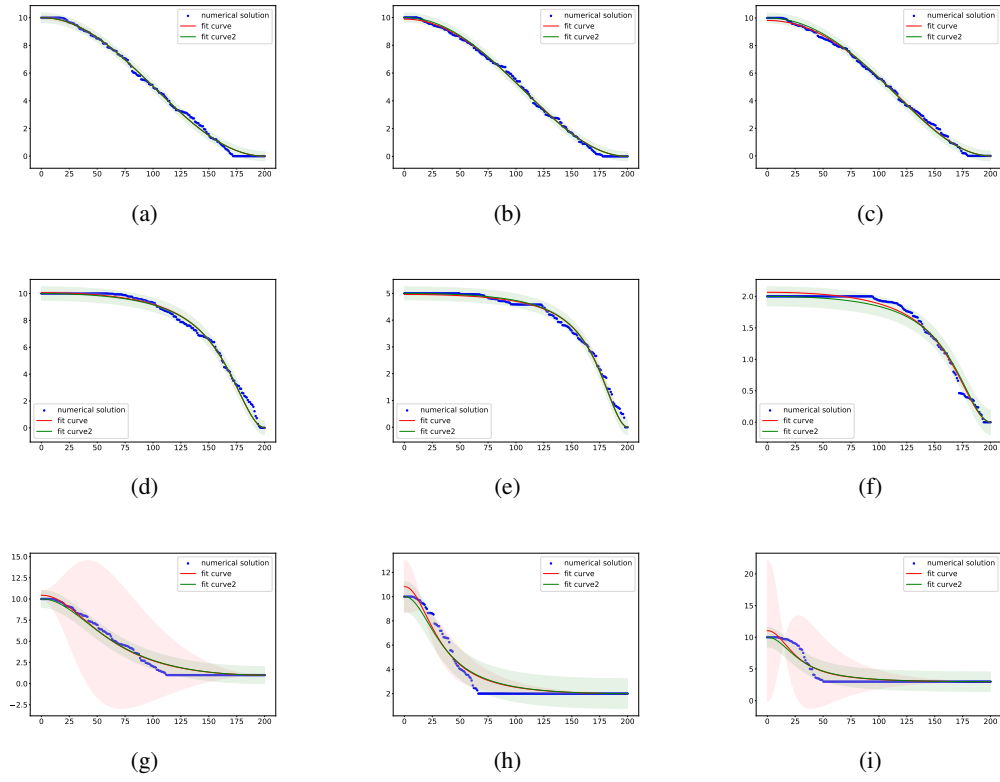


Figure 4: The curve fitting of numerical solutions.

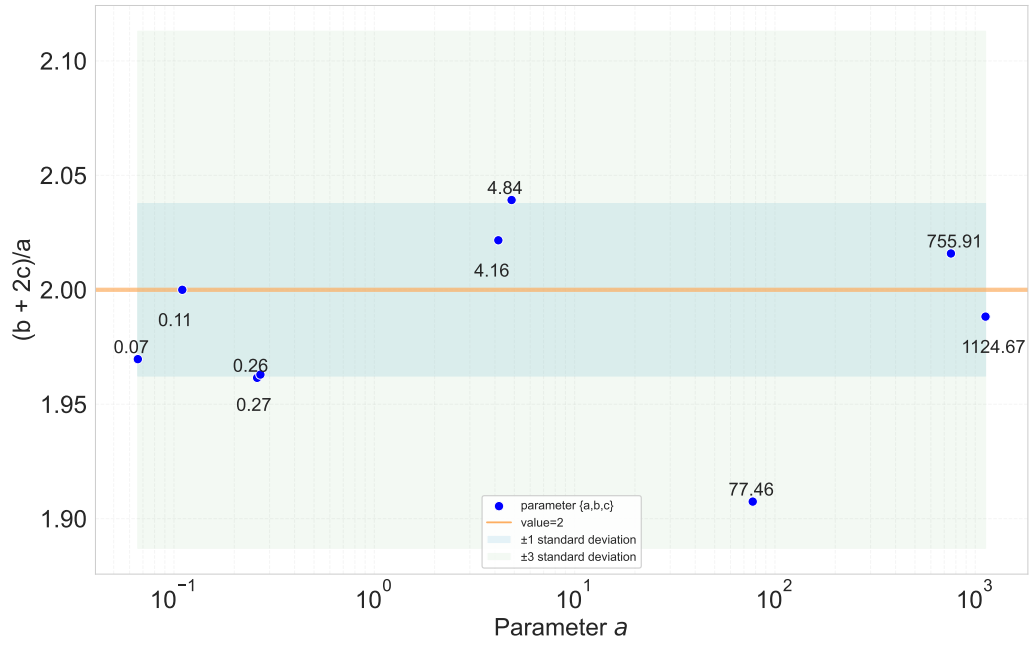


Figure 5: Visualization of relation between parameter  $a$ ,  $b$  and  $c$ .

## 915 C Adaptive simulation of existing schedules

916 Our schedule has adaptability owing to the parameter  $\varphi$ , enabling it to approximate the behavior of  
 917 existing schedules (e.g. step decay, cosine annealing, cyclic schedule or Rex schedule) through simple  
 918 parameter adjustments. More importantly, it outperforms these schedules by jointly optimizing  
 919 training stability and final accuracy.

920 We visualize the adaptive simulation of in Figure 6. By setting our parameter  $\varphi$  to 2, the UBA  
 921 schedule degenerates to a cosine schedule (figure 6(a)). By setting  $\varphi$  to large value such as 30,  
 922 the UBA schedule mimics exponential schedule (figure 6(b)). By setting  $\varphi$  to small value such  
 923 as 0.8, the UBA schedule mimics Rex schedule (figure 6(c)). By setting  $\varphi$  to 0, the proposed  
 924 schedule degenerates to a constant. Therefore, we can control  $\eta_{\max}$  as piece-wise value depending  
 925 on iteration, and the UBA schedule degenerates to a step schedule (figure 6(d)). By setting  $k$  to any  
 926 integer larger than 1, the UBA schedule possesses cyclic characters, thus it mimics cyclic schedule  
 927 (figure 6(e) 6(f)). Since the OneCycle learning rate schedule can be viewed as a single-period  
 928 version of cyclical learning rates, the UBA schedule mimics the OneCycle approach by emulating  
 929 the cyclical schedule.

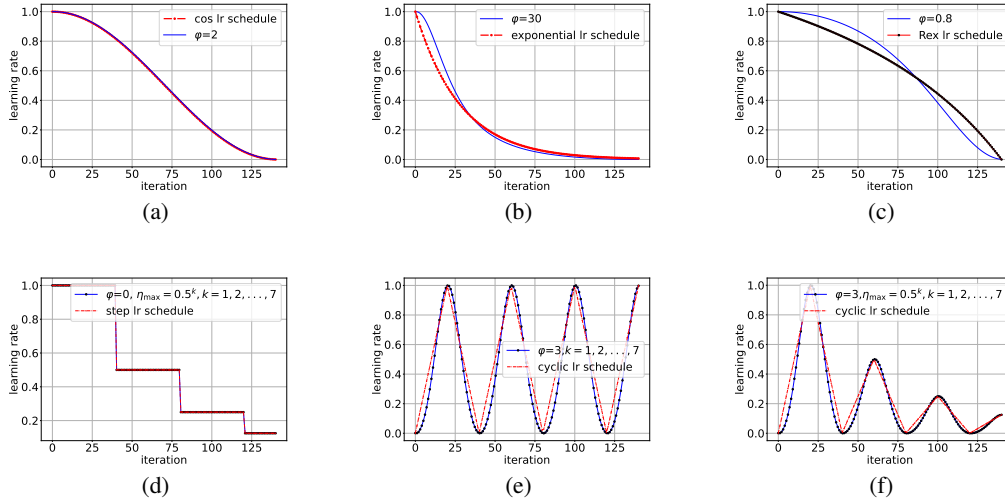


Figure 6: Adaptive simulation by UBA schedule.

## D Propositions and lemmas

### D.1 Proof of Proposition 1

**Proposition 1** *The fitted function (5) is the exact closed-form solution*  $\frac{2(1+\cos(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}))}{2\varphi+(2-\varphi)(1+\cos(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}))}$   
*to the min-max optimization problem:*

$$\min_{\eta_{1+T_k}, \eta_{2+T_k}, \dots, \eta_{T_{k+1}}} \max_{\lambda_l^{(k)} \leq \lambda_i^{(k)} \leq \lambda_u^{(k)}} \prod_{t=1+T_k}^{T_{k+1}} \left[ \left( 1 - \left( \frac{1}{\lambda_l^{(k)}} - \frac{1}{\lambda_u^{(k)}} \right) \eta_t + \frac{1}{\lambda_u^{(k)}} \right) \lambda_i^{(k)} \right]^2 \quad (18)$$

when the hyper-parameter  $\varphi$  are determined by  $\lambda_l^{(k)}$  and  $\lambda_u^{(k)}$  through the relation  $\varphi = 2 \frac{\lambda_u^{(k)}}{\lambda_l^{(k)}}$  and  $\eta_{\max} = 1, \quad \eta_{\min} = 0$ .

*Proof.* According to the theorem in [13], among all polynomials of degree  $n$  in  $\mu$  that take the value  $+1$  at  $\mu = d > 1$ , the Chebyshev polynomial  $S_n(\mu) = \frac{C_n(\mu)}{C_n(d)}$  is the unique solution that minimizes the maximum absolute value over the interval  $(-1, +1)$ . Here,  $C_n(\mu) = \cos(n \arccos(\mu))$  is the  $n$ -th order Chebyshev polynomial.

We define the polynomial  $Q(\lambda_i^{(k)})$  as:

$$Q(\lambda_i^{(k)}) := \prod_{t=T_k+1}^{T_{k+1}} \left[ 1 - \left( \left( \frac{1}{\lambda_l^{(k)}} - \frac{1}{\lambda_u^{(k)}} \right) \eta_t + \frac{1}{\lambda_u^{(k)}} \right) \lambda_i^{(k)} \right]$$

which is a degree  $n = T_{k+1} - T_k$  polynomial in  $\lambda_i^{(k)}$ .

Next, we introduce the variable transformation:

$$\gamma = \frac{-2\lambda_i^{(k)} + \lambda_l^{(k)} + \lambda_u^{(k)}}{\lambda_u^{(k)} - \lambda_l^{(k)}}$$

which maps the interval  $\lambda_l^{(k)} \leq \lambda_i^{(k)} \leq \lambda_u^{(k)}$  to the interval  $-1 \leq \gamma \leq 1$ , such that  $\lambda_i^{(k)} = \lambda_u^{(k)}$  corresponds to  $\gamma = -1$  and  $\lambda_i^{(k)} = \lambda_l^{(k)}$  corresponds to  $\gamma = 1$ .

The polynomial  $P(\gamma) = Q(\lambda_i^{(k)})$  now satisfies the condition in the theorem from [13], i.e.,

$$P\left(\frac{\lambda_l^{(k)} + \lambda_u^{(k)}}{\lambda_u^{(k)} - \lambda_l^{(k)}}\right) = 1 \quad \text{since} \quad Q(0) = 1.$$

Therefore, the original min-max problem

$$\min_{\eta_{T_k+1}, \eta_{T_k+2}, \dots, \eta_{T_{k+1}}} \max_{\lambda_l^{(k)} \leq \lambda_i^{(k)} \leq \lambda_u^{(k)}} \prod_{t=T_k+1}^{T_{k+1}} \left[ 1 - \left( \left( \frac{1}{\lambda_l^{(k)}} - \frac{1}{\lambda_u^{(k)}} \right) \eta_t + \frac{1}{\lambda_u^{(k)}} \right) \lambda_i^{(k)} \right]^2$$

is equivalent to the problem of finding a polynomial in  $\gamma$  of degree  $T_{k+1} - T_k$  that minimizes the maximum absolute value in the interval  $-1 \leq \gamma \leq 1$ .

By the theorem in [13], the desired solution to the problem is the Chebyshev polynomial:

$$S_{T_{k+1}-T_k}(\gamma) = \frac{C_{T_{k+1}-T_k}(\gamma)}{C_{T_{k+1}-T_k}\left(\frac{\lambda_l^{(k)} + \lambda_u^{(k)}}{\lambda_u^{(k)} - \lambda_l^{(k)}}\right)}$$

To match  $P(\gamma) = S_{T_{k+1}-T_k}(\gamma)$ , we equate the corresponding zeros. The roots of  $Q(\lambda_i^{(k)}) = 0$  are given by:

$$\lambda_i^{(k)} = \frac{1}{\left( \left( \frac{1}{\lambda_l^{(k)}} - \frac{1}{\lambda_u^{(k)}} \right) \eta_t + \frac{1}{\lambda_u^{(k)}} \right)} \quad \text{for} \quad t = 1, 2, \dots, T_{k+1} - T_k.$$

On the other hand, the zeros of  $P(\gamma)$  correspond to the values of  $\gamma$  where the polynomial  $P(\gamma)$  vanishes, and these zeros are given by:

$$\gamma_t = \frac{\lambda_l^{(k)} + \lambda_u^{(k)}}{\lambda_u^{(k)} - \lambda_l^{(k)}} - \frac{2}{\left(\left(\frac{1}{\lambda_l^{(k)}} - \frac{1}{\lambda_u^{(k)}}\right)\eta_t + \frac{1}{\lambda_u^{(k)}}\right)(\lambda_u^{(k)} - \lambda_l^{(k)})} \quad \text{for } t = 1, 2, \dots, T_{k+1} - T_k.$$

The zeros of the Chebyshev polynomial  $S_{T_{k+1}-T_k}(\gamma)$  are given by the values:

$$\cos\left(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}\right) \quad \text{for } t = 1, 2, \dots, T_{k+1} - T_k.$$

Equating the zeros of  $S_{T_{k+1}-T_k}(\gamma)$  and  $P(\gamma)$ , we have:

$$\frac{\lambda_l^{(k)} + \lambda_u^{(k)}}{\lambda_u^{(k)} - \lambda_l^{(k)}} - \frac{2}{\left(\left(\frac{1}{\lambda_l^{(k)}} - \frac{1}{\lambda_u^{(k)}}\right)\eta_t + \frac{1}{\lambda_u^{(k)}}\right)(\lambda_u^{(k)} - \lambda_l^{(k)})} = \cos\left(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}\right)$$

Solving this equation for  $\eta_t$ , we obtain:

$$\eta_t = \frac{(1 + \cos\left(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}\right))}{2\frac{\lambda_u^{(k)}}{\lambda_l^{(k)}} - \left(\frac{\lambda_u^{(k)}}{\lambda_l^{(k)}} - 1\right)\left(1 + \cos\left(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}\right)\right)}.$$

□

## D.2 Proof of Proposition 2

**Proposition 2** Consider the training process within the interval  $t \in [1 + T_k, T_{k+1}]$ . When the hyper-parameter  $\varphi$  is set sufficiently close to 2, the proposed learning rate scheduling formula reduces to the cosine learning rate schedule.

*Proof.*

$$\begin{aligned} & \lim_{\varphi \rightarrow 2} (\eta_{\max} - \eta_{\min}) \frac{2(1 + \cos(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}))}{2\varphi + (2 - \varphi)(1 + \cos(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}))} + \eta_{\min} \\ &= (\eta_{\max} - \eta_{\min}) \left(\frac{1}{2} + \frac{1}{2} \cos\left(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}\right)\right) + \eta_{\min} \\ &= \frac{1}{2} (\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{(t - T_k)\pi}{T_{k+1} - T_k}\right) \cos\left(\frac{\pi}{2(T_{k+1} - T_k)}\right) + \sin\left(\frac{(t - T_k)\pi}{T_{k+1} - T_k}\right) \sin\left(\frac{\pi}{2(T_{k+1} - T_k)}\right)\right) + \eta_{\min} \\ &\approx \frac{1}{2} (\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{(t - T_k)\pi}{T_{k+1} - T_k}\right)\right) + \eta_{\min} \end{aligned} \tag{19}$$

The final approximation holds because each training phase typically contains many iterations, i.e., making  $\frac{\pi}{T_{k+1} - T_k} \approx 0$ . Consequently,  $\sin(\frac{\pi}{2(T_{k+1} - T_k)}) \approx 0$  and  $\cos(\frac{\pi}{2(T_{k+1} - T_k)}) \approx 1$ . □

## D.3 Proof of Theorem 1

**Lemma 1.** Let  $\eta_j$  be the learning rate at the iteration  $j \in [1 + T_k, t]$ , defined by the proposed form (5). Let  $\lambda_i^{(k)}$  be the eigenvalues of the Hessian matrix  $H_f^{(k)}(\xi)$  at the  $k$ -th strict minimum  $\bar{W}^{(k)}$ . Then for any  $t \in [1 + T_k, T_{k+1}]$ , the following inequalities hold,

968 **Case 1** ( $\varphi > 2$ ):

$$\begin{aligned}
& \prod_{j=1+T_k}^t \left[ (1 - \eta_j \lambda_i^{(k)}) \right]^2 \\
& \leq \exp \left( -2\lambda_i^{(k)} \eta_{\min}(t - T_k) \right) \\
& \quad \cdot \left( \frac{4(T_{k+1} - T_k) + (\varphi - 2)\pi}{4(T_{k+1} - T_k) + (\varphi - 2)\pi(t - T_k)} \right)^{\frac{4\lambda_i^{(k)}(\eta_{\max} - \eta_{\min}) \left( 1 + \cos \left( \frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \right) (T_{k+1} - T_k)}{(\varphi - 2)\pi}}
\end{aligned} \tag{20}$$

969 **Case 2** ( $\varphi < 2$ ):

$$\begin{aligned}
& \prod_{j=1+T_k}^t \left[ (1 - \eta_j \lambda_i^{(k)}) \right]^2 \\
& \leq \exp \left( -2\lambda_i^{(k)} \eta_{\min}(t - T_k) \right) \\
& \quad \cdot \left( \frac{(2\varphi + 2\pi - \varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1} - T_k)}(t - T_k - 0.5)}{(2\varphi + 2\pi - \varphi\pi) + \frac{(2-\varphi)\pi}{2(T_{k+1} - T_k)}} \right)^{\frac{4\lambda_i^{(k)}(\eta_{\max} - \eta_{\min}) \left( 1 + \cos \left( \frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \right) (T_{k+1} - T_k)}{(2 - \varphi)\pi}}
\end{aligned} \tag{21}$$

970 *Proof.* By the fact that  $1 - x \leq \exp(-x)$ , we have

$$\prod_{j=1+T_k}^t \left[ (1 - \eta_j \lambda_i^{(k)}) \right]^2 \leq \exp \left( -2 \sum_{j=1+T_k}^t \eta_j \lambda_i^{(k)} \right) \tag{22}$$

971 Substituting the expression for  $\eta_j$  by the proposed form (5) yields:

$$\begin{aligned}
& \exp \left( -2 \sum_{j=1+T_k}^t \eta_j \lambda_i^{(k)} \right) \\
& = \exp \left( -2\lambda_i^{(k)} \sum_{j=1+T_k}^t (\eta_{\max} - \eta_{\min}) \frac{2(1 + \cos \left( \frac{(2(j - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right))}{2\varphi + (2 - \varphi)(1 + \cos \left( \frac{(2(j - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right))} + \eta_{\min} \right) \\
& = \exp \left( -2\lambda_i^{(k)} \eta_{\min}(t - T_k) \right) \exp \left( -2\lambda_i^{(k)} \sum_{j=1+T_k}^t (\eta_{\max} - \eta_{\min}) \frac{2 \left( 1 + \cos \left( \frac{(2(j - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \right)}{2\varphi + (2 - \varphi) \left( 1 + \cos \left( \frac{(2(j - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \right)} \right)
\end{aligned} \tag{23}$$

972 The behavior of this term depends on the value of the parameter  $\varphi$ . We consider the following cases:

973 • if  $\varphi \geq 2, 2 - \varphi \leq 0$

974 Since

$$\begin{aligned}
& \cos \left( \frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \geq \cos \left( \frac{(2(j - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \geq 1 - \left( \frac{(2(j - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \\
& \geq 1 - \pi \left( \frac{(j - T_k)}{(T_{k+1} - T_k)} \right),
\end{aligned} \tag{24}$$

we have

$$\begin{aligned}
& \sum_{j=1+T_k}^t \frac{1}{2\varphi + (2-\varphi) \left(1 + \cos \left( \frac{(2(j-T_k)-1)\pi}{2(T_{k+1}-T_k)} \right) \right)} \\
& \leq \sum_{j=1+T_k}^t \frac{1}{2\varphi + (2-\varphi) \left(1 + 1 - \pi \left( \frac{(j-T_k)}{(T_{k+1}-T_k)} \right) \right)} \\
& = \sum_{j=1+T_k}^t \frac{1}{4 + \frac{(\varphi-2)\pi}{(T_{k+1}-T_k)}(j-T_k)}
\end{aligned} \tag{25}$$

Then the equality (23) becomes

$$\begin{aligned}
& \exp \left( -2\lambda_i^{(k)} \eta_{\min}(t-T_k) \right) \exp \left( -2\lambda_i^{(k)} \sum_{j=1+T_k}^t (\eta_{\max} - \eta_{\min}) \frac{2 \left(1 + \cos \left( \frac{(2(j-T_k)-1)\pi}{2(T_{k+1}-T_k)} \right) \right)}{2\varphi + (2-\varphi) \left(1 + \cos \left( \frac{(2(j-T_k)-1)\pi}{2(T_{k+1}-T_k)} \right) \right)} \right) \\
& \leq \exp \left( -2\lambda_i^{(k)} \eta_{\min}(t-T_k) \right) \exp \left[ -4\lambda_i^{(k)} (\eta_{\max} - \eta_{\min}) \left(1 + \cos \left( \frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)} \right) \right) \right. \\
& \quad \left. \sum_{j=1+T_k}^t \frac{1}{2\varphi + (2-\varphi) \left(1 + \cos \left( \frac{(2(j-T_k)-1)\pi}{2(T_{k+1}-T_k)} \right) \right)} \right] \\
& \leq \exp \left( -2\lambda_i^{(k)} \eta_{\min}(t-T_k) \right) \exp \left[ -4\lambda_i^{(k)} (\eta_{\max} - \eta_{\min}) \left(1 + \cos \left( \frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)} \right) \right) \right. \\
& \quad \left. \sum_{j=1+T_k}^t \frac{1}{4 + \frac{(\varphi-2)\pi}{(T_{k+1}-T_k)}(j-T_k)} \right]
\end{aligned} \tag{26}$$

Note that the function  $h(j) := \frac{1}{4 + \frac{(\varphi-2)\pi}{(T_{k+1}-T_k)}(j-T_k)}$  is monotone decreasing over  $j \in [1 +$

$T_k, t]$ , so we can bound the summation from below by the integral:

$$\begin{aligned}
& \sum_{j=1+T_k}^t \frac{1}{4 + \frac{(\varphi-2)\pi}{(T_{k+1}-T_k)}(j-T_k)} \\
& \geq \int_{1+T_k}^{t+1} \frac{1}{4 + \frac{(\varphi-2)\pi}{(T_{k+1}-T_k)}(j-T_k)} dj \\
& \geq \int_{1+T_k}^t \frac{1}{4 + \frac{(\varphi-2)\pi}{(T_{k+1}-T_k)}(j-T_k)} dj \\
& = \frac{(T_{k+1}-T_k)}{(\varphi-2)\pi} \ln \left( \frac{4(T_{k+1}-T_k) + (\varphi-2)\pi(t-T_k)}{4(T_{k+1}-T_k) + (\varphi-2)\pi} \right)
\end{aligned} \tag{27}$$

Thus the inequality (26) becomes

$$\begin{aligned}
& \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \exp\left[-4\lambda_i^{(k)}(\eta_{\max}-\eta_{\min})\left(1+\cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)\right. \\
& \quad \left.\sum_{j=1+T_k}^t \frac{1}{4+\frac{(\varphi-2)\pi}{(T_{k+1}-T_k)}(j-T_k)}\right] \\
& \leq \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \exp\left[4\lambda_i^{(k)}(\eta_{\max}-\eta_{\min})\left(1+\cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)\right. \\
& \quad \left.\frac{(T_{k+1}-T_k)}{(\varphi-2)\pi} \ln\left(\frac{4(T_{k+1}-T_k)+(\varphi-2)\pi}{4(T_{k+1}-T_k)+(\varphi-2)\pi(t-T_k)}\right)\right] \\
& = \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \left(\frac{4(T_{k+1}-T_k)+(\varphi-2)\pi}{4(T_{k+1}-T_k)+(\varphi-2)\pi(t-T_k)}\right)^{\frac{4\lambda_i^{(k)}(\eta_{\max}-\eta_{\min})\left(1+\cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)(T_{k+1}-T_k)}{(\varphi-2)\pi}} \\
& \leq \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \left(\frac{4(T_{k+1}-T_k)+(\varphi-2)\pi}{4(T_{k+1}-T_k)+(\varphi-2)\pi(t-T_k)}\right)^{\frac{4\lambda_i^{(k)}(\eta_{\max}-\eta_{\min})\left(1+\cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)(T_{k+1}-T_k)}{(\varphi-2)\pi}}
\end{aligned} \tag{28}$$

The last inequality is because  $\lambda_i^{(k)} \leq \lambda_i^{(k)}$  and  $\frac{4(T_{k+1}-T_k)+(\varphi-2)\pi}{4(T_{k+1}-T_k)+(\varphi-2)\pi(t-T_k)} \leq 1$ .

- if  $\varphi \leq 2$ ,  $2-\varphi \geq 0$  Since

$$1 + \cos\left(\frac{(2(j-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right) \leq \pi - \frac{(2(j-T_k)-1)\pi}{2(T_{k+1}-T_k)}. \tag{29}$$

We obtain

$$\begin{aligned}
& \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \exp\left(-2\lambda_i^{(k)} \sum_{j=1+T_k}^t (\eta_{\max}-\eta_{\min}) \frac{2\left(1+\cos\left(\frac{(2(j-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)}{2\varphi+(2-\varphi)\left(1+\cos\left(\frac{(2(j-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)}\right) \\
& \leq \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \exp\left[-4\lambda_i^{(k)}(\eta_{\max}-\eta_{\min})\left(1+\cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)\right. \\
& \quad \left.\sum_{j=1+T_k}^t \frac{1}{2\varphi+(2-\varphi)\left(1+\cos\left(\frac{(2(j-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)}\right] \\
& \leq \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \exp\left[-4\lambda_i^{(k)}(\eta_{\max}-\eta_{\min})\left(1+\cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)\right. \\
& \quad \left.\sum_{j=1+T_k}^t \left(\frac{1}{(2\varphi+2\pi-\varphi\pi)-\frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(j-T_k-0.5)}\right)\right]
\end{aligned} \tag{30}$$

983 because function  $h(j) := \frac{1}{(2\varphi+2\pi-\varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(j-T_k-0.5)}$  is monotone increasing in the  
 984 range  $[1+T_k, +\infty)$ , then it holds that

$$\begin{aligned}
 & \sum_{j=1+T_k}^t \frac{1}{(2\varphi+2\pi-\varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(j-T_k-0.5)} \\
 & \geq \int_{T_k}^t \frac{1}{(2\varphi+2\pi-\varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(j-T_k-0.5)} dj \\
 & = -\frac{(T_{k+1}-T_k)}{(2-\varphi)\pi} \ln \left( \frac{(2\varphi+2\pi-\varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(t-T_k-0.5)}{(2\varphi+2\pi-\varphi\pi) + \frac{(2-\varphi)\pi}{2(T_{k+1}-T_k)}} \right)
 \end{aligned} \tag{31}$$

985 Thus, we have

$$\begin{aligned}
 & \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \exp\left[-4\lambda_i^{(k)}(\eta_{\max}-\eta_{\min})\left(1+\cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)\right. \\
 & \quad \left.\sum_{j=1+T_k}^t \left(\frac{1}{(2\varphi+2\pi-\varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(j-T_k-0.5)}\right)\right] \\
 & \leq \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \exp\left[4\lambda_i^{(k)}(\eta_{\max}-\eta_{\min})\left(1+\cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)\right. \\
 & \quad \left.\frac{(T_{k+1}-T_k)}{(2-\varphi)\pi} \ln \left(\frac{(2\varphi+2\pi-\varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(t-T_k-0.5)}{(2\varphi+2\pi-\varphi\pi) + \frac{(2-\varphi)\pi}{2(T_{k+1}-T_k)}}\right)\right] \\
 & \leq \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \\
 & \quad \cdot \left(\frac{(2\varphi+2\pi-\varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(t-T_k-0.5)}{(2\varphi+2\pi-\varphi\pi) + \frac{(2-\varphi)\pi}{2(T_{k+1}-T_k)}}\right)^{\frac{4\lambda_i^{(k)}(\eta_{\max}-\eta_{\min})\left(1+\cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)}{(2-\varphi)\pi}(T_{k+1}-T_k)} \\
 & \leq \exp\left(-2\lambda_i^{(k)}\eta_{\min}(t-T_k)\right) \\
 & \quad \cdot \left(\frac{(2\varphi+2\pi-\varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(t-T_k-0.5)}{(2\varphi+2\pi-\varphi\pi) + \frac{(2-\varphi)\pi}{2(T_{k+1}-T_k)}}\right)^{\frac{4\lambda_i^{(k)}(\eta_{\max}-\eta_{\min})\left(1+\cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)}{(2-\varphi)\pi}(T_{k+1}-T_k)}
 \end{aligned} \tag{32}$$

986 The last inequality is because  $\lambda_l^{(k)} \leq \lambda_i^{(k)}$  and  $\left(\frac{(2\varphi+2\pi-\varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(t-T_k-0.5)}{(2\varphi+2\pi-\varphi\pi) + \frac{(2-\varphi)\pi}{2(T_{k+1}-T_k)}}\right) \leq 1$ .

987 □

988 **Theorem 1** Let  $n_t = H_f^{(k)}(W_t - \bar{W}^{(k)}) - H_f^{(k)}(\xi)(W_t - \bar{W}^{(k)})$  be the stochastic curvature noise  
 989 introduced by sampling at iteration  $t$ . Assume that the sampling Hessian satisfies:  $\mathbb{E}_\xi[n_t n_t^\top] \preceq$   
 990  $\sigma^2 H_f^{(k)}$  for some constant  $\sigma$ .

991 • If we set the learning rate as the proposed form (5) where hyper-parameter  $\varphi > 2$ , the loss  
 992 uncertainty introduced by stochastic gradient method within the interval  $t \in [1+T_k, T_{k+1}]$



can be quantified as two terms, bias term and variance term, bounded as follows,

$$\begin{aligned}
& \mathbb{E} \left[ \mathcal{L}(f(W_t, \xi)) - \mathcal{L}(f(\bar{W}^{(k)}, \xi)) \right] \\
& \leq \left( \frac{4(T_{k+1} - T_k) + (\varphi - 2)\pi}{4(T_{k+1} - T_k) + (\varphi - 2)\pi(t - T_k)} \right) \frac{4\lambda_l^{(k)}(\eta_{\max} - \eta_{\min}) \left( 1 + \cos \left( \frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \right) (T_{k+1} - T_k)}{(\varphi - 2)\pi} \\
& \quad \cdot \exp \left( -2\lambda_i^{(k)} \eta_{\min}(t - T_k) \right) \lambda_u^{(k)} \|W_{1+T_k} - \bar{W}^{(k)}\|_2^2 \\
& + \sigma^2 \sum_{i=1+T_k}^t \eta_i^2 \sum_{j=1}^N (\lambda_j^{(k)})^2 \exp \left( -2\lambda_j^{(k)} \eta_{\min}(t - i) \right) \\
& \quad \cdot \left( \frac{4(T_{k+1} - T_k) + (\varphi - 2)\pi(i - T_k)}{4(T_{k+1} - T_k) + (\varphi - 2)\pi(t - T_k)} \right) \frac{4\lambda_l^{(k)}(\eta_{\max} - \eta_{\min}) \left( 1 + \cos \left( \frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \right) (T_{k+1} - T_k)}{(\varphi - 2)\pi}
\end{aligned} \tag{33}$$

- If we set the learning rate as the proposed form (5) where hyper-parameter  $\varphi < 2$ , the loss uncertainty introduced by stochastic gradient method within the interval  $t \in [1 + T_k, T_{k+1}]$  can be quantified as two terms, bias term and variance term, bounded as follows,

$$\begin{aligned}
& \mathbb{E} \left[ \mathcal{L}(f(W_t, \xi)) - \mathcal{L}(f(\bar{W}^{(k)}, \xi)) \right] \\
& \leq \left( \frac{(2\varphi + 2\pi - \varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(t - T_k - 0.5)}{(2\varphi + 2\pi - \varphi\pi) + \frac{(2-\varphi)\pi}{2(T_{k+1}-T_k)}} \right) \frac{4\lambda_i^{(k)}(\eta_{\max} - \eta_{\min}) \left( 1 + \cos \left( \frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \right) (T_{k+1} - T_k)}{(2-\varphi)\pi} \\
& \quad \cdot \exp \left( -2\lambda_i^{(k)} \eta_{\min}(t - T_k) \right) \lambda_u^{(k)} \|W_{1+T_k} - \bar{W}^{(k)}\|_2^2 \\
& + \leq \sigma^2 \sum_{i=1+T_k}^t \eta_i^2 \sum_{j=1}^N (\lambda_j^{(k)})^2 \exp \left( -2\lambda_j^{(k)} \eta_{\min}(t - i) \right) \\
& \quad \cdot \left( \frac{(2\varphi + 2\pi - \varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(t - T_k - 0.5)}{(2\varphi + 2\pi - \varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(i - T_k - 0.5)} \right) \frac{4\lambda_i^{(k)}(\eta_{\max} - \eta_{\min}) \left( 1 + \cos \left( \frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)} \right) \right) (T_{k+1} - T_k)}{(2-\varphi)\pi}
\end{aligned} \tag{34}$$

*Proof.* Let  $H_f^{(k)}$  denote full batch Hessian at  $k$ -th minimum, such that  $\mathbb{E}(H_f^{(k)}(\xi)) = H_f^{(k)}$ , and

denote  $n_t = H_f^{(k)}(W_t - \bar{W}^{(k)}) - H_f^{(k)}(\xi)(W_t - \bar{W}^{(k)})$ , which indicates the bias. The uncertainty

brought by stochastic gradient method is measured as follows,

$$\begin{aligned}
W_{t+1} - \bar{W}^{(k)} &= W_t - \bar{W}^{(k)} - \eta_t H_f^{(k)}(\xi)(W_t - \bar{W}^{(k)}) \\
&= W_t - \bar{W}^{(k)} - \eta_t H_f^{(k)}(W_t - \bar{W}^{(k)}) \\
&\quad + \eta_t H_f^{(k)}(W_t - \bar{W}^{(k)}) - \eta_t H_f^{(k)}(\xi)(W_t - \bar{W}^{(k)}) \\
&= (I - \eta_t H_f^{(k)})(W_t - \bar{W}^{(k)}) + \eta_t n_t \\
&= (I - \eta_t H_f^{(k)})(I - \eta_{t-1} H_f^{(k)}) \cdots (I - \eta_{t-q} H_f^{(k)})(W_{t-q} - \bar{W}^{(k)}) \\
&\quad + \sum_{i=t-q}^t (I - \eta_t H_f^{(k)})(I - \eta_{t-1} H_f^{(k)}) \cdots (I - \eta_{i+1} H_f^{(k)}) \eta_i n_i \\
&:= P_t^{(k)} P_{t-1}^{(k)} \cdots P_{t-q}^{(k)}(W_{t-q} - \bar{W}^{(k)}) + \sum_{i=t-q}^t P_t^{(k)} P_{t-1}^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i
\end{aligned} \tag{35}$$

1000 where  $P_i^{(k)} := (I - \eta_i H_f^{(k)})$ .

1001 According to the approximation approach of the objective function (1) in Section 3.1, i.e.  
 1002  $\mathcal{L}(f(W, \xi)) \approx \mathcal{L}(f(\bar{W}^{(k)}, \xi)) + \frac{1}{2}(W - \bar{W}^{(k)})^\top H_f^{(k)}(\xi)(W - \bar{W}^{(k)})$ , the loss uncertainty in-  
 1003 troduced by stochastic gradient method within the interval  $t \in [1 + T_k, T_{k+1}]$  is quantified as  
 1004  $\mathbb{E}[(W_{T_{k+1}} - \bar{W}^{(k)})^\top H_f^{(k)}(W_{T_{k+1}} - \bar{W}^{(k)})]$ , where the loss function is approximated by leverag-  
 1005 ing the full batch Hessian.

$$\begin{aligned}
 & \mathbb{E} \left[ \mathcal{L}(f(W_t, \xi)) - \mathcal{L}(f(\bar{W}^{(k)}, \xi)) \right] = \mathbb{E} \left[ (W_t - \bar{W}^{(k)})^\top H_f^{(k)}(W_t - \bar{W}^{(k)}) \right] \\
 &= \mathbb{E} \left[ \left( P_t^{(k)} P_{t-1}^{(k)} \cdots P_{1+T_k}^{(k)} (W_{1+T_k} - \bar{W}^{(k)}) + \sum_{i=1+T_k}^t P_t^{(k)} P_{T_{k+1}-1}^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right)^\top \right. \\
 & \quad \left. H_f^{(k)} \left( P_t^{(k)} P_{t-1}^{(k)} \cdots P_{1+T_k}^{(k)} (W_{1+T_k} - \bar{W}^{(k)}) + \sum_{i=1+T_k}^t P_t^{(k)} P_{t-1}^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right) \right] \\
 &= \mathbb{E} \left[ (W_t - \bar{W}^{(k)})^\top P_{1+T_k}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{1+T_k}^{(k)} (W_t - \bar{W}^{(k)}) \right] \\
 & \quad + 2\mathbb{E} \left[ (W_t - \bar{W}^{(k)})^\top P_{1+T_k}^{(k)} \cdots P_t^{(k)} H_f^{(k)} \left( \sum_{i=1+T_k}^t P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right) \right] \\
 & \quad + \mathbb{E} \left( \sum_{i=1+T_k}^t P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right)^\top H_f^{(k)} \left( \sum_{i=1+T_k}^t P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right)
 \end{aligned} \tag{36}$$

1006 Since  $\mathbb{E}[n_t] = 0$ ,  $\mathbb{E} \left[ (W_t - \bar{W}^{(k)})^\top P_{1+T_k}^{(k)} \cdots P_t^{(k)} H_f^{(k)} \left( \sum_{i=1+T_k}^t P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right) \right] = 0$ .  
 1007 Meanwhile, we notice the data samplings between different iteration are independent, thus  
 1008  $\mathbb{E}[(n_i)^\top n_j] = 0$  ( $i \neq j$ ). We obtain

$$\begin{aligned}
 & \mathbb{E} \left( \sum_{i=1+T_k}^t P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right)^\top H_f^{(k)} \left( \sum_{i=1+T_k}^t P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right) \\
 &= \mathbb{E} \left( \sum_{i,j=1+T_k}^t \eta_i n_i^\top P_{i+1}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{j+1}^{(k)} \eta_j n_j \right) \\
 &= \sum_{i=1+T_k}^t \mathbb{E} \left( \eta_i n_i^\top P_{i+1}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right)
 \end{aligned} \tag{37}$$

1009 Therefore, Eq.(36) can be reformulated as follows,

$$\begin{aligned}
 & \mathbb{E} \left( \sum_{i=1+T_k}^t P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right)^\top H_f^{(k)} \left( \sum_{i=1+T_k}^t P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right) \\
 &= \mathbb{E} \left[ (W_t - \bar{W}^{(k)})^\top P_{1+T_k}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{1+T_k}^{(k)} (W_t - \bar{W}^{(k)}) \right] \\
 & \quad + \sum_{i=1+T_k}^t \mathbb{E} \left( \eta_i n_i^\top P_{i+1}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right) \\
 &:= B + V
 \end{aligned} \tag{38}$$

1010 where we denote  $B := \mathbb{E} \left[ (W_t - \bar{W}^{(k)})^\top P_{1+T_k}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{1+T_k}^{(k)} (W_t - \bar{W}^{(k)}) \right]$  bias  
 1011 term and  $V := \sum_{i=1+T_k}^t \mathbb{E} \left( \eta_i n_i^\top P_{i+1}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right)$  variance term.

1012 Denote  $u_i^{(k)}$   $i \in \{1, 2, \dots, N\}$  are linearly independent eigenvectors, which can form the basis  
 1013 for the  $N$  dimension vector space. Then we have  $H_f^{(k)}(\xi)u_i^{(k)} = \lambda_i^{(k)}u_i^{(k)}$  ( $i = 1, 2, \dots, N$ ) and  
 1014 the initial deviation from the optimal solution can be expressed as  $W_t - \bar{W}^{(k)} = \sum_{i=1}^N s_i^{(k)}u_i^{(k)}$  where  
 1015  $s_i^{(k)}$  are the coefficients corresponding to the eigenvector components. Combined with Lemma 1,  
 1016 we obtain  
 1017 **Case 1** ( $\varphi > 2$ ):

$$\begin{aligned}
 B &= (W_t - \bar{W}^{(k)})^\top P_{1+T_k}^{(k)} \dots P_t^{(k)} H_f^{(k)} P_t^{(k)} \dots P_{1+T_k}^{(k)} (W_t - \bar{W}^{(k)}) \\
 &= \left[ (I - \eta_t H_f^{(k)}) \dots (I - \eta_{1+T_k} H_f^{(k)}) (W_{1+T_k} - \bar{W}^{(k)}) \right]^\top H_f^{(k)} (I - \eta_t H_f^{(k)}) \dots (I - \eta_{1+T_k} H_f^{(k)}) (W_{1+T_k} - \bar{W}^{(k)}) \\
 &= \sum_{i=1}^N (s_i^{(k)})^2 \|u_i^{(k)}\|_2^2 \lambda_i^{(k)} \prod_{j=1+T_k}^t \left[ (1 - \eta_j \lambda_i^{(k)}) \right]^2 \\
 &\leq \sum_{i=1}^N (s_i^{(k)})^2 \|u_i^{(k)}\|_2^2 \lambda_i^{(k)} \exp\left(-2\lambda_i^{(k)} \eta_{\min}(t - T_k)\right) \\
 &\quad \cdot \left( \frac{4(T_{k+1} - T_k) + (\varphi - 2)\pi}{4(T_{k+1} - T_k) + (\varphi - 2)\pi(t - T_k)} \right)^{\frac{4\lambda_i^{(k)}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}\right)\right)}{(\varphi - 2)\pi}} (T_{k+1} - T_k) \\
 &\leq \left( \frac{4(T_{k+1} - T_k) + (\varphi - 2)\pi}{4(T_{k+1} - T_k) + (\varphi - 2)\pi(t - T_k)} \right)^{\frac{4\lambda_i^{(k)}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}\right)\right)}{(\varphi - 2)\pi}} (T_{k+1} - T_k) \\
 &\quad \cdot \exp\left(-2\lambda_i^{(k)} \eta_{\min}(t - T_k)\right) \lambda_u^{(k)} \|W_{1+T_k} - \bar{W}^{(k)}\|_2^2
 \end{aligned} \tag{39}$$

1018 In addition,

$$\begin{aligned}
 V &= \sum_{i=1+T_k}^t \mathbb{E} \left( \eta_i n_i^\top P_{i+1}^{(k)} \dots P_t^{(k)} H_f^{(k)} P_t^{(k)} \dots P_{i+1}^{(k)} \eta_i n_i \right) \\
 &= \sum_{i=1+T_k}^t \eta_i^2 \mathbb{E} \left[ \text{tr} \left( n_i^\top P_{i+1}^{(k)} \dots P_t^{(k)} H_f^{(k)} P_t^{(k)} \dots P_{i+1}^{(k)} n_i \right) \right] \\
 &= \sum_{i=1+T_k}^t \eta_i^2 \mathbb{E} \left[ \text{tr} \left( P_{i+1}^{(k)} \dots P_t^{(k)} H_f^{(k)} P_t^{(k)} \dots P_{i+1}^{(k)} n_i n_i^\top \right) \right] \\
 &= \sum_{i=1+T_k}^t \eta_i^2 \text{tr} \left[ P_{i+1}^{(k)} \dots P_t^{(k)} H_f^{(k)} P_t^{(k)} \dots P_{i+1}^{(k)} \mathbb{E} (n_i n_i^\top) \right] \\
 &\leq \sigma^2 \sum_{i=1+T_k}^t \eta_i^2 \text{tr} \left[ P_{i+1}^{(k)} \dots P_t^{(k)} H_f^{(k)} P_t^{(k)} \dots P_{i+1}^{(k)} H_f^{(k)} \right] \\
 &= \sigma^2 \sum_{i=1+T_k}^t \eta_i^2 \sum_{j=1}^N (\lambda_j^{(k)})^2 \prod_{q=i+1}^t \left( 1 - \eta_q \lambda_j^{(k)} \right)^2 \\
 &\leq \sigma^2 \sum_{i=1+T_k}^t \eta_i^2 \sum_{j=1}^N (\lambda_j^{(k)})^2 \exp\left(-2\lambda_j^{(k)} \eta_{\min}(t - i)\right) \\
 &\quad \cdot \left( \frac{4(T_{k+1} - T_k) + (\varphi - 2)\pi(i - T_k)}{4(T_{k+1} - T_k) + (\varphi - 2)\pi(t - T_k)} \right)^{\frac{4\lambda_i^{(k)}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{(2(t - T_k) - 1)\pi}{2(T_{k+1} - T_k)}\right)\right)}{(\varphi - 2)\pi}} (T_{k+1} - T_k)
 \end{aligned} \tag{40}$$

1019 where the second equality comes from cyclic property of trace and the first inequality comes from  
 1020  $\mathbb{E}_\xi [n_t n_t^\top] \preceq \sigma^2 H_f^{(k)}$ .

1021 **Case 2** ( $\varphi < 2$ ):

$$\begin{aligned}
 B &= (W_t - \bar{W}^{(k)})^\top P_{1+T_k}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{1+T_k}^{(k)} (W_t - \bar{W}^{(k)}) \\
 &= \left[ (I - \eta_t H_f^{(k)}) \cdots (I - \eta_{1+T_k} H_f^{(k)}) (W_{1+T_k} - \bar{W}^{(k)}) \right]^\top H_f^{(k)} (I - \eta_t H_f^{(k)}) \cdots (I - \eta_{1+T_k} H_f^{(k)}) (W_{1+T_k} - \bar{W}^{(k)}) \\
 &= \sum_{i=1}^N (s_i^{(k)})^2 \|u_i^{(k)}\|_2^2 \lambda_i^{(k)} \prod_{j=1+T_k}^t \left[ (1 - \eta_j \lambda_i^{(k)}) \right]^2 \\
 &\leq \sum_{i=1}^N (s_i^{(k)})^2 \|u_i^{(k)}\|_2^2 \lambda_i^{(k)} \exp\left(-2\lambda_i^{(k)} \eta_{\min}(t - T_k)\right) \\
 &\quad \cdot \left( \frac{(2\varphi + 2\pi - \varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(t - T_k - 0.5)}{(2\varphi + 2\pi - \varphi\pi) + \frac{(2-\varphi)\pi}{2(T_{k+1}-T_k)}} \right)^{\frac{4\lambda_i^{(k)}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)(T_{k+1}-T_k)}{(2-\varphi)\pi}} \\
 &\leq \left( \frac{(2\varphi + 2\pi - \varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(t - T_k - 0.5)}{(2\varphi + 2\pi - \varphi\pi) + \frac{(2-\varphi)\pi}{2(T_{k+1}-T_k)}} \right)^{\frac{4\lambda_i^{(k)}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)(T_{k+1}-T_k)}{(2-\varphi)\pi}} \\
 &\quad \cdot \exp\left(-2\lambda_i^{(k)} \eta_{\min}(t - T_k)\right) \lambda_u^{(k)} \|W_{1+T_k} - \bar{W}^{(k)}\|_2^2
 \end{aligned} \tag{41}$$

1022 In addition,

$$\begin{aligned}
 V &= \sum_{i=1+T_k}^t \mathbb{E} \left( \eta_i n_i^\top P_{i+1}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{i+1}^{(k)} \eta_i n_i \right) \\
 &= \sum_{i=1+T_k}^t \eta_i^2 \mathbb{E} \left[ \text{tr} \left( n_i^\top P_{i+1}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{i+1}^{(k)} n_i \right) \right] \\
 &= \sum_{i=1+T_k}^t \eta_i^2 \mathbb{E} \left[ \text{tr} \left( P_{i+1}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{i+1}^{(k)} n_i n_i^\top \right) \right] \\
 &= \sum_{i=1+T_k}^t \eta_i^2 \text{tr} \left[ P_{i+1}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{i+1}^{(k)} \mathbb{E} (n_i n_i^\top) \right] \\
 &\leq \sigma^2 \sum_{i=1+T_k}^t \eta_i^2 \text{tr} \left[ P_{i+1}^{(k)} \cdots P_t^{(k)} H_f^{(k)} P_t^{(k)} \cdots P_{i+1}^{(k)} H_f^{(k)} \right] \\
 &= \sigma^2 \sum_{i=1+T_k}^t \eta_i^2 \sum_{j=1}^N (\lambda_j^{(k)})^2 \prod_{q=i+1}^t \left( 1 - \eta_q \lambda_j^{(k)} \right)^2 \\
 &\leq \sigma^2 \sum_{i=1+T_k}^t \eta_i^2 \sum_{j=1}^N (\lambda_j^{(k)})^2 \exp\left(-2\lambda_j^{(k)} \eta_{\min}(t - i)\right) \\
 &\quad \cdot \left( \frac{(2\varphi + 2\pi - \varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(t - T_k - 0.5)}{(2\varphi + 2\pi - \varphi\pi) - \frac{(2-\varphi)\pi}{(T_{k+1}-T_k)}(i - T_k - 0.5)} \right)^{\frac{4\lambda_i^{(k)}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{(2(t-T_k)-1)\pi}{2(T_{k+1}-T_k)}\right)\right)(T_{k+1}-T_k)}{(2-\varphi)\pi}}
 \end{aligned} \tag{42}$$

1023 where the second equality comes from cyclic property of trace and the first inequality comes from  
 1024  $\mathbb{E}_\xi [n_t n_t^\top] \preceq \sigma^2 H_f^{(k)}$ .

1025  $\square$

1026 In this proof, we employ assumption  $\mathbb{E}_\xi [n_t n_t^\top] \preceq \sigma^2 H_f^{(k)}$  used in the work [15, 35]. Moreover, we  
1027 acknowledge that the idea of the proof is inspired by the approach introduced in [15, 35]

## E Experimental details

### E.1 Baselines

We detail the baselines as follows,

- **Step schedule(SS)** reduces the learning rate in a piecewise manner. It implements discrete learning rate drops at predefined epoch intervals following schedule  $g(t) = d_{\lfloor t/t_s \rfloor}$ , where  $t_s$  is the predefined decaying step and  $d_i (d_i \geq d_{i+1})$  is predefined decaying scalar [15]. A typical implementation decreases the learning rate by a factor of 0.1 after 50% of the epochs and further by a factor of 0.01 after 75% of the epochs [20]. However, it requires careful tuning of step timing since abrupt changes may destabilize optimization. In this paper, we employ such setting of step schedule ( $\times 0.1$  after 50% and  $\times 0.01$  after 75%) for all our experiments.
- **Cosine schedule(CS)** controls the  $t$ -th iteration learning rate  $\eta_t$  following the mathematical formulation as  $\eta_t = \eta_{min} + \frac{1}{2}(\eta_0 - \eta_{min})(1 + \cos(\frac{t\pi}{T}))$  where  $T$  is total iterations. It provides smooth transition between learning rates, shown to improve final model accuracy step schedule. Moreover, it is proposed to mitigate abrupt decreases in the learning rate that could hinder models from escaping local minima. This approach has demonstrated effectiveness, particularly in transformer training [32]. Besides, it includes restart mechanisms where  $T$  becomes the cyclic period.
- **Cyclical Learning Rates (CLR)** oscillates between base learning rate  $\eta_{min}$  and maximal learning rate  $\eta_{max}$  with triangular and triangular2 policies. The cyclic property help traverse loss landscape barrier, so that it enables neural networks to train significantly faster, often by an order of magnitude, compared to standard training methods [42, 43]. Besides, the commonly-used OneCycle learning rate schedule can be seen as the one period version of Cyclical Learning Rates. In our experiments, this baseline **CLR** includes Cyclical and OneCycle schedules(**1C**). For CLR, we set linear mode for each period and period is two.
- **Budgeted training(BT)** is conducted by the proposed budget-aware setting of existing learning rate schedule under epoch budget [29]. It reformulates standard schedules such as linear, step and cosine schedule as functions of remaining budget. The classic linear decay version is formulated as  $\eta_t = \eta_0 \times (1 - t/T)$  where  $T$  is total iterations. It is particularly effective for hyper-parameter tuning.
- **Reflected Exponential (REX)** controls the  $t$ -th iteration learning rate  $\eta_t$  following the mathematical formulation as  $\eta_t = \eta_0 \left( \frac{1-t/T}{0.5+0.5 \times (1-t/T)} \right)$ . It is inspired by the performance of delayed linear schedules, aggressively reducing the learning rate near the end of training and reflecting an exponential decay [5].

### E.2 Evaluation benchmarks of language task

We evaluate models on a diverse set of open benchmarks. The benchmarks are listed in Table 5. For language models evaluation, we adopt the [14] for standardized performance comparisons.

### E.3 Implementation Details

Consistent with CIFAR, ImageNet and OLMo practices, we use step-wise scheduling for all training procedures.

**Vision tasks on CIFAR** CIFAR10/100 are simple benchmark datasets that are widely used for quick and efficient evaluation of deep learning tasks. CIFAR10 consists of a training split of 50000 examples and a test split of 10000 examples. Each example is a  $32 \times 32$  color image in 10 classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck). CIFAR100 comprises 60000  $32 \times 32$  pixels color images which are divided into 100 classes. Each class contains 600 images and the classes are grouped into 20 super-classes. Each image comes with a fine label (the class to which it belongs) and a coarse label (the super-class to which it belongs). We use the `torch.optim.SGD` and `torch.optim.AdamW` API to configure the optimizer. For vision tasks in Section 4.1, we all adopt the SGD optimizer. For ablation studies in Section 4.3, we adopt the AdamW optimizer. We

Table 5: Evaluation benchmarks of language task

Datasets name	abbreviation
ARC-Easy [10]	ARC-E
ARC-Challenge [10]	ARC-C
HellaSwag [54]	HSWAG
PIQA [3]	PIQA
WinoGrande [38]	WG
OpenbookQA [34]	OBQA
BoolQ [9]	BoolQ
SciQ [49]	SciQ
COPA [17]	COPA
CommonsenseQA [45]	CSQA
SocialIQA [39]	SIQA
MMLU stem [21]	STEM
MMLU humanities [21]	HUM
MMLU social sci [21]	SOC
MMLU other [21]	OTH

1077 initialize the learning rate to  $1e-1$ , set the batch size to 128. Each epoch consists of  $50000/128 + 1$   
1078  $= 391$  steps. We leave the weight decay value as  $5e-4$ , and complete the training task on PyTorch  
1079 2.4.1+cu118 with RTX 5880 and RTX 3090.

1080 **Vision tasks on ImageNet** To evaluate the scalability of our schedule on larger-scale dataset, we  
1081 conduct experiments on the ImageNet dataset, a dataset that more closely reflects real-world vi-  
1082 sual recognition challenges. ImageNet consists of approximately 1.28 million training images and  
1083 50000 validation images across 1000 classes, which also come with an official dataset split. This  
1084 benchmark, particularly with the ResNet-50 architecture, has been extensively studied and is widely  
1085 regarded as a standard evaluation setup. We adopt the common ResNet50 architecture, and sepa-  
1086 rately train it with different learning rate schedulers. To provide a comprehensive evaluation, we  
1087 conduct experiments using both the classical SGD optimizer and the popularized AdamW optimizer.  
1088 For the experiments with SGD, we set the peak learning rate to 0.5 and apply the weight decay  
1089 factor of  $1e-4$ . For the experiments using AdamW, the peak learning rate is set to  $3e-3$ , with a decou-  
1090 pled weight decay of 0.02. Following default configuration which requires learning rate warmup,  
1091 we integrate 10% warmup phases of the total training tokens into all schedules. For schedules that  
1092 inherently include a warmup-like behavior, such as the 1C schedule, where the learning rate initially  
1093 rises from zero to a target value, we retain their original settings during the ascending phase. We set  
1094 the batch size to 2048 and complete these training tasks with 8 NVIDIA A100 GPUs.

1095 **Language tasks** For language tasks, we use the OLMo model. OLMo [18] is a decoder-only  
1096 transformer-based language model that builds upon the vanilla Transformer architecture [48]. It  
1097 incorporates several key improvements inspired by recent large language models (LLMs) such as  
1098 PaLM [8], the LLaMA family [47], OpenLM and Falcon [1]. Notably, OLMo stands out as a  
1099 truly open language model, offering full transparency through its open training data, released train-  
1100 ing/evaluation code, intermediate model checkpoints, and comprehensive training logs, making it a  
1101 valuable resource for reproducibility and further research.

1102 For the training hyper-parameters on OLMo, we primarily adopt the configuration outlined in OLMo  
1103 [18]. We conduct experiment on H100 and A100. Following default configuration which requires  
1104 learning rate warmup, we integrate 10% warmup phases of the total training tokens into all sched-  
1105 ules. For schedules that inherently include a warmup-like behavior, such as the 1C schedule, where  
1106 the learning rate initially rises from zero to a target value, we retain their original settings during the  
1107 ascending phase. Besides, large language models commonly provide multiple scales to accommo-  
1108 date different compute constraints. We adjust both model size and training steps to explore budgets.  
1109 To ensure robust conclusions, we maintain a data-to-parameters ratio (D/N) of 500 across scales.  
1110 The specific configurations are as follows, 36M-parameter model is trained on 50 billion tokens,  
1111 73M-parameter model is trained on 50 billion tokens, 151M-parameter model is trained on 75 bil-  
1112 lion tokens, 300M-parameter model is trained on 150 billion tokens. This scaling strategy allows us

1113 to systematically study the relationship between model size, training data, and performance while  
 1114 controlling for the D/N ratio. The OLMo model is trained using AdamW optimizer. The detailed  
 1115 configuration are shown in Table 6.

Table 6: Configurations of OLMo

Model scale	36M	73M	151M	300M
token number	50B	50B	75B	150B
Layers ( $L$ )	12	16	20	24
Hidden dim ( $D$ )	512	640	768	1024
Attention heads	8	10	8	8
Inner dim ( $H$ )	1280	1536	2240	2688
Vocab size ( $V$ )		100,352		

1116 The code of UBA schedule is available at <https://github.com/Ttt-answer/UBA.git>.

#### 1117 E.4 Overall experiment results for Vision Classification Tasks

1118 Overall results for the vision classification tasks are presented in Table 7, depicting validation  
 1119 accuracy on vision benchmarks (e.g., CIFAR10/100 and ImageNet) using different architectures  
 1120 (VGG16, ResNet18, ResNet34, ResNet50). It is obvious that our proposed UBA schedule out-  
 1121 perform baselines, achieving the highest validation accuracy over both small-scale and large-scale  
 1122 benchmarks across all training budgets.

1123 Our analysis reveals the following key findings. (i) On CIFAR10-ResNet18 and CIFAR100-  
 1124 ResNet34 task, UBA shows the strongest performance across all training budgets. On the large-  
 1125 scale ImageNet benchmark with ResNet50, UBA maintains consistent superiority. The consistent  
 1126 improvements on both small-scale and large-scale benchmarks highlight UBA’s **generalizability**  
 1127 **across scales and data regimes**. (ii) UBA outperforms baselines not only at 100% training budget  
 1128 but also at 25% and 50% iteration budgets, demonstrating its **budget efficiency**, i.e., effectiveness  
 1129 in computation-constrained scenarios. (iii) Notably, UBA shows robust performance even while the  
 1130 second-best schedule varies depending on both the datasets, architectures and training budgets. This  
 1131 instability among competing schedules suggests their performance is highly sensitive to datasets-  
 1132 architectures characteristics and training dynamics at different learning phases. Therefore, for prac-  
 1133 titioners seeking reliable schedules without extensive method selection, UBA provides a default-  
 1134 strong choice. UBA eliminates the need for case-by-case baseline comparison and delivers **stable**  
 1135 **superiority and reliability** regardless of datasets, architectures, training budgets and scales.

#### 1136 E.5 Overall experiment results for language models

1137 For most tasks, we conduct a comprehensive evaluation across six representative baselines to ensure  
 1138 broad coverage. For large model with 300M parameters, we focus our comparison on the top three  
 1139 performing baselines due to the computational tractability.

1140 The overall performance of the OLMo models is presented in Table 8. We also plot training curves  
 1141 in Figure 7, 8, 9 and 10, showing training and validation losses along with downstream evaluation  
 1142 results across model sizes ranging from 36M to 300M parameters and training data scales from 50B  
 1143 to 150B tokens. As shown, UBA schedule consistently achieves lower training and validation losses  
 1144 while delivering superior downstream performance.

1145 From Table 8, UBA achieves state-of-the-art performance on approximately 50% of the benchmarks  
 1146 across all scales while the second-best schedule varies. Furthermore, it achieves *consistent supe-*  
 1147 *rior average performance* among all baselines. It highlights the stable superiority and reliability  
 1148 of UBA, providing a *default-strong choice*. Moreover, it demonstrates significant improvements  
 1149 in SciQ-73M(+1.7) and ARC-E-300M(+2.63), highlighting its ability to enhance generalization  
 1150 across diverse benchmarks. Besides, in Figure 2, UBA consistently achieves lower training loss  
 1151 and validation loss throughout the training, indicating the *efficient training ability* and downstream  
 1152 performance enhancement. Notably, while task-specific tuning of  $\varphi$  can further improve model per-  
 1153 formance, we intentionally fix its value across all tasks and architectures to ensure fair evaluation.



Table 7: Generalization accuracy.

Dataset	Network	Method	training budget (epoch(%))			
			75 (25%)	150(50%)	300(100%)	
CIFAR10	ResNet18	SS	92.41	93.90	93.94	
		CS	93.66	94.15	95.40	
		CLR	91.89	92.23	95.32	
		1C	94.36	95.02	95.48	
		BT	93.24	94.28	95.55	
		REX	<b>94.79</b>	94.96	95.59	
		UBA(ours)	<i>94.54</i>	<b>95.26</b>	<b>95.74</b>	
			75 (25%)	150(50%)	300(100%)	
CIFAR100	VGG16	SS	67.97	69.91	72.17	
		CS	<i>71.07</i>	71.37	72.16	
		CLR	70.38	69.90	70.91	
		1C	70.69	71.28	73.24	
		BT	69.05	70.90	71.74	
		REX	69.84	<i>71.46</i>	72.73	
		UBA(ours)	<b>71.75</b>	<b>73.57</b>	<b>75.16</b>	
				30 (25%)	60(50%)	120(100%)
	ResNet34	SS	70.85	75.58	77.28	
		CS	63.88	68.84	70.43	
		CLR	72.63	73.81	74.80	
		1C	73.72	75.53	77.90	
		BT	72.53	75.40	78.49	
		REX	73.12	75.48	77.99	
		UBA(ours)	<b>74.57</b>	<b>76.68</b>	<b>78.97</b>	
			75 (25%)	150(50%)	300(100%)	
ImageNet	ResNet50	SS	74.84	76.96	78.10	
		CS	<i>75.99</i>	77.79	<i>79.10</i>	
		CLR	72.87	74.88	76.91	
		1C	75.28	77.37	78.79	
		BT	75.71	77.48	78.66	
		REX	75.28	77.03	78.46	
		UBA(ours)	<b>76.00</b>	<b>77.99</b>	<b>79.32</b>	

Remarkably, even with this universal  $\varphi$  setting, UBA consistently outperforms baselines on diverse benchmarks, demonstrating *inherent robustness* to varying benchmarks and model scales.

## E.6 Performance across different optimizers

Our scheduling strategy originates from the solution to a optimization problem under gradient descent dynamics. While modern optimizer (e.g., AdamW [31]) introduce additional momentum and adaptive mechanisms, they maintain the fundamental property of performing gradient-based updates. To verify the performance of schedule, we conduct cross-optimizer ablation studies. We evaluate the model on CIFAR100 and ImageNet datasets using AdamW optimizer. For AdamW, we use a learning rate of 0.003 and weight decay of 0.0001. We report the validation accuracy for each configuration. The results are summarized in Table 9.

As shown in Table 9, our UBA schedule achieves state-of-the-art validation accuracy on both SGD and AdamW optimizers, consistently outperforming baselines across CIFAR100-ResNet34 and ImageNet-ResNet50 benchmarks. This demonstrates that although our schedule is theoretically derived from standard gradient descent dynamics, it generalizes effectively to modern optimizers like AdamW despite their additional momentum and adaptive mechanisms highlighting its broad applicability.

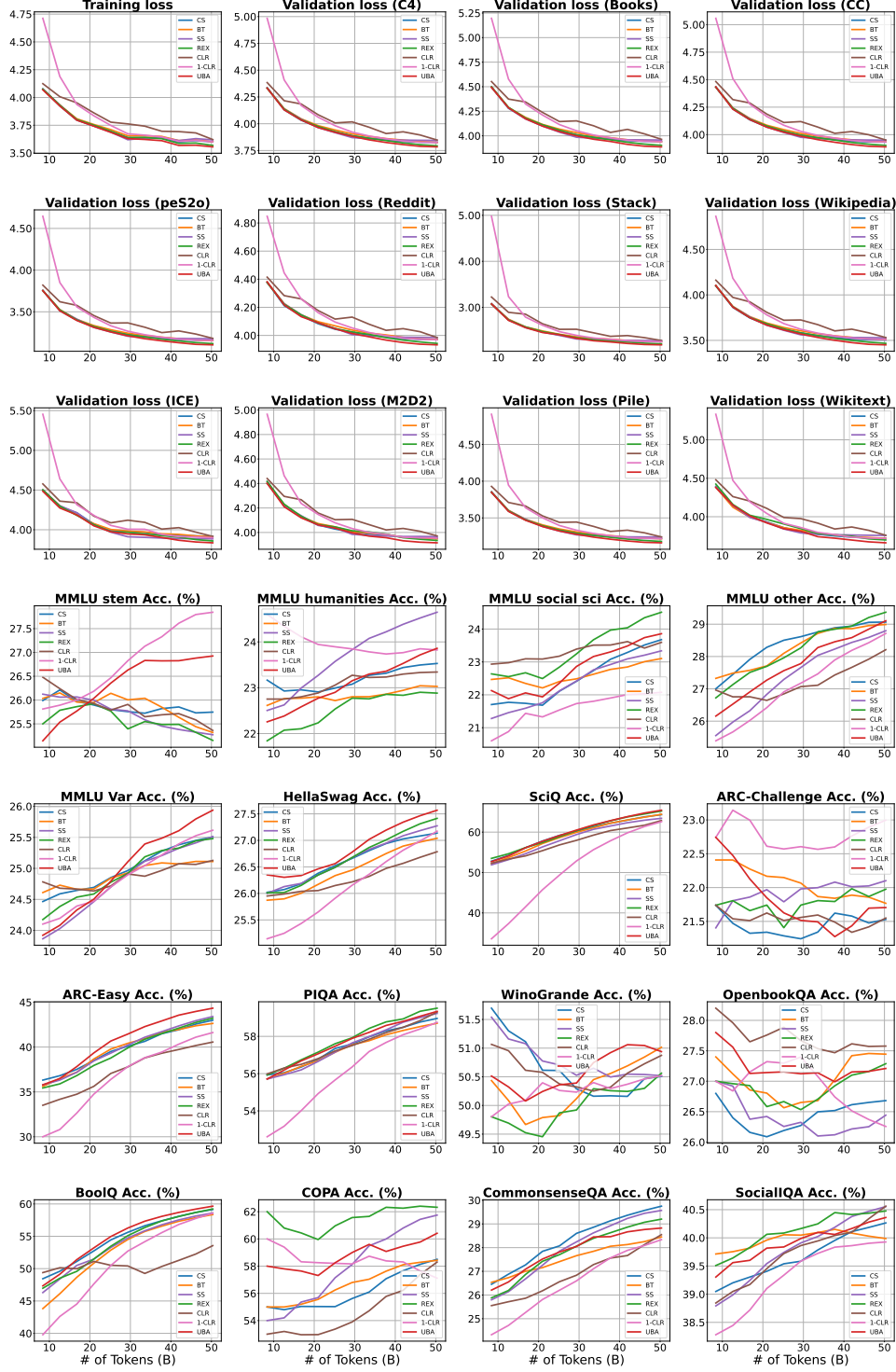


Figure 7: Overall performance for language tasks on OLMo-36M.

## 1170 E.7 Parameter analysis of $\varphi$

1171 In this subsection, we perform a sensitivity analysis of the key parameter  $\varphi$  in equation (5) ,  
 1172 which controls the variation speed of learning rate. Our experiments systematically evaluate

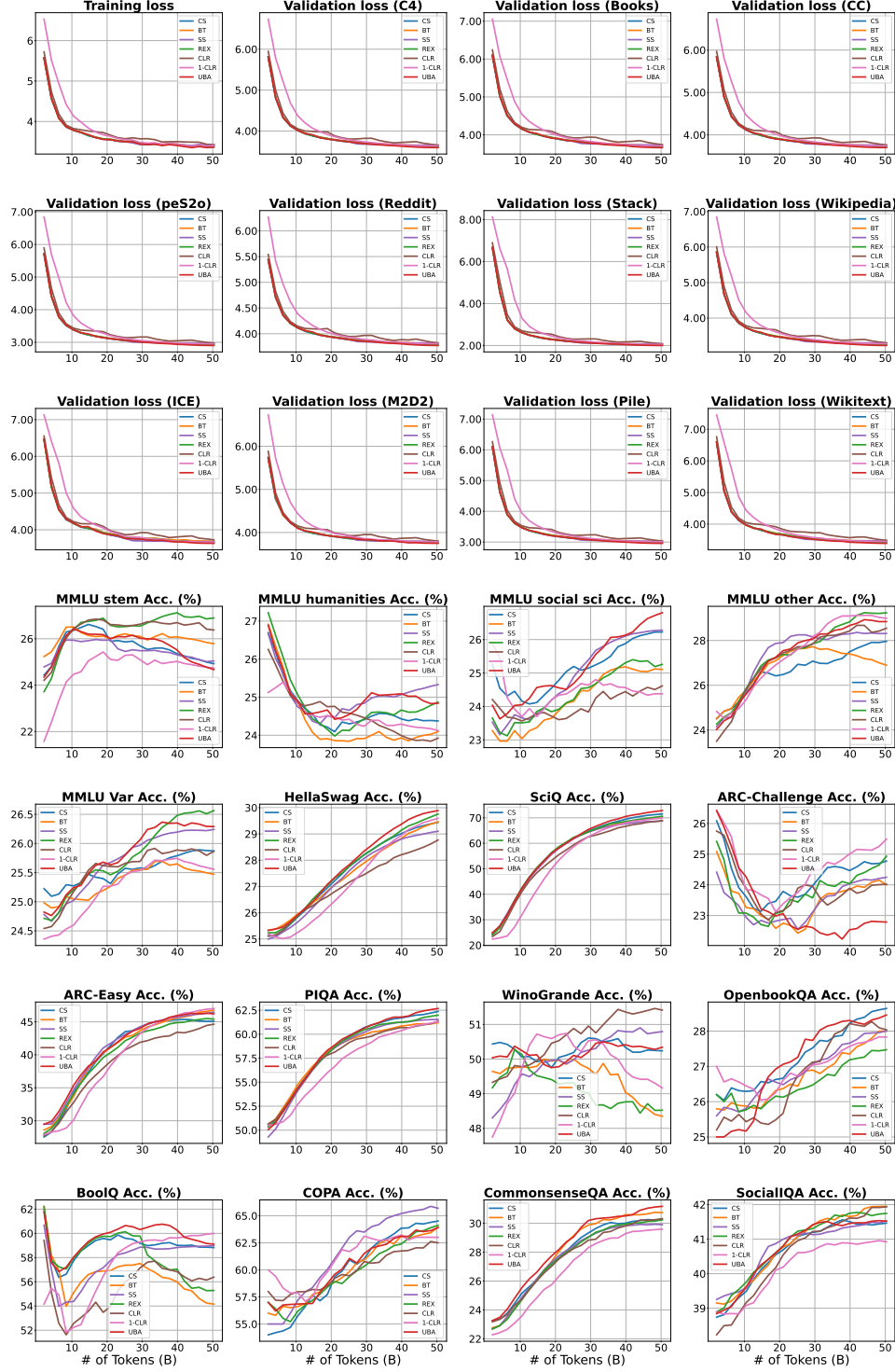


Figure 8: Overall performance for language tasks on OLMo-73M.

1173  $\varphi \in \{0.25, 0.5, 1.0, 2.5, 5, 10\}$  across different optimization scenarios. The results are plotted in  
 1174 Figure 11 and listed in Table 10).

1175 **Observations** From the experimental results, we find an interesting dichotomy of  $\varphi$ . Phenomenon(a): By choose proper  $\varphi$ , UBA demonstrates consistent performance gains over all base-  
 1176

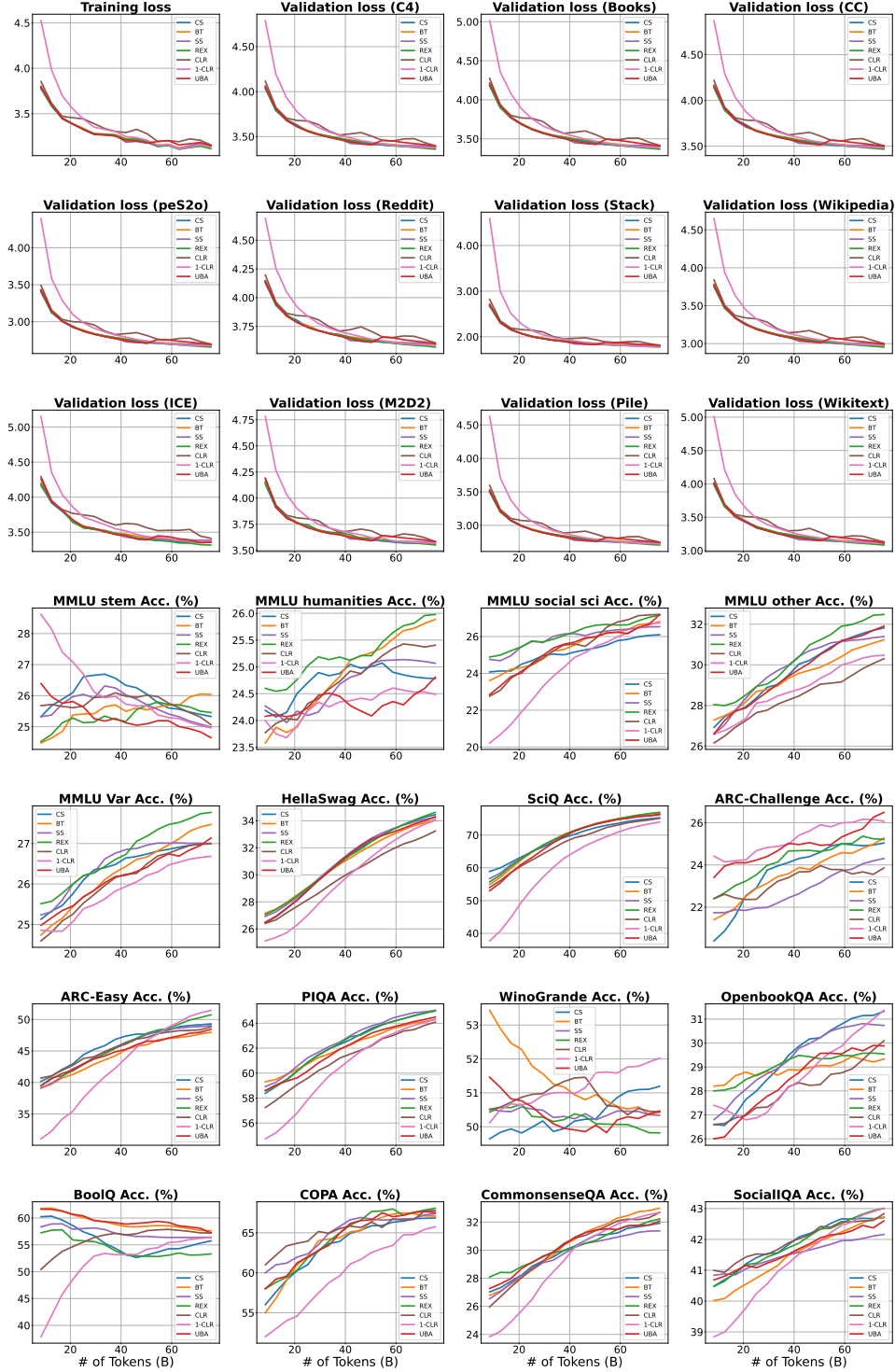


Figure 9: Overall performance for language tasks on OLMo-150M.

lines across the full spectrum of training budgets (25%, 50% and 100%), irrespective of optimizer choice (SGD/AdamW). This suggests the effectiveness of UBA is orthogonal to specific optimization strategies. Besides, with optimizer changed, UBA maintains comparable accuracy on the same dataset-architecture, which shows that UBA can exhaustively exploit the model's capacity regardless of optimizer variants. Phenomenon(b): As shown in Figure 11 and Table 10, on CIFAR100

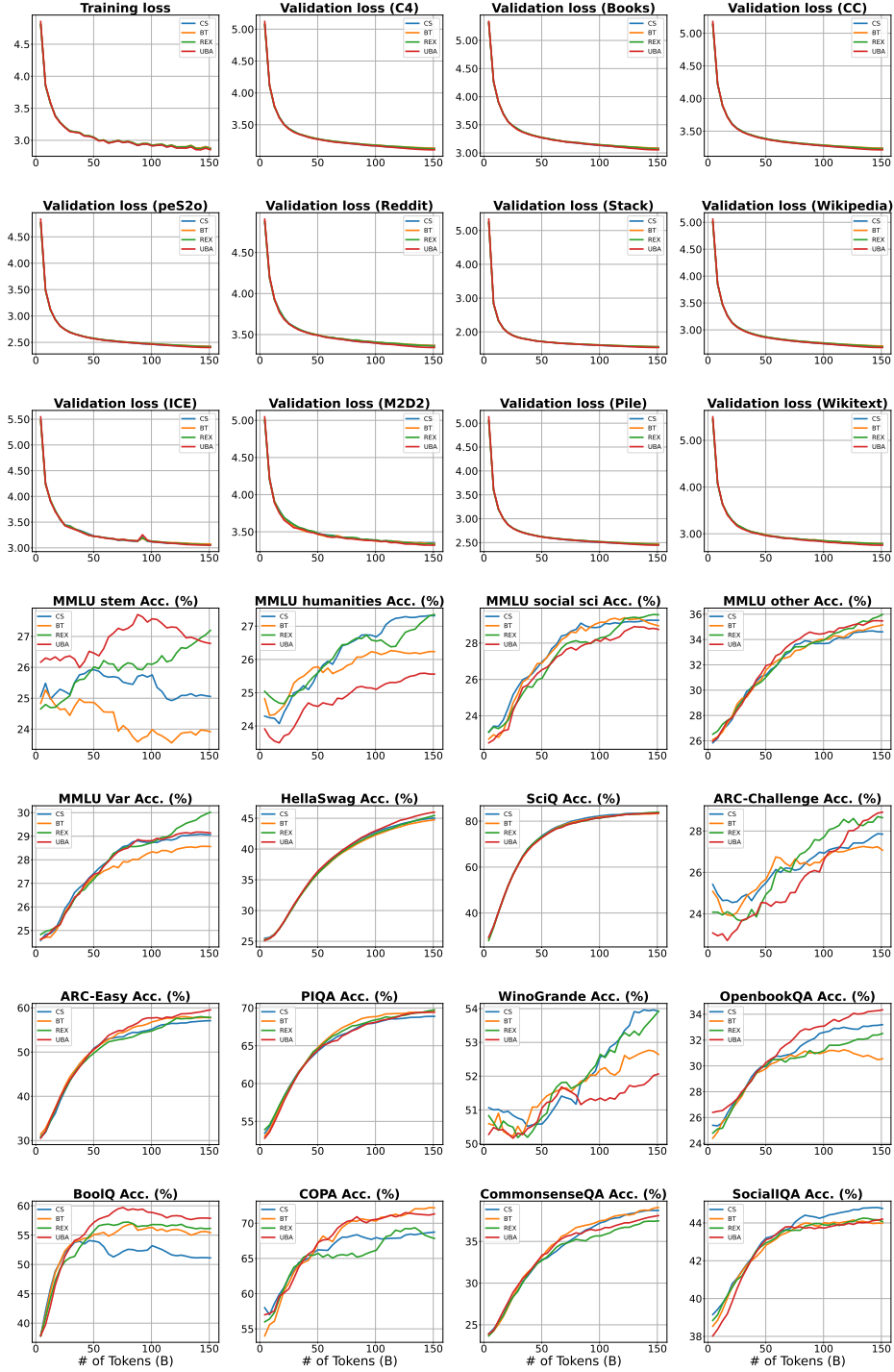


Figure 10: Overall performance for language tasks on OLMo-300M.

dataset, the best performance for AdamW is achieved with  $\varphi = 1$ , whereas SGD performs optimally  
with  $\varphi = 2.5$ . On ImageNet dataset, the best performance for AdamW is achieved with  $\varphi = 0.5$ ,  
whereas SGD performs optimally with  $\varphi = 10$ . Our experiments reveal a notable trend: when using  
AdamW, a smaller  $\varphi$  yields better performance, whereas a larger  $\varphi$  is preferred for SGD.

Table 8: Performance comparison on OLMo model

Size	Sched.	Benchmark(accuracy %)										
		PIQA	HSWAG	OBQA	SciQ	ARC-E	ARC-C	COPA	SIQA	SOC	OTH	Avg.
36M	CS	59.74	27.36	26.80	66.50	44.74	21.74	60.00	40.58	24.33	29.09	40.09
	BT	59.41	27.33	27.40	67.20	43.68	21.40	59.00	39.82	23.48	29.13	39.79
	SS	60.12	27.63	27.20	65.30	45.26	22.41	63.00	40.89	23.87	29.62	40.53
	REX	60.17	27.82	<b>27.80</b>	68.10	45.09	22.41	62.00	40.63	<b>25.18</b>	30.02	40.92
	CLR	<b>61.15</b>	27.21	27.60	67.50	42.11	22.07	62.00	<b>41.45</b>	24.29	29.38	40.48
	1C	60.17	27.91	25.80	67.40	43.68	<b>23.41</b>	55.00	40.02	22.26	29.86	39.55
	UBA	60.39	<b>27.98</b>	27.40	<b>68.20</b>	<b>45.79</b>	21.74	<b>63.00</b>	40.69	24.33	<b>30.14</b>	<b>40.97</b>
73M	CS	63.06	29.68	28.80	72.60	45.09	25.08	65.00	41.56	26.24	28.17	42.53
	BT	61.70	29.79	28.00	71.40	47.54	23.41	<b>66.00</b>	<b>41.97</b>	25.05	26.48	42.13
	SS	61.53	29.30	28.20	69.40	47.19	24.41	65.00	41.76	26.32	28.33	42.14
	REX	62.46	<b>30.22</b>	27.60	72.20	45.09	26.09	65.00	41.81	25.52	<b>29.34</b>	42.53
	CLR	62.30	29.38	27.80	70.40	45.44	24.08	62.00	42.02	25.05	29.05	41.75
	1C	61.81	29.90	27.80	71.10	<b>47.54</b>	<b>26.42</b>	63.00	40.79	24.37	28.81	42.15
	UBA	<b>63.17</b>	30.09	<b>28.80</b>	<b>74.30</b>	45.79	22.74	65.00	41.45	<b>27.13</b>	28.85	<b>42.73</b>
150M	CS	<b>66.00</b>	35.19	32.00	76.60	49.82	25.42	67.00	42.84	26.24	32.27	45.34
	BT	64.85	34.95	29.80	78.20	48.95	26.42	67.00	42.99	26.83	31.87	45.19
	SS	65.45	34.76	30.60	77.10	49.65	24.75	68.00	42.37	26.58	31.59	45.09
	REX	65.56	<b>35.72</b>	29.40	78.30	52.46	25.08	<b>69.00</b>	43.30	27.68	32.64	45.91
	CLR	65.07	34.76	31.60	77.20	50.70	25.08	68.00	<b>43.71</b>	27.30	31.23	45.47
	1C	65.29	35.16	<b>32.80</b>	76.30	<b>53.16</b>	25.75	67.00	43.45	27.64	30.62	45.72
	UBA	65.23	35.50	29.80	<b>78.30</b>	50.35	<b>27.42</b>	67.00	43.50	<b>29.29</b>	<b>32.72</b>	<b>45.91</b>
300M	CS	68.88	45.32	33.40	83.20	57.19	27.76	69.00	<b>44.58</b>	29.25	34.57	49.32
	BT	69.64	45.21	30.80	83.70	57.37	26.42	72.00	43.96	28.66	35.49	49.33
	REX	<b>70.18</b>	46.30	33.00	<b>84.40</b>	57.72	28.43	67.00	43.71	<b>29.50</b>	<b>36.74</b>	49.70
	UBA	69.48	<b>46.44</b>	<b>34.60</b>	83.90	<b>60.35</b>	<b>29.10</b>	<b>72.00</b>	44.42	28.53	35.41	<b>50.42</b>

**Analysis** Furthermore, we intend to explain the dual behavior of  $\varphi$  through theoretical analyses (Proposition 1 and Theorem 1). We attribute Phenomenon (a) to UBA’s optimal scheduling dynamics, governed by the  $\varphi$  value. Moreover, Phenomenon (b) reveals a fundamental dichotomy in  $\varphi$  value selection between optimizers. We attribute this phenomenon(b) to the preconditioning effect of AdamW. Unlike SGD, AdamW adapts the learning rate by scaling gradients with the square root of the second moment estimate  $\sqrt{v_t}$ . In regions where gradients  $g_t$  are large, the surrounding landscapes are steep. The denominator  $\sqrt{v_t}$  is also large, effectively reducing the learning rate. This adaptive behavior mimics preconditioning, implicitly lowering the condition number of the optimization landscape.

Our theoretical framework (Proposition 1) establishes a connection between parameter  $\varphi$  and the condition number of the landscape around local minima: smaller  $\varphi$  is favored for well-conditioned problems (low condition number), while larger  $\varphi$  is beneficial for large condition number scenarios. Since preconditioning effect of AdamW naturally mitigates ill-conditioning, the schedule with smaller  $\varphi$  is preferred. In contrast, SGD lacks such adaptive mechanisms, thus a larger  $\varphi$  is more suitable for this situation. This alignment between theory and experiment underscores the importance of tailoring  $\varphi$  to the optimizer’s characteristics. *Thus, we recommend selecting higher values of  $\varphi$  when facing challenging optimization difficulty, while employing lower  $\varphi$  values in scenarios with smoother optimization difficulty.*

## E.8 Performance across different periods

Our schedule has a periodic phase-based learning rate adjustment setting, where the learning rate at the  $k$ -th phase is dynamically determined by the  $k$ -th local minimum of the loss landscape. In the original conception, this design captures the optimization dynamics around successive local minima, with the hyper-parameter  $\varphi$  related to the difficulty of each phase. Specifically,  $\varphi$  controls the trade-off between exploration (large steps) and exploitation (small steps) within each phase.

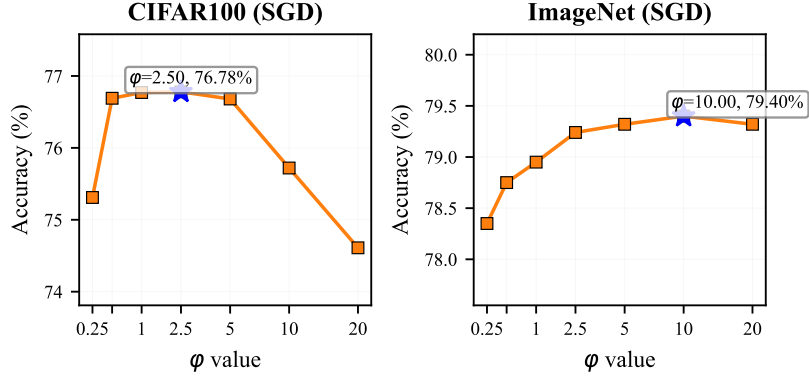
Table 9: Cross-optimizer ablation studies. We list validation accuracy for vision classification tasks across SGD and AdamW.

Dataset	Optimizer	Schedule	training budget (epoch(%))		
			30 (25%)	60(50%)	120(100%)
CIFAR100	SGD	SS	70.85	75.58	77.28
		CS	63.88	68.84	70.43
		CLR	72.63	73.81	74.80
		1C	73.72	75.53	77.90
		BT	72.53	75.40	78.49
		REX	73.12	75.48	77.99
		UBA(ours) $\varphi = 5$	<b>74.57</b>	<b>76.68</b>	<b>78.97</b>
-ResNet34	AdamW	SS	64.01	70.93	73.65
		CS	62.56	68.91	72.98
		CLR	62.87	70.13	73.22
		1C	64.27	71.60	74.06
		BT	64.01	71.44	74.34
		REX	<b>65.19</b>	71.72	74.08
		UBA(ours) $\varphi = 0.5$	<i>64.44</i>	<b>72.10</b>	<b>74.48</b>
			75 (25%)	150(50%)	300(100%)
ImageNet	SGD	SS	74.84	76.96	78.10
		CS	75.99	77.79	79.10
		CLR	72.86	74.88	76.91
		1C	75.28	77.37	78.79
		BT	75.71	77.48	78.66
		REX	75.28	77.03	78.46
		UBA(ours) $\varphi = 5$	<b>76.00</b>	<b>77.99</b>	<b>79.32</b>
-ResNet50	AdamW	SS	74.64	77.51	79.01
		CS	75.68	78.10	79.04
		CLR	74.93	77.20	78.76
		1C	76.38	78.11	79.21
		BT	76.10	78.14	79.05
		REX	76.42	78.28	78.86
		UBA(ours) $\varphi = 0.5$	<b>76.57</b>	<b>78.37</b>	<b>79.38</b>

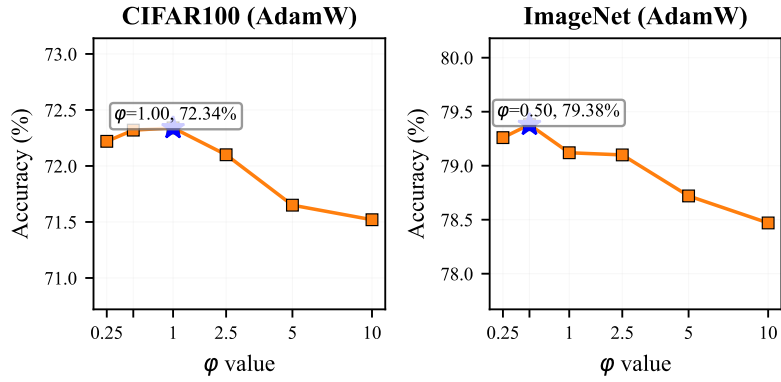
Table 10: Cross- $\varphi$  ablation studies.

Dataset	Network	Epoch	Optimizer	UBA					
				$\varphi=0.25$	$\varphi=0.5$	$\varphi=1$	$\varphi=2.5$	$\varphi=5$	$\varphi=10$
CIFAR100	ResNet34	60	SGD	75.10	75.89	76.31	76.40	<b>76.68</b>	75.64
			AdamW	<b>72.10</b>	71.34	71.54	71.21	70.40	69.01
ImageNet	ResNet50	300	SGD	78.35	78.75	78.95	79.24	79.32	<b>79.40</b>
			AdamW	79.26	<b>79.38</b>	79.12	79.10	78.72	78.47

To validate our scheduling strategy, we conduct experiments by varying  $K$  (see detailed results in Appendix E.8). We observe that when roughly setting  $\varphi$  as a constant in each phase, performance degrades as training progresses. This suggests that a fixed  $\varphi$  strategy cannot adapt to the changing landscape of multi-phase scheduling. Since the optimization difficulty should decrease during training process, we drop  $\varphi$  as phase increases. By finely evaluate  $\varphi$  in each phase, it achieves performance improvement, confirming the need for dynamic control. It makes sense since multi-phase captures the dynamic features of the loss surface more finely, it needs careful selection for  $\varphi$ , where  $\varphi$  reflects optimization difficulty. These results highlight a fundamental trade-off: when  $\varphi$  values per phase for multi-phase scheduling can be finely evaluated, multi-phase scheduling better captures the loss landscape's non-stationary behavior than single-phase scheduling. However, selecting optimal



(a)



(b)

Figure 11: Visualization of cross- $\phi$  performance.

1220  $\phi$  values per phase for multi-phase remains non-trivial, which motivates future work on automated  
 1221 landscape-aware  $\phi$  tuning.

Table 11: Performance on multi-phase version UBA.

Dataset	Network	$\phi$	Original	Multi-phase		
				K=4	K=6	K=8
CIFAR100	ResNet34	5	76.68	76.54	75.37	75.05
		$5 \times 0.8^k$		76.31	76.60	76.45



## 1222 F Related work

1223 **Budgeted training** Researchers face significant challenges in achieving optimal model perfor-  
 1224 mance under fixed hardware and limited time. To address this, budgeted training has emerged as a  
 1225 broad research domain focusing on optimizing performance while minimizing resource usage. This  
 1226 research spans multiple aspects, including computation efficiency, model compression, training sta-  
 1227 bility, convergence improvement, and optimization [40]. By strategically integrating efficient train-  
 1228 ing approaches, budgeted training advances cost-effective model development, offering pathways to  
 1229 achieve high performance under realistic constraints.

1230 Budgeted training can be divided from three main perspectives: data, model, and optimization. From  
 1231 the data perspective, budgeted training emphasizes the allocation of resources between the dataset  
 1232 and the algorithms that process it, such as the balance between the model size and the amount of data  
 1233 [2, 6, 23, 27]. From the model perspective, budgeted training searches the optimal number of model  
 1234 parameters within the given compute budget [16, 25, 30, 37]. Beyond data and model perspective,  
 1235 optimization based methods guide budgeted training based on learning rate schedules [5, 29, 42, 43],  
 1236 batch size [16] and other weight averaging method [25, 26].

1237 Among these three aspects, optimization-based methods are particularly aligned with the objectives  
 1238 of budgeted-iteration training. In this domain, learning rate scheduling based methods are partic-  
 1239 ularly aligned with the objectives of budgeted-iteration training. Smith et al. [42] propose a new  
 1240 learning rate schedule, named cyclical learning rates (CLR). It improves accuracy in fewer itera-  
 1241 tions without tuning and is relevant to super-convergence phenomenon [43]. Li et al. [29] introduce  
 1242 an alternative setting of existing learning rate schedules for budgeted training. Chen et al. [5] pro-  
 1243 pose the reflected exponential schedule (REX) via a profile and sampling fashion. Learning rate  
 1244 based approaches achieve robust and high-performing results under various lengths of training it-  
 1245 erations, which corresponds to our purpose in this work, i.e. budgeted-iteration training. Besides,  
 1246 this approach is plug-and-play, requiring no substantial alterations to the underlying model structure,  
 1247 making it readily adaptable to various deep network frameworks. Therefore, we explore the proper  
 1248 learning rate schedule to achieve budgeted-iteration training.

1249 **Learning rate schedule** The learning rate plays a pivotal role in controlling the non-convex opti-  
 1250 mization process during network training. Generally, the learning rate is gradually adjusted progres-  
 1251 sively alongside the iterations, which can be represented as  $\eta = g(t)\eta_0$ , where  $g(t)$  is the schedule  
 1252 function which control the variation of learning rate,  $t$  is the current iteration and  $\eta_0$  is the initial  
 1253 learning rate.

1254 Learning rate schedules have been extensively studied to improve training efficiency and model per-  
 1255 formance. The common scheme is the step decay schedule. Its schedule function can be represented  
 1256 as  $g(t) = d_{\lfloor t/t_s \rfloor}$ , where  $t_s$  is the predefined decaying step and  $d_i (d_i \geq d_{i+1})$  is predefined decaying  
 1257 scalar. A typical instance decreases the learning rate by a decaying scalar 0.1 after 50% epochs and  
 1258 by a decaying scalar 0.01 after 75% epochs [20]. Then Loshchilov et al. [32] observe that sharp  
 1259 decreases may prevent models from escaping local minima. Thus they propose cosine schedule  
 1260 function as  $\frac{1}{2}(1 + \cos(\frac{t\pi}{t_s}))$ , where  $t_s$  is the predefined stage to restart the learning rate schedule,  
 1261 which has proven effective in transformer training. In addition, cosine schedule is a commonly-used  
 1262 schedule nowadays. Pan et al. [35, 36] further introduce a learning rate schedule called elastic step  
 1263 decay as  $\eta_t = \eta_0/2^k$ , if  $t \in [(1-r^k)T, (1-r^{k+1})T)$  to accelerate the pre-training for Berts, where  
 1264  $r$  and  $k$  are hyper-parameters that control learning rate interval. Hu et al. [24] propose a Warmup-  
 1265 Stable-Decay(WSD) for large model pretraining and promoting continuous training. WSD consists  
 1266 of three consecutive phases: warmup, stable constant, and decay. Luo et al. [33] discover a new  
 1267 schedule that outperforms the cosine schedule, resembling WSD but with lower final loss.

1268 **Adaptive learning rate** Compared to schedule methods, adaptive learning rate based methods fo-  
 1269 cus on the learning rate adaptation based on local loss landscape statistics. Typical methods includes  
 1270 AdaGrad [12], Adadelta [53], RMSprop [22] and Adam [28]. These methods have been shown sta-  
 1271 bilizing the training process while effectively optimizing learning rate. Then Loshchilov et al. [31]  
 1272 propose AdamW optimizer which improves the performance in the transformer training. Recently,  
 1273 Chen et al. [7] discovers Lion optimizer for more memory-efficiency. Xie et al. [51] propose the  
 1274 Adan optimizer which implements 50% faster training than Adam and AdamW on commonly-used  
 1275 networks. Despite these improvements, some studies suggest that adaptive methods may under-

1276 perform momentum SGD in terms of generalization [29, 50], a critical factor for budgeted-iteration  
1277 training.

1278 Learning rate design is not only significant in general training, but also critical in budgeted-iteration  
1279 training which still remains a topic of debate. Some analyses advocate for small, constant learning  
1280 rates to ensure stability and convergence [11]. On the contrast, one prevailing hypothesis suggests  
1281 that large learning rates may facilitate crossing over sharp local minima in the optimization land-  
1282 scape [55]. Despite the lack of comprehensive theoretical explanations, a range of learning rate  
1283 schedules inspired by the above analyses as heuristic guidelines has been widely adopted in practice,  
1284 using variable learning rates to budgeted-iteration training [5, 29]. In this work, we explore learning  
1285 rate schedule from optimization problem tailored to budgeted-iteration training, aiming to balance  
1286 iteration budget constraints and generalization.